Руководство пользователя

KOMPAS-Invisible (API КОМПАС-3D)

KOMPAS-Invisible (API КОМПАС-3D)

API КОМПАС-3D - это ориентированные на прикладного программиста инструментальные средства разработки приложений (библиотек конструктивов, прикладных САПР) на базе системы КОМПАС.

API КОМПАС-3D включает в свой состав 2D API и 3D API.

2D API обеспечивает доступ к системе КОМПАС для формирования и обработки двумерных графических документов. В его состав входят следующие варианты реализации:

1. Набор экспортных функций, оформленных в виде динамически подключаемых DLL-модулей.
2. Автоматизация.

3D API обеспечивает доступ к системе КОМПАС для создания и редактирования трехмерных моделей. В его состав входят следующие варианты реализации:

1. Стандартные COM-объекты. Использование COM-интерфейсов позволяет получить максимальную производительность системы.
2. Автоматизация.

Создание прикладных библиотек

Общие сведения о прикладных библиотеках системы КОМПАС

Прикладная библиотека представляет собой набор команд. Команды библиотеки можно вызывать следующими способами:

- использование меню библиотеки,
- использование кнопок библиотечной или пользовательской панели инструментов,
- использование механизма хот-точек,
- обработка событий, например, сдвиг, перемещение библиотечного элемента и т.п.,
- команды контекстного меню **Редактировать** для библиотечного элемента 3D,
- двойной щелчок мышью по библиотечному элементу в окне документа.

Идентификатор меню команд определяется с помощью предопределенной функции LIBRARYID. Если эта функция отсутствует, то считается, что библиотека имеет одну команду, имя которой совпадает с именем библиотеки.

Имя библиотеки определяется с помощью предопределенной функции LIBRARYNAME. Если имя библиотеки не определено, то ей присваивается имя **Неименованная библиотека**. Имя библиотеки, полученное при помощи функции LIBRARYNAME, передается в создаваемые библиотекой макроэлементы. При редактировании таких макроэлементов по двойному щелчку мыши или через интерфейс хот-точек, система находит библиотеку для редактирования по имени файла, имени библиотеки, по номеру команды.

Функция DisplayLibraryName позволяет получить имя библиотеки, которое будет отображаться на экране во время работы системы КОМПАС (в меню, в менеджере библиотек, на панелях). Функция DisplayLibraryName является необязательной. Если эта функция не используется, то в качестве имени библиотеки будет отображаться возвращаемое значение функции LibraryName.

Отображаемых команд может быть не более 1000. Номера команд должны лежать в диапазоне 1...1000. В ресурсном файле для каждой команды можно определить слайд или битмап для отрисовки в окне менеджера библиотек, в окне или диалоге библиотеки. Чтобы слайд отрисовывался, нужно определить размер окна слайда с помощью предопределенной функции LIBRARYBMPSIZE, размер слайда произвольный. Этот же слайд или битмап масштабируется и отрисовывается в виде иконки размером 22 на 22 пиксела рядом с именем команды в менеджере библиотек, если задан режим отображения библиотеки **Панель**. Идентификаторы слайдов равны номерам соответствующих команд.

Для каждой команды можно определить значок на кнопке Панели инструментов. Для использования значков нескольких размеров, в библиотеке следует определить функцию LibraryBmpBeginID.

Если она не будет определена, система будет использовать старую систему использования значков (интервал идентификаторов значков начинается с 1000, значки автоматически масштабируются к нужному размеру).

Если такой значок не определен, то кнопку вызова команды разместить на Панели инструментов нельзя.

Для команд меню библиотеки или кнопок вызова команд на инструментальных панелях может быть организован вызов разделов справочной системы библиотеки.

Типы библиотек системы КОМПАС

- Простые библиотеки. Такие библиотеки пользователь может отключать или подключать к системе КОМПАС по собственному желанию. Библиотеки отображаются в менеджере библиотек.
- Библиотеки-добавления типа Addins. Такие библиотеки подключаются к системе КОМПАС во время запуска системы автоматически. В менеджере библиотек не отображаются, но видны в меню **Библиотеки**. Для того, чтобы библиотека стала добавляемой, нужно зарегистрировать ее в реестре Windows определенным образом.
- Библиотеки-конверторы. Такие библиотеки регистрируются определенным образом в реестре и должны реализовывать интерфейс IKompasConverter (см. sdk\libs\ConvertLibInterfaces.tlb и пример MyConverter).

  В системе КОМПАС под команды библиотек зарезервировано 10000 уникальных номеров. Максимальное число одновременно подключенных библиотек равно 25. Максимальное число инструментальных панелей библиотек равно 50.

Возможные состояния библиотеки системы КОМПАС

Прикладная библиотека может находиться в следующих состояниях.

- Библиотека не подключена к системе КОМПАС. При этом библиотека может отображаться или не отображаться в менеджере библиотек.
- Библиотека подключена к системе КОМПАС. В менеджере библиотек рядом с именем такой библиотеки включена «галочка». Имя библиотеки отображается в меню **Библиотеки**. Если библиотека подключена, она захватывает соответствующее количество уникальных номеров для отображаемых команд и панелей. Если для запуска библиотеки необходима оплаченная лицензия, библиотека захватывает экземпляр лицензии.
- Библиотека запущена на выполнение. Библиотека считается запущенной в следующих случаях:
  - пользователь выполняет команду библиотеки,
  - пользователь редактирует библиотечный элемент по двойному щелчку мыши или при помощи хот-точек,
  - библиотека подписалась на события системы КОМПАС.

Описание значков на кнопках инструментальных панелей

1. Интервалов идентификаторов значков кнопок может быть несколько. Значения идентификаторов должны быть больше 1000, так как интервал от 0 до 1000 предназначен для ID слайдов, отображающихся в менеджере библиотек.
2. Поддерживаются следующие размеры значков (в пикселах):

- 16*16,
- 24*24,

- 32*32,
- 48*48.

3. Глубина цвета значков может составлять 24 бита (TrueColor).

   Примечание: Цвет RGB: 192, 192, 192 считается прозрачным.

4. Для значка собственно панели команд, отображаемой на компактной панели, достаточно создать файл значка с расширением ico, содержащий значки размеров 12*12, 18*18, 24*24, 36*36 пикселов. Значки должны располагаться в файле в порядке возрастания размера. Глубина цвета значка должна составлять 24 бита (TrueColor). Файл может содержать один значок. В таком случае остальные типоразмеры значка будут получены масштабированием.

Рекомендации по созданию прикладных библиотек

Управление окнами, создаваемыми прикладной библиотекой

Если в библиотеке при вызове диалога параметр parent передан как NULL, то возможна ситуация, когда диалог не санкционированно закроется. Для предотвращения подобных случаев параметру parent нужно задавать значение дескриптора главного окна КОМПАС. Значение дескриптора можно получить, используя функции GetHWindow или KompasObject::ksGetHWindow.

Для библиотек, написанных на Visual C++, которые имеют свой WinApp, нужно использовать:

```
CWnd parent;
parent.Attach((HWND)::GetHWindow());
.... работа с диалогом
parent.Detach();
//---
```

Для библиотек без WinApp можно использовать:

```
CWnd::FromHandlePermanent((HWND)::GetHWindow());
```

Редактирование зеркально отраженных библиотечных макроэлементов

В Placement (см. ksDocument2D::ksPlacement, IPlacement) макроэлементов хранится флаг зеркальной симметрии объекта. Если при операциях, выполняемых базовым функционалом системы КОМПАС, происходит зеркальное отображение макроэлемента, имеющего Placement, то для полученного в результате операции макроэлемента флаг зеркальной симметрии автоматически инвертируется. Поэтому, если макроэлементы, создаваемые при помощи прикладных библиотек, будут в дальнейшем редактировать библиотеками, необходимо задавать этим макроэлементам Placement.

Макроэлементы, у которых не задан Placement и геометрия которых была зеркально отражена (например, операцией **Симметрия**), не могут быть корректно отредактированы

библиотеками - после редактирования их геометрия вернется в нормальное (не зеркальное) состояние.

Чтобы определить, является ли геометрия макроэлемента зеркально отображенной, можно воспользоваться функцией ksGetMacroPlacementEx (ksDocument2D::ksGetMacroPlacementEx). Если в параметре mirrorSymmetry (флаг зеркальной симметрии объекта) вернется ненулевое значение, то геометрия макроэлемента зеркально отображена.

Если редактирование при помощи библиотеки макроэлемента, имеющего Placement, reference на который получен через EditMacroMode (ksDocument2D::ksEditMacroMode), запускается двойным щелчком мыши или через хот-точки, проверять, является ли он зеркально отображенным, и принимать какие-то другие меры для сохранения правильного отображения геометрии не нужно. Корректное отображение геометрии в данном случае обеспечится функционалом КОМПАС. В данном случае функции SetMacroPlacement и GetMacroPlacement можно использовать без ограничений.

Если при создании нового макроэлемента нужно, чтобы его геометрия была сразу зеркально отображена, то следует задать макроэлементу Placement функцией ksSetMacroPlacementEx (ksDocument2D::ksSetMacroPlacementEx), значение параметра mirrorSymmetry которой надо задать равным 1. Далее создание геометрии для такого макроэлемента не отличается от создании геометрии "нормального" макроэлемента, т.к. зеркальная трансформация геометрии производится функционалом системы КОМПАС-3D.

Если редактирование макроэлемента, запускаемое двойным щелчком мыши или через хот-точки, проводится путем удаления исходного макроэлемента и построения нового, то при необходимости сохранения зеркального построения геометрии следует выполнить следующие действия.

1. Получить у исходного макроэлемента признак зеркальной симметрии объекта (параметр mirrorSymmetry функции GetMacroPalcementEx).
2. Задать Placement нового макроэлемента при помощи функции ksSetMacroPlacementEx (ksDocument2D::ksSetMacroPlacementEx), передав в нее в параметре mirrorSymmetry полученный ранее флаг зеркальной симметрии объекта. Далее создание геометрии для такого макроэлемента не отличается от создании геометрии "нормального" макроэлемента, т.к. зеркальная трансформация геометрии производится функционалом КОМПАС.

Примечание:

Если в макроэлементе есть вставки рисунков, то следует учитывать, что в них так же хранится флаг зеркальной симметрии объекта, который обрабатывается независимо от аналогичного флага в макроэлементе.

Вызов контекстно-зависимой справки по командам прикладных библиотек

Контекстно-зависимая справка для команд меню библиотеки и кнопок вызова команд, расположенных на инструментальных панелях, может быть вызвана стандартным способом Windows. Для этого следует активизировать кнопку или команду и нажать клавишу F1.

Справочная система прикладной библиотеки может быть реализована в формате WinHelp или HTML Help. Имя файла справочной системы должно совпадать с именем файла библиотеки. Если имена файлов справочной системы и библиотеки не совпадают, имя файла справки должно быть задано функцией LIBRARYHELPFILE или LIBRARYHELPFILEW. Файл справки должен находиться в той же папке, в которой сохранен файл библиотеки, или функция LIBRARYHELPFILE (LIBRARYHELPFILEW) должна вернуть полное абсолютное имя файла справки.

При вызове справки выполняются следующие действия.

1. Система пытается открыть файл, возвращаемый функцией LIBRARYHELPFILE или LIBRARYHELPFILEW.
2. В случае неудачи система пытается открыть файл <имя файла библиотеки>.chm.
3. В случае неудачи система пытается открыть файл <имя файла библиотеки>.hlp.
4. Если ни один из файлов с указанными именами не найден, на экране появится сообщение справочной системы КОМПАС о недоступности справочной системы библиотеки.
5. Если файл найден, но в нем отсутствует необходимый раздел, который должен соответствовать идентификатору команды, на экране появится сообщение программы просмотра справки об отсутствии этого раздела.

Примечание:

Если для отображения библиотеки выбран режим Диалог, идентификатор, передаваемый в WinHelp, в этом случае соответствует идентификатору главного меню библиотеки.

Особенности работы с документом 2D в режиме редактирования макроэлементов

При необходимости контролировать из приложения состояние видов в чертеже, следует учитывать следующую особенность работы КОМПАС с видами, связанную с возможностью визуального редактирования состава макроэлементов.

- При входе в ручной режим редактирования макроэлемента в КОМПАСе создается служебный вид, который от обычного вида ничем не отличается (только имеет тот же номер что и вид редактируемого макрообъекта), и в который "перекладываются" объекты редактируемого макро. "Отличить" обычный вид от служебного можно по флагу IView1::EditMacroVisibleRegime или IKompasDocument2D1::EditMacroVisibleRegime.
- При выходе из режима служебный вид удаляется, при этом создается событие ksObject2DNotify::Delete.

Инструкция по работе через ODBC с базами данных ACCESS в 64–разрядных приложениях

В случае, если при запуске под КОМПАС-3D x64 библиотека не может открыть свою ACCESS-базу, рекомендуется установить драйверы ODBC для x64 (Microsoft.ACE.OLEDB.12.0)http://www.microsoft.com/en-us/download/details.aspx?displaylang=en&id=13255. По умолчанию в ОС x64 они не установлены.

Для установки 64-разрядных драйверов требуется удалить 32-разрядные. Одновременно они работать не могут (см. http://msdn.microsoft.com/ru-ru/library/office/ff965871.aspx#DataProgrammingWithAccess2010_using32vs64ace. Соответственно и 32-разрядные приложения не могут работать с ODBC. Если установлена 32-разрядная версия приложения Microsoft Office, то требуется удалить и его.

В КОМПАС-3D для решения проблемы используется вспомогательный 32-разрядный exe-файл Wow32Util.exe, находящийся в каталоге Bin 64-разрядного КОМПАСа. Его назначение - создавать 32-разрядные интерфейсы, используя 32-разрядные драйверы.

Работает следующим образом:

64-разрядный КОМПАС сначала пытается сам создать нужный интерфейс через CoCreateInctance и выполнить соединение через Microsoft.ACE.OLEDB.12.0 и Microsoft.Jet.OLEDB.4.0. Если это не получается, то через CoCreateInctance поднимается Wow32Util.exe и в интерфейсе, реализованном в данной утилите, вызывается функция создания нужного интерфейса, куда передается нужный classID или GUID, затем повторяется попытка соединения через оба драйвера.

Этот механизм работает, хотя время импорта ниже чем при обычном подключении.

Оптимизация процесса перерисовки в чертежах и фрагментах

Если в результате работы приложения нужно перерисовывать изображение в чертеже, то рекомендуется обновлять только его измененную часть, а не весь чертеж целиком, т.к. в случае большого (насыщенного) чертежа, полная его перерисовки может потребовать длительного времени.

Для перерисовки части изображения чертежа можно использовать функцию ksReDrawDocPartEx. Следует при этом учитывать, что функция обновляет заданную область во всех окнах перерисовываемого документа.

Для обновления только одного окна документа, можно воспользоваться функцией WinAPI — InvalidateRect. (HWND окна документа можно получить через IDocumentFrame::GetHWND).

Создание контекстной панели для библиотечных макроэлементов и работа с ней

В 2D для библиотечных макроэлементов существует возможность создания "своей" контекстной панели. Наличие для макроэлемента специфической для него контекстной панели повышает удобство и скорость вызова команд для работы с этим макроэлементом. Состав контекстной панели полностью определяется библиотекой, аналогично определению состава библиотечной панели инструментов.

Состав контекстной панели формируется так же, как и для обычной панели команд - через возврат идентификатора панели в функции LibToolBarId;

nt WINAPI LibToolBarId(int barType, // Тип запрашиваемой панелей (0 - компактная панель, 1 - простая инструментальная панель, 2 - контекстная)

int index); // Индекс панели.

Чтобы библиотека могла определить, что команда вызвана из контекстной панели, для команд контекстной панели необходимо задать свой диапазон идентификаторов.

На контекстной панели располагаются кнопки команд библиотеки.

В контекстной панели может быть два ряда кнопок: для верхнего index - 0, для нижнего - 1.

Можно оставить только один (верхний) ряд кнопок или вообще запретить показ панели, возвратив для обоих индексов 0.

Управление доступностью кнопок контекстной панели происходит через функцию LibCommandState.

Если библиотека не определяет свою контекстную панель, то показывается контекстная панель, определенная в КОМПАС.

Примечание:

Отображение контекстной панели подчиняется так же системной настройке **Система – Общие – Контекстная панель**.

Пример обработки динамического запроса при создании панели инструментов

В качестве примера используется процесс указания точки и угла Placement.

Процесс содержит системную панель, элементы управления которой позволяют задать два параметра - точку и угол. Можно выделить 3 состояния процесса:

- указание точки,
- указание угла,
- все задано.

Если процесс переходит в состояния **все задано,** то возможны два варианта.

Автоматически запускается создание объекта.

Процесс ожидает ручного нажатия кнопки **Создать объект**. При этом никакие функции CallBack не выполняются. В API за это отвечает свойство IProcessParam::AutoReduce,

Таким образом, создавая панель, содержащую кнопку **Автосоздание объекта** с возможностью управлять ее состоянием (нажата/отжата) через ButtonUpdate, необходимо в первый раз синхронизировать состояния кнопки и флага IProcessParam::AutoReduce.

Предположим, что процесс запускается с нажатой кнопкой автосоздания. Необходимо задать значение IProcessParam::AutoReduce, равное TRUE. Поля задания точки и угла на панели расфиксированы. Первый щелчок мыши в поле чертежа фиксирует точку, второй фиксирует угол и по флагу IProcessParam::AutoReduce запускается создание объекта. После этого поля задания точки и угла расфиксируются, то есть процесс переходит в начальное состояние (благодаря флагу IProcessParam::AutoReduce=TRUE).

В случае нажатия кнопки автосоздания в процессе, событие придет в библиотеку, и инвертирует флаг IProcessParam::AutoReduce в самом процессе автоматически.

Пользовательская панель может содержать только элементы управления, а не поля задания параметров. На построения они не влияют, так как таблица состояний статическая и описывается до запуска процесса. Таким образом, если процесс перешел в состояние **все задано**, то есть значения всех параметров установлены и на пользовательской па-

нели можно только отрабатывать изменение состояния элементов управления, никакие CallBack - функции не выполняются.

У процесса указания точки Cursor аналогичная схема работы, но задаваемый параметр всего один и первый же щелчок мыши в поле чертежа переводит процесс в состояние **все задано**.

Для имитации работы процесса Cursor с постоянным присутствием вызовов CallBack'ов необходимо выполнить следующие действия:

1. На старте процесса задать значение IProcessParam::AutoReduce, равное TRUE.
2. В событии ButtonClick на нажатия кнопки автосоздания всегда возвращать FALSE, то есть не давать процессу инвертировать значение IProcessParam::AutoReduce.
3. Управлять состоянием кнопки в ButtonUpdate по собственному усмотрению.
4. Всю логику работы реализовать в функции обратной связи CallBack, то есть, если заданы не все значения параметров, то возвращать 1, а если все, то 0.

Использование Unicode

Что такое Unicode?

Unicode - стандарт кодирования символов, позволяющий представить знаки практически всех письменных языков, т.е. это уникальный код для любого символа, независимо от платформы, независимо от программы, независимо от языка.

Стандарт состоит из двух основных разделов: универсальный набор символов (UCS, Universal Character Set) и семейство кодировок (UTF, Unicode Transformation Format).

Коды в стандарте Unicode разделены на несколько областей. Область с кодами от U+0000 до U+007F содержит символы набора ASCII с соответствующими кодами. Далее расположены области знаков различных письменностей, знаки пунктуации и технические символы. Часть кодов зарезервирована для использования в будущем. Под символы кириллицы выделены коды от U+0400 до U+052F (см. http://ru.wikipedia.org/wiki/Кириллица_в_Юникоде.

Имеется несколько форм представления: http://ru.wikipedia.org/wiki/UTF-8, http://ru.wikipedia.org/wiki/UTF-16 (UTF-16BE, UTF-16LE) и UTF-32 (UTF-32BE, UTF-32LE)

В системах Windows 2000 и XP используется двухбайтовая форма UTF-16LE для внутреннего представления имен файлов и других системных строк. В UNIX-подобных операционных системах GNU/Linux, BSD и Mac OS X принята форма UTF-8 для файлов и UTF-32 или UTF-8 для обработки символов в оперативной памяти.

Начиная с Windows 2000, служебная программа «Таблица символов» позволяет вывести на экран таблицу всех символов от U+0000 до U+FFFF, поддерживаемых конкретным шрифтом. Эта программа позволяет выделять отдельные символы и копировать их в буфер обмена. Более универсальный способ ввода символа, код которого известен - зажать *<Alt>,* нажать клавишу «плюс» в дополнительном блоке клавиатуры, и затем набрать шестнадцатеричный код требуемого символа. Например, нажатие *<Alt>*+*<Plus>*+*<F1>* вставит букву «с». Этот способ, однако, работает не во всех элементах управления, позволяющих вводить текст.

Как любая изобретенная человеком система, Unicode не свободен от недостатков, например, некоторые системы письма все еще не представлены должным образом, а еще файлы с текстом в Unicode занимают больше места в памяти, так как один символ кодируется не одним байтом, как в различных национальных кодировках, а последовательностью байтов (исключение составляет UTF-8 для языков, алфавит которых укладывается в ASCII).

Ссылки:

http://www.unicode.org/standard/translations/russian.html

http://ru.wikipedia.org/wiki/Unicode

Конвертирование исходного кода (C/C++) из ANSI в Unicode

1. Код "двойного назначения"

   Чтобы иметь возможность реализации приложения в двух вариантах - ANSI и Unicode, следует обеспечить возможность компиляции "двойного назначения". Для этого в проект необходимо добавить новую конфигурацию "Unicode", в свойствах компилятора которой объявить два определения UNICODE и _UNICODE.

2. Типы данных

   Тип данных символа Unicode – wchar_t, в отличие от простого char - это 16-битное число! Для возможность компиляции "двойного назначения" (ANSI и Unicode) необходимо использовать тип (макрос) TCHAR, который в зависимости от объявленных определений разворачивается либо в char, либо в wchar_t.

   При замене char или char* необходимо понимать назначение переменной с таким типом. Если это строка или символ, то нужно использовать TCHAR, а если это бинарный массив или простой байт данных то byte.

3. Прототипы функций

   Код, содержащий явные вызовы str-функций, просто так компилировать с использованием и ANSI и Unicode нельзя. Чтобы реализовать возможность компиляции "двойного назначения", необходимо заменить все функции на _t-образные (макросы,) например strlen на _tcslen или sprintf на _stprintf. Эти макросы заменяют явные вызовы str- или wcs-функций в зависимости от определения _UNICODE.

   Так же происходит и с функциями Win32 API, например SetWindowText подразумевает либо SetWindowTextA, либо SetWindowTextW.

4. Символы и строки в С-коде

   По умолчанию компилятор транслирует строки как состоящие из символов ANSI, а не Unicode. Чтобы компилятор сгенерировал Unicode-строку перед ней надо добавить букву L, например L"text". Тогда, размещая строку в области данных программы, компилятор вставит между всеми символами нулевые байты. При этом возникает другая проблема - программа компилируется, только если _UNICODE определен. Для этого существует макрос _TEXT() или просто _T(), например _T("text").

5. В API системы КОМПАС все структуры параметров, содержащие строковые переменные имеют аналог для UNICODE с суффиксом W.

При использовании функций GetObjParam / SetObjParam для автоматического выбора структуры параметров используется определение с суффиксом T.

Например:

Тип данных:

```
#ifdef _UNICODE
#define ALLPARAM_T  ALLPARAM_W
#else
#define ALLPARAM_T  ALLPARAM
#endif // !UNICODE
```

Структура параметров:

```
//--------------------------------------------------------------------------
// Структура параметров вида (ANSI)
// ---
struct ViewParam {
unsigned short state;           // состояние вида
double        x,y;              // точка привязки вида
double        scale;            // масштаб вида
double        ang;              // угол поворота вида
unsigned long  color;           // цвет вида в активном состоянии
char          name[TEXT_LENGTH]; // имя вида
};

//--------------------------------------------------------------------------
// Структура параметров вида ( Unicode )
// ---
struct ViewParamW {
unsigned short state;           // состояние вида
double        x,y;              // точка привязки вида
double        scale;            // масштаб вида
double        ang;              // угол поворота вида
unsigned long  color;           // цвет вида в активном состоянии
wchar_t       name[TEXT_LENGTH]; // имя вида
};

#ifdef _UNICODE
#define ViewParamT  ViewParamW
```

```
#else
#define ViewParamT  ViewParam
#endif // !_UNICODE
```

Ссылки:

http://www.microsoft.com/globaldev/getwr/steps/wrg_unicode.mspx

Рекомендации по использованию метода IUnknown::QueryInterface

В данном разделе приведены примеры получения дополнительных интерфейсов с использованием метода IUnknown::QueryInterface для различных языков программирования.

Visual Basic

```
Dim kompasDoc As KompasAPI7.KompasDocument
Dim doc2D     As KompasAPI7.DrawingDocument

Set kompasDoc = KomApp.ActiveDocument
Set doc2D     = kompasDoc // Здесь выполнится QueryInterface.
```

Delphi

```
var

  kompasDoc : IKompasDocument;
  doc2D     : IDrawingDocument;
begin

  kompasDoc := KomApp.ActiveDocument;
  doc2D     := kompasDoc As IDrawingDocument;     // Здесь выполнится QueryInterface
 или
doc2D     := KomApp.ActiveDocument As IDrawingDocument; // Здесь выполнится QueryInterface
```

C++

В C++ для работы с интерфейсами удобнее всего использовать "умные указатели".

Они умеют выполнять QueryInterface и правильно выполняют захват и освобождение интерфейсов, используя методы AddRef и Release.

```
IKompasDocumentPtr kompasDoc( KomApp.ActiveDocument );
IDrawingDocumentPtr doc2D( kompasDoc );      // Здесь выполнится QueryInterface
```

или

```
IDrawingDocumentPtr doc2D( KomApp.ActiveDocument ); // Здесь выполнится QueryInterface.
```

C#

```
IKompasDocument kompasDoc = KomApp.ActiveDocument;
IDrawingDocument doc2D = (IDrawingDocument) kompasDoc;      // Здесь выполнится QueryInterface
```

или

```
IDrawingDocument doc2D = (IDrawingDocument) KomApp.ActiveDocument // Здесь выполнится QueryInterface
```

Python

```
kompasDoc = KomApp.ActiveDocument
doc2D = kompasDoc._oleobj_.QueryInterface(KAPI7.NamesToIIDMap['IDrawingDocument'], pythoncom.IID_IDispatch)
doc2D = API7.IDrawingDocument(doc2D)
```

Мастер создания библиотек

Мастер создания библиотек; общие сведения

Мастер создания библиотек позволяет создать заготовку для библиотеки КОМПАС. Он сохранен в файле LibraryWizard.awx.

Подключение мастера создания библиотек

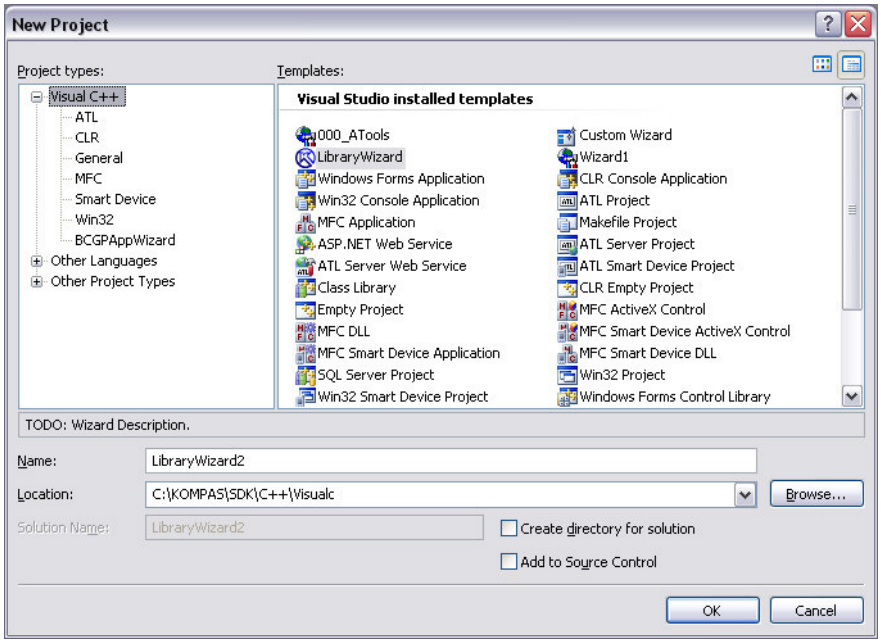
Для подключения мастера создания библиотек:

1. Скопируйте файлы **LibraryWizard.vsz, LibraryWizard.vsdir** и **LibraryWizard.ico** в папку *C:\Program Files\Microsoft Visual Studio 8\VC\vcprojects* (или в подпапку *vcprojects* папки, куда установлен Microsoft Visual Studio 2005).
2. Создайте в папке *C:\Program Files\Microsoft Visual Studio 8\VC\VCWizards* (или аналогичной) папку **LibraryWizard2005**.
3. Скопируйте в папку **LibraryWizard2005** папки: **1033**, **HTML, Images**, **Scripts** и **Templates**.

Создание заготовки библиотеки с использованием Мастера

Чтобы создать заготовку библиотеки с использованием Мастера, следует после запуска Microsoft Visual Studio 2005 выполнить следующие действия.

1. Вызовите команду **File – New – Project**. На экране появится окно нового проекта (см. рис.).

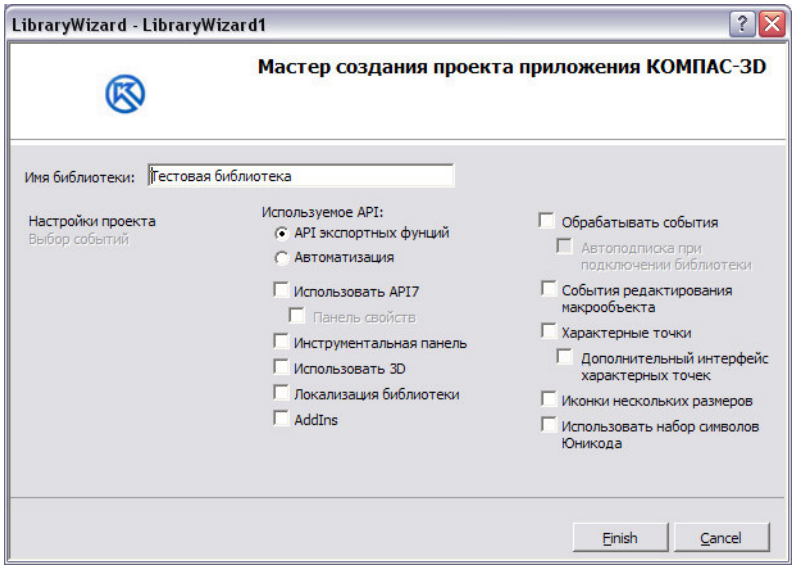


Окно создания нового проекта

2. В списке **Project types:** выделите пункт **Visual C++**, затем в списке **Templates:** выберите **LibraryWizard**.
3. Введите в поле **Name:** имя проекта.

4. В поле **Location:** задайте путь к Include папкам. (По умолчанию эти пути настроены для папки \SDK\C++\Visualc.)
5. Отключите опцию **Create directory for solution**.
6. Чтобы завершить задание свойств нового проекта, нажмите кнопку **OK**.

   На экране появится окно Мастера создания библиотек (см. рис.).



Мастер создания библиотек - Главное окно

Элементы управления окна позволяют задать параметры заготовки библиотеки.

В поле **Имя библиотеки:** следует ввести имя, которое будет отображаться в системе КОМПАС при подключении этой библиотеки.

Выберите тип API для программирования – **API экспортных функций** или **Автоматизация**.

При использовании **Автоматизации** добавляется функция для получения интерфейса **KompasObject – GetKompas**. Добавляется переменная **KompasObjectPtr kompas**.

При включении опции **Характерные точки** в проект добавляются файлы, позволяющие библиотеке работать в режиме редактирования характерных точек.

При включении опции **События редактирования** макрообъекта будет добавлен файл, позволяющий библиотеке обрабатывать события редактирования макрообъекта.

При включении опции **Использовать 3D** в файле stdfx.h будут сделаны настройки, позволяющие использовать в проекте константы 3D и константы событий 3D.

При включении опции **Использовать API7** в файле stdfx.h будут сделаны настройки, позволяющие использовать в проекте интерфейсы API7. Также добавляется функция для получения интерфейса **IApplication – GetNewKompasAPI** и переменная **IApplication newKompasAPI**.

При включении опции **Панель свойств** (она будет доступна только после включения опции **Использовать API7**) в проект будет добавлен класс для работы с панелью свойств и событиями панели свойств.

При включении опции **Панель команд** появляется возможность использования библиотеки в режиме «компактная панель команд», также в проект будет добавлена иконка для компактной панели и будут внесены необходимые изменения в файл resource.h.

При включении опции **Addins** библиотека будет содержать все необходимые функции для работы в этом режиме. Добавляются функции:

- **DllRegisterServer** - Регистрация библиотеки в реестре,
- **DllUnregisterServer** - Разрегистрация библиотеки.

Эти функции обеспечивают регистрацию библиотеки в разделе реестра "Software\\ASCON\\KOMPAS-3D\\AddIns\\имя_библиотеки".

Создаются ключи:

- **AutoConnect** = 1 - Загружать библиотеку при запуске Компас-3D,
- **Path** - Путь к файлу библиотеки; определяется в момент подключения библиотеки.

При включении опции **Локализации библиотеки** в проект добавляется файл словаря и специальным образом модифицируется функция для загрузки строк из ресурсов, что позволит в дальнейшем осуществить локализацию библиотеки.

Также имеется возможность отредактировать главное меню библиотеки – **Список команд**; команды библиотеки можно добавлять, удалять, переименовать и устанавливать очередность их появления в списке.

При включении опции **Обрабатывать события** в мастере создания библиотек появится возможность перехода к диалогу обрабатываемых событий (см. рис.). Перейти в этот диалог можно по ссылке **Выбор событий**. В диалоге обрабатываемых событий отображаются все возможные события в зависимости от настроек в Главном окне мастера.

В данном диалоге включите нужные опции. В проект будут добавлены все необходимые изменения и файлы, позволяющие библиотеке обрабатывать выбранные события.

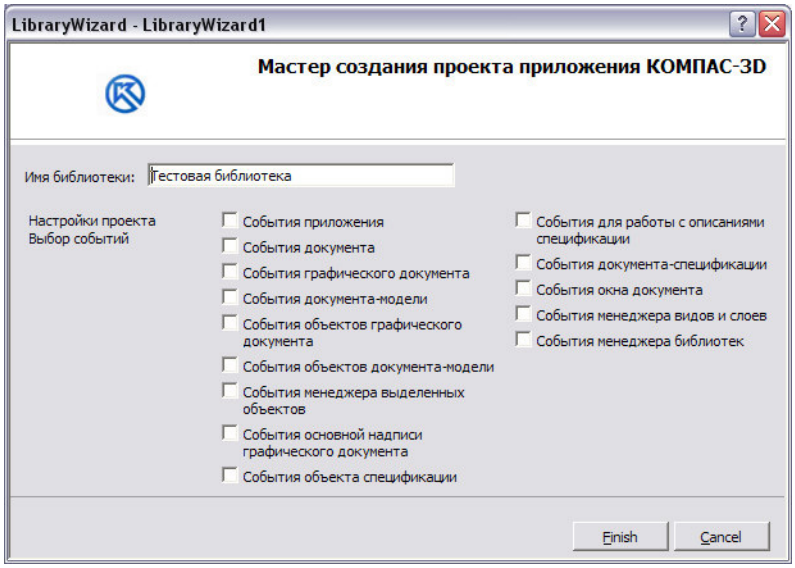
Чтобы завершить создание заготовки библиотеки, нажмите кнопку **Finish**. Чтобы закрыть окно Мастера, отказавшись от сделанных настроек, нажмите кнопку **Cancel**.

Создание прикладных библиотек в различных средах программирования

Delphi

Чтобы создать библиотеку, нужно выполнить следующие действия.

1. Вызвать команду **File – New....** Выбрать тип **DLL** на вкладке **NEW**.
2. В настройках проекта установить выравнивание: выключить опцию **Project Options – Compiler – Code Generation – Aligned Record fields**.
3. В настройках проекта или в дальнейшем после компиляции изменить расширение файла библиотеки с *dll* на *rtw*.



Мастер создания библиотек - выбор обрабатываемых событий

4. Для создания КОМПАС-библиотеки необходимо вставить несколько экспортных функций.

   Примеры реализации простейшей библиотеки расположены в папках:

- ..SDK\PASCAL\DELPHIAUTO\Step1 (библиотека написана с использованием Автоматизации);
- ..SDK\PASCAL\DELPHI\Step1 (библиотека написана с использованием экспортных функций).

Visual Basic

1. Вызвать из меню **File** команду **New – Project**, тип **ActiveX DLL**.
2. Для создания КОМПАС-библиотеки необходимо вставить несколько функций.
3. Библиотеки данного типа подключаются к КОМПАС на вкладке **ActiveX**, где их необходимо зарегистрировать при первом подключении, нажав кнопку **Добавить**.

   Пример реализации простейшей библиотеки расположен в папке ..SDK\Basic\Step1 (библиотека написана с использованием Автоматизации).

Visual C++

1. Вызвать из меню **File** команду **New.**
2. На вкладке **Projects** выбрать из списка вариант **MFC AppWizard (dll).** Ввести в поле **Project Name:** имя проекта и нажать кнопку **OK**.

3. В появившемся на экране диалоге **MFC AppWizard – Step 1 of 1** выбрать в группе **What type of DLL would you like to create?** вариант **MFC Extension DLL (using shared MFC DLL)**. Другие настройки этого диалога оставить без изменения и нажать кнопку **Finish**.
4. Вызвать команду **Project – Settings....**
5. В появившемся диалоге **Project Settings** раскрыть вкладку **C/C++**.
6. Выбрать из раскрывающегося списка **Category** вариант **General**. В поле **Preprocessor definition** подстроку _DEBUG заменить на NDEBUG.
7. Выбрать из раскрывающегося списка **Category** вариант **Code Generation**. Выбрать из раскрывающегося списка **Use run-time library** вариант **Multithread DLL**.
8. Закрыть диалог, нажав кнопку **OK**.
9. В настройках проекта установить выравнивание:

- выбрать из раскрывающегося списка **Category** вариант **Code Generation**;
- выбрать из раскрывающегося списка **Struct member argument** вариант **1Byte**.

10. Для создания КОМПАС- библиотеки необходимо вставить несколько функций.
11. В настройках проекта или в дальнейшем после компиляции изменить расширение файла библиотеки с *dll* на *rtw*.

    Пример реализации простейшей библиотеки расположен в папках:

- ..\SDK\C++\VisualcAUTO\Step1 (библиотека написана с использованием Автоматизации);
- ..SDK\C++\Visualc\Step1 (библиотека написана с использованием экспортных функций).

Сведения по настройке конфигурации проекта библиотеки в среде VC++ 2005 для платформы x64

1. Библиотеки SDK для платформы x64 по умолчанию размещаются в каталоге SDK\Lib64.
2. Дополнительные lib - файлы указываются только в свойствах проекта: **Configuration Properties – Linker – Input – Additional Dependencies**. В дереве проекта не должно быть подключенных lib - файлов.
3. Рекомендуется настроить разные пути для компиляции Win32 и платформы x64 во всех настройках проекта.
4. Примеры настроек конфигураций проектов SDK находятся в папках ..SDK\C++\Visualc и ..SDK\C++\VisualAuto.

   Примечание:

   Для создания библиотек на платформе x64 должны быть установлены компоненты Visual Studio - *X64 Compilers and Tools*.

C#

1. В свойствах проекта (**Configuration Properties – Build**) включите опцию **Register for COM Interop** (TRUE).
2. Создайте DLL-обертки для TLB Компас API (они расположены в папке ..\SDK\Lib) с помощью утилиты TlbImp.exe

3. Подключите созданные DLL к проекту, вызвав команду контекстного меню **Add Reference**.
4. Для регистрации библиотеки в системе КОМПАС класс библиотеки должен реализовывать статический метод типа .htmSample (Type type) с атрибутом ComRegisterFunctionAttribute, в котором для ключа реестра в HKCR, отвечающего за регистрацию класса, создается раздел с именем Kompas_Library.
5. Чтобы зарегистрировать библиотеку на компьютере пользователя, необходимо воспользоваться утилитой RegAsm.exe. Нажмите кнопку **Пуск**, вызовите команду **Выполнить...**, введите командную строку вида RegAsm.exe /codebase <файл_библиотеки> и нажмите кнопку **OK**.

C++ Builder

1. Вызвать из меню **File** команду **New....** Выбрать тип **DLL Wizard** на вкладке **NEW**. Другие настройки на этой вкладке оставить без изменения.
2. В настройках проекта установить выравнивание:
   Project Options – Advanced Compiler – Data Alignment = Byte.
3. В настройках проекта или в дальнейшем после компиляции изменить расширение файла библиотеки с *dll* на *rtw*.
4. Для создания КОМПАС-библиотеки необходимо вставить несколько экспортных функций.

   Пример реализации простейшей библиотеки расположен в папках:

- ..SDK\C++\CBUILDERAUTO\Step1 (библиотека написана с использованием Автоматизации);
- ..SDK\C++\CBUILDER\Step1 (библиотека написана с использованием экспортных функций).

КОМПАС–Макро

Существует возможность создания макросов средствами библиотеки КОМПАС-Макро.

Использование Библиотеки КОМПАС-Макро рассматривается в ее справочной системе.

Оформление прикладных библиотек типа DLL

Функции оформления библиотек типа DLL

В отличие от библиотек системы КОМПАС 4.x, имеющих собственный уникальный формат представления, библиотеки систем КОМПАС 3D последующих версий создаются с помощью стандартных систем программирования и оформляются в виде разделяемых DLL-библиотек WINDOWS, Они динамически подключаются при запуске приложения из среды КОМПАС-ГРАФИК. Файл библиотеки может иметь расширение RTW или DLL.

Чтобы создать библиотеку, следует использовать функции оформления библиотек. Они обеспечивают стыковку системы КОМПАС и приложения.

Функция LIBRARYENTRY является обязательной. Ее наличие позволяет системе КОМПАС идентифицировать произвольный DLL-файл как собственную библиотеку,

LIBRARYENTRY является головной функцией библиотеки (точкой входа) и ей передается управление при обращении к приложению. Остальные функции являются необязательными и позволяют определить дополнительные параметры приложения.

Все функции оформления библиотек должны быть объявлены в def файле как экспортные, например:

```
EXPORTS
LIBRARYID @1
LIBRARYENTRY @2
```

Если ни одна из этих функций не определена, то считается, что библиотека без имени и может выполнять только одну команду.

Пассивные клавиши (Dead keys)

Предназначены для дополнения раскладок клавиатуры национальными символами (например денежные знаки или буквы с точками вверху, как в немецком языке).

Dead key - это клавиша (сочетание клавиш), влияющая на последующую нажатую клавишу (deadable key).

Например, чтобы ввести символ **ã** , следует нажать последовательность клавиш: <CTRL>+<SHIFT>+~, a, то есть, нажать одновременно <CTRL>+<SHIFT>+~, отпустить их и нажать a (латинскую).

Комбинация <CTRL>+<SHIFT>+~ будет являться Dead key.

Пример простейшей программы в среде VC++, оформленной в виде библиотеки КОМПАС

```
#ifndef __LIBTOOL_H
#include «libtool.h»
#endif
char * WINAPI  LIBRARYNAME()
{
  return "Самая простая библиотека";
}
void WINAPI LIBRARYENTRY( unsigned int comm )
{
  Message("Строим отрезок!");
  LineSeg( 10, 10, 100, 100, 1 ); // x1, y1, x2, y2, тип линии - основная
}
```

Функция LIBRARYENTRY является точкой входа в библиотеку при вызове из КОМПАС. Наличие этой функции обязательно, так как именно ей передает управление КОМПАС

при обращении к библиотеке. В данном случае она обеспечивает выдачу сообщения и построение одного единственного объекта чертежа - отрезка.

Все функции оформления библиотек должны быть объявлены в def файле как экспортные, например:

```
EXPORTS
LIBRARYID @1
LIBRARYENTRY @2
```

Функции

CreateMacroFromSample – блок обработки команды "Объект по образцу"

Синтаксис:

extern "C" int WINAPI _export CreateMacroFromSample(long sourceReference);

Входной параметр:

sourceReference	- идентификатор (указатель reference) выбранного образца библиотечного макрообъекта.

Примечание:

1. При запуске команды **Объект по образцу** одновременно с другими элементами **КОМПАС-График** будут подсвечиваться и библиотечные макрообъекты, если создавшая их библиотека подключена и имеет блок обработки CreateMacroFromSample. Подсвеченные макрообъекты могут быть выбраны в качестве образца, после чего управление передается в блок обработки библиотеки.
2. В блок обработки команды **Объект по образцу** передается идентификатор (указатель reference) выбранного макрообъекта. По этому указателю на макрообъект можно получить параметры макрообъекта, обработать их и запустить соответствующую команду библиотеки.

DisplayLibraryName – Получить имя библиотеки,отображаемое при работе КОМПАС

Синтаксис:

extern "C" char * WINAPI __export DisplayLibraryName();

Возвращаемое значение:

- Имя библиотеки, отображаемое на экране во время работы системы КОМПАС.

Примечание:

1. Необязательная функция. Позволяет получить имя библиотеки, которое будет отображаться на экране во время работы системы КОМПАС (в меню, в менеджере библиотек, на панелях). Если эта функция не используется, то в качестве имени библиотеки будет отображаться возвращаемое значение функции LibraryName. Имя библиотеки, полученное при помощи функции LibraryName, передается в создаваемые библиотекой макроэлементы. При редактировании таких макроэлементов по двойному щелчку мыши или через интерфейс хот-точек, система находит библиотеку для редактирования по имени файла, имени библиотеки, по номеру команды. Если выходит новая версия библиотеки, в ней можно переопределить имя, отображаемое функцией DisplayLibraryName, оставив прежним имя, возвращаемое LibraryName. В этом случае новая библиотека может быть использована, чтобы редактировать макроэлементы, созданные старыми версиями библиотеки.
2. При использовании Unicode следует использовать функцию DisplayLibraryNameW.

DisplayLibraryNameW – Получить имя библиотеки, Unicode

Синтаксис:

extern "C" LPWSTR WINAPI __export DisplayLibraryNameW();

Возвращаемое значение:

- Имя библиотеки, отображаемое на дисплее во время работы КОМПАС.

Примечание:

Необязательная функция. Позволяет получить имя библиотеки, которое будет отображаться на дисплее во время работы системы КОМПАС (в меню, в менеджере библиотек, на панелях). Если эта функция не используется, то в качестве имени библиотеки будет отображаться возвращаемое значение функции LibraryName. Имя библиотеки, полученное при помощи функции LibraryName, передается в создаваемые библиотекой макроэлементы. При редактировании таких макроэлементов по двойному щелчку мыши или через интерфейс хот-точек, система находит библиотеку для редактирования по имени файла, имени библиотеки, по номеру команды. Если выходит новая версия библиотеки, в ней можно переопределить имя, отображаемое функцией DisplayLibraryName, оставив прежним имя, возвращаемое LibraryName. В этом случае новая библиотека может быть использована, чтобы редактировать макроэлементы, созданные старыми версиями библиотеки.

При использовании ANSI следует использовать функцию DisplayLibraryName.

GetLibToolbarSettings – Получить параметры инструментальных и компактных панелей

Функция не поддерживается

Синтаксис:

extern "C" int WINAPI _export GetLibToolbarSettings(int barId, LibToolbarSettings * settings);

Входной параметр:

barId - идентификатор панели.

Выходной параметр:

settings - указатель на структуру параметров библиотечной панели свойств.

Возвращаемое значение:

- Не используется.

Примечание:

Функция позволяет задать для панели признак отображения панели при активизации нужных типов документов. Если массив типов документов не задан, есть возможность сделать панель видимой по умолчанию.

LibGetDisableReason – Причина недоступности команды

Синтаксис

char * WINAPI LibGetDisableReason(unsigned int comm);

Входные параметры:

comm - номер команды.

Возвращаемое значение:

- Строка пояснение с причиной недоступности команды.
- Строка не удаляется.

LibGetDisableReasonW – Причина недоступности команды. Unicode

Синтаксис

wchar_t * WINAPI LibGetDisableReasonW(unsigned int comm);

Входные параметры:

comm - номер команды.

Возвращаемое значение:

- Строка пояснение с причиной недоступности команды.
- Строка удаляется функцией SysFreeString

LIBRARYNAME – задать имя библиотеки

Синтаксис:

extern "C" char * far_export pascal LIBRARYNAME()

Примечание:

1. Имя библиотеки, полученное при помощи функции LIBRARYNAME, передается в создаваемые библиотекой макроэлементы. При редактировании таких макроэлементов по двойному щелчку мыши или через интерфейс хот-точек, система находит библиотеку для редактирования по имени файла, имени библиотеки, по номеру команды.
2. После подключения библиотеки имя, полученное при помощи функции LIBRARYNAME, будет отражено в соответствующем меню системы КОМПАС. Если в библиотеке задана функция DisplayLibraryName, то во время работы системы КОМПАС в меню, в менеджере библиотек, на панелях будет отображаться имя библиотеки, полученное при помощи этой функции.
3. При использовании Unicode следует использовать функцию LIBRARYNAMEW.

LIBRARYNAMEW – Задать имя библиотеки (Unicode)

Синтаксис:

extern "C" LPWSTR WINAPI __export LIBRARYNAMEW()

Примечание:

1. После подключения библиотеки ее имя будет отражено в соответствующем меню системы КОМПАС.
2. При использовании ANSI следует использовать функцию LIBRARYNAME.

LibIsOnApplication7 – Задать тип версии API, используемого библиотекой

Синтаксис для C/C++:

extern "C" int WINAPI __export LibIsOnApplication7();

Синтаксис для PASCAL:

function LibIsOnApplication7() : Integer; stdcall;

Возвращаемое значение:

1 - библиотека использует API версии 7,
0 - библиотека использует API версии 5.

Примечание:

Необязательный метод оформления библиотеки. Позволяет задать тип интерфейса, который будет передан библиотеке в параметре application метода LibInterfaceNotifyEntry. Если метод в библиотеке не объявлен, то считается что библиотека использует API версии 5.

LIBRARYENTRY – Головная функция библиотеки

Синтаксис:

extern "C" void far_export pascal LIBRARYENTRY(UINT Comm)

Входной параметр:

Comm - идентификатор выбранной команды.

Примечание:

1. При вызове команды библиотеки из меню, панели инструментов или при редактировании библиотечного элемента по двойному нажатию кнопки мыши управление передается этой функции с номером выбранной команды.
2. Библиотека должна обязательно содержать предопределенную функцию LIBRARYENTRY, которая определяет точку входа в приложение или LibInterfaceNotifyEntry. При обращении к библиотеке именно ей передается управление. Кроме того, ее наличие позволяет системе КОМПАС идентифицировать произвольный DLL-файл как собственное приложение.

LibInterfaceNotifyEntry – Головная функция библиотеки. Подписка на обработку событий от системы

Синтаксис:

int LibInterfaceNotifyEntry (IDispatch *application);

Входной параметр:

application - указатель интерфейса приложения KompasObject (для COM не используется).

Возвращаемое значение:

1 - в случае успешного завершения (подписка прошла успешно),
0 - в случае неудачи.

Примечание:

Библиотека должна обязательно содержать предопределенную функцию LibraryEntry, которая определяет точку входа в приложение, или LibInterfaceNotifyEntry. При обращении к библиотеке именно ей передается управление. Кроме того, ее наличие позволяет системе КОМПАС идентифицировать произвольный DLL-файл как собственное приложение.

LIBRARYENTRYDEMO – Головная функция библиотеки (демонстрационный режим)

Синтаксис:

extern "C" void far _export pascal LIBRARYENTRYDEMO (UINT Comm)

Входной параметр:

comm - идентификатор выбранной команды.

Описание:

Если задан демонстрационный режим работы (демоверсия КОМПАС), эта функция (если она найдена в библиотеке), будет вызываться вместо функции LIBRARYENTRY.

LIBRARYENTRYDEMOEX– Головная функция библиотеки (демонстрационный режим)

Синтаксис:

extern "C" void far _export pascal LIBRARYENTRYDEMOEX (UINT comm)

Входной параметр:

comm - идентификатор выбранной команды.

Описание:

Если задан демонстрационный режим работы (не найден ключ защиты библиотеки), эта функция (если она найдена в библиотеке), будет вызываться вместо функции LIBRARYENTRY.

LIBRARYID –Задать идентификатор ресурсов

Пример...

Синтаксис:

extern "C" unsigned int far __export pascal LIBRARYID();

Примечание:

Функция возвращает идентификатор, по которому система может считать четыре разных типа ресурсов:

- меню с описанными операциями, которые может выполнить библиотека;
- размеры растрового слайда;
- имя библиотеки;
- имя иконки для минимизированного окна библиотеки.

LIBRARYBMPSIZE – Задать размер окна вывода растрового слайда

Синтаксис:

```
long far __export pascal LIBRARYBMPSIZE(void)          //для VISUAL C++
function LIBRARYHELPFILE: LongInt; Pascal;             //для PASCAL
```

Примечание:

Задает размер окна, в котором отображается слайд библиотечной функции. Данное назначение имеет более высокий приоритет по сравнению с назначением, сделанным в файле ресурсов.

LIBRARYHELPFILE – Определить имя файла справочной системы, подключаемого к библиотеке

Синтаксис:

extern "C" char far * pascal LIBRARYHELPFILE()

Описание:

Разделы Справочной системы могут иметь как библиотечные функции, так и команды меню, создаваемые внутри них и передаваемые в функции Placement и Cursor.

Примечание:

При использовании Unicode следует использовать функцию LIBRARYHELPFILEW.

LIBRARYHELPFILEW – Определить имя файла справочной системы, подключаемого к библиотеке (Unicode)

Синтаксис:

extern "C" LPWSTR WINAPI __export LIBRARYHELPFILEW()

Описание:

Разделы Справочной системы могут иметь как библиотечные функции, так и команды меню, создаваемые внутри них и передаваемые в функции Placement и Cursor.

Примечание:

При использовании ANSI следует использовать функцию LIBRARYHELPFILE.

LibToolBarId – Получить идентификаторы инструментальных и компактных панелей

Функция не поддерживается

Синтаксис:

extern "C" int WINAPI _export LibToolBarId (int barType, int index);

Входные параметры:

barType	- тип запрашиваемой панели (0 - компактная панель, 1 - простая инструментальная панель, 2 - контекстная всплывающая панель),

index - индекс панели.

Возвращаемое значение: (для компактной и простой инструментальной панели)

Идентификатор панели
-1 - если для заданного barType нет панели с заданным индексом.

Возвращаемое значение: (для всплывающих панелей)

Идентификатор панели
0 - запрет добавления панели,
-1 - отображать стандартную панель для макроэлемента,
>0 - идентификатор ресурсов для загрузки пользовательской панели.

Примечание:

1. Панель описывается тремя ресурсами:

- RCDATA состава панели;
- имя панели;
- значок инструментальной панели для возможности включения ее в компактную панель.

2. При описании панелей в ресурсах блок RCDATA необходимо заканчивать значением 0xffff.
3. Сначала запрашиваются обычные инструментальные панели, потом компактные панели.

   Система будет запрашивать функцию пока возвращаемое значение не будет равно -1. Если описанная в ресурсе инструментальная панель входит в компактную, то упоминание инструментальной панели в LibToolBarId не обязательно.
4. Отображение созданных панелей можно включить, вызвав команду **Вид – Панели инструментов** после подключения библиотеки.
5. Кнопки типа Fly-Out.
6. Пример описания ресурсов панелей и кнопки типа Fly-Out.
7. Индексы панелей начинаются с 0.
8. Размер, цвет, тип файла значков на кнопках…
9. Если в библиотеке нет компактной панели, то при barType = 0 нужно вернуть -1. Если в библиотеке нет инструментальной панели, то при barType = 1 нужно вернуть -1.
10. При селектировании библиотечного макроэлемента можно заменить или запретить отображение всплывающих панелей. Всплывающая панель может содержать две панели.

    Для получения списка контекстных панелей функция LibToolBarId вызывается первый раз при подключении библиотеки и требуется вернуть идентификаторы всех всплывающих панелей для всех библиотечных элементов. После подключения библиотеки функция вызывается при селектировании макроэлементов.

В зависимости от типа макроэлемента библиотека может вернуть свои или запретить всплывающие панели. Подробнее о создании контекстных панелей...

LibObjInterfaceEntry – Получить управление характерными точками библиотечного 2D элемента или реакцию на его редактирование

Синтаксис:

extern "C" void WINAPI LibObjInterfaceEntry(int idType, unsigned int comm, void** object);

Входные параметры:

idType	Тип интерфейса: 1 - интерфейс характерных точек ILibHPObject, 2 - интерфейс внешних воздействий IILibExternalObject, 3 - интерфейс внешних свойств объекта ILibPropertyObject

Выходные параметры:

object	- адрес указателя на запрошенный интерфейс.

Примечание:

Экспортная функция. Должна быть реализована на стороне библиотеки разработчиком приложения, если необходимо обеспечить управление характерными точками библиотечного элемента 2D или отреагировать на редактирование библиотечного элемента 2D.

Вызывается КОМПАС при селектировании библиотечного элемента для получения интерфейса характерных точек ILibHPObject или при редактировании библиотечного элемента для получения интерфейса внешних воздействий ILibExternalObject.

LibraryBmpBeginID – Получить для указанного размера иконок начало диапазона идентификаторов иконок команд библиотеки

Синтаксис:

extern "C" unsigned int WINAPI __export LibraryBmpBeginID (unsigned int bmpSizeType);

Входные параметры:

bmpSizeType	- тип размера иконок (типы определены в ksBmpSizeEnum).

Описание:

Функция предназначена для обеспечения отображения иконок команд библиотеки с размером, соответствующим настройкам интерфейса КОМПАС-3D.

Пример описания ресурсов:

...

MENUITEM "Команда 1", 1

```
MENUITEM "Команда 2", 2
MENUITEM "Команда 3", 3
...
// начало диапазона иконок размером 16x16 - 1000
1001 BITMAP "1.bmp"
1002 BITMAP "2.bmp"
1003 BITMAP "3.bmp"
// начало диапазона иконок размером 24x24 - 2000
2001 BITMAP "1_24.bmp"
2002 BITMAP "2_24.bmp"
2003 BITMAP "3_24.bmp"
...
// начало диапазона иконок размером 48x48 - 4000
4001 BITMAP "1_48.bmp"
4002 BITMAP "2_48.bmp"
4003 BITMAP "3_48.bmp"
```

Примечание:

1. Функция возвращает начало диапазона для иконок команд в зависимости от требуемого размера. Значения параметра bmpSizeType определены в ksBmpSizeEnum.
2. Если иконок нужного размера нет, функция должна вернуть 0, тогда иконки будут промасштабированы до запрашиваемого размера от ближайшего размера.
3. Наличие функции не обязательно. В случае ее отсутствия иконки будут автоматически масштабироваться от имеющегося размера до запрашиваемого.
4. Идентификатор иконки каждого размера должен быть задан как сумма значений начала диапазона (для каждого диапазона свое) и идентификатора пункта меню команды.

LibCommandState – Получить состояние команды библиотеки

Синтаксис:

extern "C" int WINAPI __export LibCommandState(unsigned int comm, int * enable, int * checked);

Входные параметры:

comm	- номер команды.

Выходные параметры:

enable	- доступность команды,
checked	- состояние команды (нажата/отжата).

Возвращаемое значение:

Не используется.

Примечание.

1. Функция возвращает состояние команды библиотеки.
2. Функция управляет состоянием команды в меню библиотеки и на библиотечных панелях команд.

LibraryHintTipsBeginID – Задать идентификатор начала диапазона ресурсов для подсказок к командам

Синтаксис:

extern "C" unsigned int WINAPI __export LibraryHintTipsBeginID();

Примечание:

1. Функция возвращает начало диапазона идентификаторов строк подсказок к командам библиотеки. Идентификатор строки подсказки команды должен быть задан как сумма значений начала диапазона и идентификатора пункта меню команды.
2. Подсказка задается в виде одной строки, состоящей из двух частей, разделенных символами «\n»: в первой части записывается текст, отображаемый в строке статуса, во второй — текст, отображаемый во всплывающей подсказке.
3. К тексту, отображаемому в строке статуса, автоматически добавляется (через «\») название библиотеки.
4. Наличие данной функции не обязательно.
5. Если функция не определена, или строка подсказки пустая, или идентификатор подсказки не существует, то подсказка будет формироваться по названию пункта меню команды.

Оформление прикладных библиотек типа ActiveX

Функции оформления библиотек типа ActiveX

Пример...

Чтобы создать библиотеку, следует использовать функции оформления библиотек. Они обеспечивают стыковку системы КОМПАС и приложения.

Пример простейшей программы в среде VB, оформленной в виде ActiveX библиотеки КОМПАС

```
Dim iKompasObject As Object 'KompasObject
Dim iDocument2D As Object 'ksDocument2D
Public Function GetLibraryName() As String
  GetLibraryName = “Самая простая библиотека”
End Function
```

```
Public Sub ExternalRunCommand(ByVal command As Integer, ByVal mode As Integer, ByVal
Kompas As Object)
  Set iKompasObject = Kompas
  If iKompasObject Is Nothing Then
    Exit Sub
  End If
  Set iDocument2D = iKompasObject.ActiveDocument2D
  If iDocument2D Is Nothing Then
    Exit Sub
  End If
  iKompasObject.ksMessage “Строим отрезок!”
  iDocument2D.ksLineSeg 10, 10, 100, 100, 1 ' x1, y1, x2, y2, тип линии - основная
End Sub
```

Функция ExternalRunCommand - аналог функции LibraryEntry. Ее наличие обязательно, так как именно ему передает управление КОМПАС при обращении к библиотеке. В данном случае она обеспечивает выдачу сообщения и построение одного единственного объекта чертежа - отрезка.

Функции

GetLibraryName – Получить имя библиотеки и определить, является ли она библиотекой КОМПАС

TAutoString GetLibraryName();

Синтаксис Visual Basic:

Public Function GetLibraryName() As String

GetHelpFile – Получить имя файла справочной системы, подключаемого к библиотеке

TAutoString GetHelpFile();

Синтаксис Visual Basic:

Public Function GetHelpFile() As String

GetImageHeight – Задать высоту окна вывода слайда

long GetImageHeight();

Синтаксис Visual Basic:

Public Function GetImageHeight() As Long

GetImageWidth – Задать ширину окна вывода слайда

long GetImageWidth();

Синтаксис Visual Basic:

Public Function GetImageWidth() As Long

ExternalRunCommand – Головная функция библиотеки

void ExternalRunCommand (short command,

short mode,

IDispatch* kompas);

Синтаксис Visual Basic:

Public Sub ExternalRunCommand (ByVal command As Integer,

ByVal mode As Integer,

ByVal kompas_ As Object)

Входные параметры:

command	- номер выполняемой команды;
mode	- режим работы:
	0 - normal - обычный режим,
	1 - demo - вызов из дистрибутивной задачи в деморежиме,
	2 - demoEx - вызов из демоверсии;
kompas	- интерфейс KompasObject или IApplication.

Примечание:

1. При вызове команды библиотеки из меню, панели инструментов или при редактировании библиотечного элемента по двойному нажатию кнопки мыши управление передается этому методу с номером выбранной команды.
2. Библиотека должна обязательно содержать один из предопределенных методов ExternalRunCommand, который определяет точку входа в приложение или LibInterfaceNotifyEntry. Наличие хотя бы одного из этих методов позволяет системе КОМПАС идентифицировать произвольный ActiveX DLL как собственное приложение.
3. Параметр kompas может быть указателем на интерфейс IApplication, если в библиотеке есть метод IsOnApplication7, и этот метод вернул значение TRUE, т.е. библиотека работает на API версии 7. Во всех остальных случаях параметр kompas будет указателем на интерфейс KompasObject.

LibInterfaceNotifyEntry – Головная функция библиотеки. Подписка на обработку событий от системы

Синтаксис:

BOOL LibInterfaceNotifyEntry (IDispatch *application);

Входные параметры:

application	- указатель интерфейса приложения KompasObject,

Примечание:

Библиотека должна обязательно содержать один из предопределенных методов ExternalRunCommand, который определяет точку входа в приложение, или LibInterfaceNotifyEntry. Наличие хотя бы одного из этих методов позволяет системе КОМПАС идентифицировать произвольный ActiveX DLL как собственное приложение.

ExternalGetMenu – Получить меню библиотеки

OLE_HANDLE ExternalGetMenu (BOOL *enableDelete);

Синтаксис Visual Basic:

Public Function ExternalGetMenu (ByVal enableDelete As Boolean) As OLE_HANDLE

Параметр:

enableDelete - признак удаления меню библиотеки (TRUE - удалять)

ExternalGetImage – Получить слайд для команды библиотеки с указанным номером

OLE_HANDLE ExternalGetImage (short command,

short * enableDelete);

Синтаксис Visual Basic:

Public Function ExternalGetImage (ByVal command As Integer,

enableDelete As Integer) As OLE_HANDLE

Параметры:

command-номер команды

enableDelete - признак удаления слайда (TRUE - удалять)

ExternalMenuItem – Получить строку меню для создания меню в виде строк

TAutoString ExternalMenuItem (short number,

short * itemType,

short * command);

Синтаксис Visual Basic:

Public Function ExternalMenuItem (ByVal number As Integer,

itemType As Integer,

command As Integer) As String

Параметры:

number - счетчик строк

itemType - тип строки

(0 - SEPARATOR (разделитель);

1 - MENUITEM (строка с названием пункта меню);

2 - POPUP (начало выпадающего подменю);

3 (или любое другое число) - ENDMENU (конец меню\подменю).

command - номер команды, если itemType = 1,

или -1 во всех остальных случаях

ExternalGetToolBarId – Задать идентификаторы панелей управления библиотеки

long ExternalGetToolBarId(short barType, short index);

Входные параметры:

barType	- тип запрашиваемой панелей (0 - компактная панель, 1 - простая инструментальная панель),
index	- индекс панели.

Примечание:

1. Панель описывается тремя ресурсами:

- RCDATA состава панели;
- имя панели;
- значок инструментальной панели для возможности включения ее в компактную панель.

2. При описании панелей в ресурсах блок RCDATA необходимо заканчивать значением 0xffff.
3. Сначала запрашиваются обычные инструментальные панели, потом компактные панели.
4. Система будет запрашивать функцию пока возвращаемое значение не будет равно -1. Если описанная в ресурсе инструментальная панель входит в компактную, то упоминание инструментальной панели в LibToolBarId не обязательно.
5. Отображение созданных панелей можно включить, вызвав команду **Вид — Панели инструментов** после подключения библиотеки.
6. Кнопки типа Fly-Out.
7. Пример описания ресурсов панелей и кнопки типа Fly-Out.

DisplayLibraryName – Получить имя библиотеки, отображаемое при работе КОМПАС

Синтаксис:

BSTR DisplayLibraryName();

Возвращаемое значение:

- Имя библиотеки, отображаемое на дисплее во время работы КОМПАС.

Примечание:

Необязательная функция. Позволяет получить имя библиотеки, которое будет отображаться на дисплее во время работы системы КОМПАС (в меню, в менеджере библиотек,

на панелях). Если эта функция не используется, то в качестве имени библиотеки будет отображаться возвращаемое значение функции GetLibraryName. Имя библиотеки, полученное при помощи функции GetLibraryName, передается в создаваемые библиотекой макроэлементы. При редактировании таких макроэлементов по двойному щелчку мыши или через интерфейс хот-точек система находит библиотеку для редактирования по имени файла, имени библиотеки, по номеру команды. Если выходит новая версия библиотеки, в ней можно переопределить имя, отображаемое функцией DisplayLibraryName, оставив прежним имя, возвращаемое GetLibraryName. В этом случае новая библиотека может быть использована, чтобы редактировать макроэлементы, созданные старыми версиями библиотеки.

IsOnApplication7 – Получить тип версии API используемого библиотекой

VARIANT_BOOL IsOnApplication7();

Возвращаемое значение:

TRUE	- библиотека использует API версии 7,
FALSE	- библиотека использует API версии 5.

Примечание:

Необязательный метод оформления библиотеки. Позволяет задать тип интерфейса, который будет передан библиотеке в параметре kompas метода ExternalRunCommand и в параметре application метода LibInterfaceNotifyEntry. Если метод в библиотеке не объявлен, то считается что библиотека использует API версии 5.

ExternalGetResourceModule – Задать модуль с ресурсами библиотеки

VARIANT ExternalGetResourceModule();

Возвращаемое значение:

HINSTANCE (тип VT_I4), либо имя файла (тип VT_BSTR) dll, в котором находятся ресурсы библиотеки.TRUE

Примечание:

Добавлена возможность указания произвольного dll-файла с ресурсами.

Оформление прикладных библиотек типа Addin

Общие сведения о библиотеках типа AddIn

Это обычная библиотека, но она регистрируется в реестре и загружается при запуске системы КОМПАС.

Без регистрации в реестре библиотеки такого типа работают как стандартные прикладные библиотеки КОМПАС.

Регистрация осуществляется в разделе HKEY_CURRENT_USER\\Software\\ASCON\\KOMPAS-3D\\AddIns или HKEY_LOCAL_MACHINE\\Software\\ASCON\\KOMPAS-3D\\AddIns. Для регистрации создается подраздел с уникальным именем (например, именем самой библиотеки). В этом разделе должны быть два значения:

- Тип значения **REG_SZ**:

  "ProgID" - для ActiveX библиотек или "Path" - полный путь к файлу для простых библиотек. Если есть оба значения "ProgID" и "Path", то "Path" игнорируется. ActiveX библиотека должна быть зарегистрирована.

- Тип значения **REG_DWORD**:

  "AutoConnect" - 1 - подключать библиотеку при запуске КОМПАС, 0 - не подключать. Если в реестре этого значения нет, то оно по умолчанию принимается равным 1.

Существует два способа регистрации библиотеки в реестре:

- при помощью функций DllRegisterServer.
- вызов команды regsvr32.exe libname.

Чтобы отменить регистрацию, необходимо использовать функцию DllUnregisterServer или вызвать команду regsvr32.exe \u libname.

AddIn библиотеки работают в режиме "Меню". Сменить режим нельзя, так же как и отключить библиотеку.

Пример создания библиотеки типа AddIn

1. #define ADDINS_PATH "HKEY_CURRENT_USER\Software\ASCON\KOMPAS-3D\AddIns\Моя библиотека"

```
STDAPI DllRegisterServer()
{
  HRESULT hr = NOERROR;
  TCHAR   szModulePath[MAX_PATH];

  GetModuleFileName( theApp.m_hInstance, szModulePath,
                sizeof(szModulePath)/sizeof(TCHAR) );

  CString strPath( ADDINS_PATH );
  HKEY hKey;
  DWORD dwDisposition;
  if ( RegCreateKeyEx(HKEY_CURRENT_USER,
      strPath, 0L, NULL, REG_OPTION_NON_VOLATILE, KEY_ALL_ACCESS, NULL,
      &hKey, &dwDisposition) != ERROR_SUCCESS )
  {
    hr = E_UNEXPECTED;
```

```
  }
  else
  {
  hr = RegSetValueEx( hKey, "Path", 0L, REG_SZ,
                (CONST BYTE*)szModulePath, strlen(szModulePath) );
    DWORD dwVal = 1;
  hr = RegSetValueEx( hKey, "AutoConnect", 0L, REG_DWORD,
                (CONST BYTE*)&dwVal, sizeof(DWORD) );
  }
  return hr;
}

STDAPI DllUnregisterServer()
{
  HRESULT hr = NOERROR;
  if ( RegDeleteKey(HKEY_CURRENT_USER, ADDINS_PATH) != ERROR_SUCCESS ) {
    hr = E_UNEXPECTED;
  }
  return hr;
}
```

2. Файлом типа reg

   Пример:

```
Файл реестра "MyLibrary.reg"
REGEDIT4
[HKEY_CURRENT_USER\Software\ASCON\KOMPAS-3D\AddIns\Моя библиотека]
"ProgID"="MyLibrary.class1"
"Path"="c:\\MyLibrary.rtw"
"AutoConnect"=dword:00000001

Содержимое реестра после регистрации файла "MyLibrary.reg"
HKEY_CURRENT_USER
  Software
    ASCON
      KOMPAS-3D
        AddIns
          Моя библиотека
            ProgID       MyLibrary.class1
```

```
Path        c:\\MyLibrary.rtw
AutoConnect  1
```

Оформление библиотек типа Converter

Общие сведения о библиотеках типа Converter

Библиотеки типа Converter позволяют выполнять импорт и экспорт документов различных форматов.

В библиотеке должен быть реализован интерфейс конвертора IKompasConverter.

В библиотеке должны быть реализованы следующие функции.

- LIBRARYID - идентификатор для ресурсов библиотеки, который позволяет получить имя библиотеки и меню команд.
- LPKOMPASCONVERTER WINAPI GetIKompasConverter() - предопределенная функция, возвращающая интерфейс конвертора.
- void WINAPI LIBRARYENTRY (unsigned int comm) - точка входа для вызова команд.

Чтобы создать библиотеку, следует использовать функции оформления библиотек. Они обеспечивают стыковку системы КОМПАС и приложения.

После регистрации конвертора в реестре указанные расширения файлов появляются в диалогах сохранения (после вызова команды **Файл – Сохранить как...**) и открытия файлов. В диалоге сохранения файлов расширения файлов появятся в том случае, если для текущего типа файла задан номер команды для сохранения SaveCommandID. В диалоге открытия файлов фильтр будет содержать все расширения, для которых задан номер команды для открытия OpenCommandID.

Примечание. Номера команд для разных типов файлов могут совпадать.

IKompasConverter::Convert - для сохранения или открытия файла из диалога, как функция интерфейса.

При сохранении документа в качестве имени исходного файла передается пустая строка, а в качестве имени выходного файла передается имя, заданное пользователем в диалоге. Значение номера команды выбирается из реестра.

При открытии файла в качестве имени исходного файла передается имя, заданное пользователем в диалоге, а в качестве выходного файла передается пустая строка.

При вызове конвертора из прикладной библиотеки может приходить и входное, и выходное имя файла. Это означает, что нужно открыть нужный файл (можно в невидимом режиме), если он еще не открыт, и конвертировать его. После конвертации, если файл был открыт конвертором, он должен быть закрыт.

Интерфейс IKompasConverter

Стандартный интерфейс импорта/экспорта.

Позволяет связать систему КОМПАС и библиотеки импорт/экспорта документов.

Реализуется в библиотеках типа Конвертор

IKompasConverter – свойства

CanUnloadLibrary – Разрешение на выгрузку библиотеки из памяти

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- библиотеку можно выгружать из памяти,
FALSE	- библиотеку нельзя выгружать из памяти.

Синтаксис Automation:

CanUnloadLibrary = iObject.CanUnloadLibrary	Получить свойство (*)
CanUnloadLibrary = iObject.GetCanUnloadLibrary( )	Получить свойство (**)

Синтаксис COM:

iObject->get_CanUnloadLibrary (&CanUnloadLibrary)	Получить свойство.

Примечание:

Если внутренний интерфейс параметров захвачен другими пользователями, то библиотеку нельзя выгружать из памяти до тех пор, пока он не освободится. Так как в системе КОМПАС предусмотрена автоматическая выгрузка библиотек, то перед выгрузкой будет вызвана эта функция и библиотека будет выгружена, только если она вернет TRUE.

ClonverterParameters – Получить параметры конвертора

Интерфейс...

Синтаксис Automation:

LPDISPATCH ConverterParameters(long command);

Синтаксис COM:

HRESULT ConverterParameters ([in] long command, [out, retval] IUnknown** iParam);

Входные параметры:

command	- номер команды.

Возвращаемое значение:

указатель на интерфейс IUnknown	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Данный метод позволяет получить интерфейс параметров конвертирования.

GetFilter –Получить фильтр и номер команды по типу документа

Интерфейс...

Синтаксис Automation:

BSTR * GetFilter (long docType, BOOL saveAs, long * command);

Синтаксис COM:

HRESULT GetFilter ([in] long docType,
[in] VARIANT_BOOL saveAs,
[out] long * command,
[out, retval] BSTR* Result);

Входные параметры:

docType	при импорте в КОМПАС-документ - тип документа из перечисления DocumentTypeEnum, при экспорте КОМПАС-документа - тип документа, в который производится экспорт; перечисление типов документов должно быть задано в конверторе. Например, для конвертора в DXF и DWG используются типы: #define FORMAT_DXF 1 // ID команды для работы с DXF #define FORMAT_DWG 2 // ID команды для работы с DWG
saveAs	- TRUE - экспорт документа КОМПАС, - FALSE - импорт в документ КОМПАС.

Выходные параметры:

command	- номер команды.

Возвращаемое значение:

фильтр	- строка с справочным значением (например, "КОМПАС-фрагменты (*.myfrw l*.myfrwl"), которое используется в диалогах открытия и сохранения файлов.

Convert – Запустить процесс конвертации

Интерфейс...

Синтаксис Automation:

long * Convert (BSTR inputFile, BSTR outfile, long command, BOOL showParam);

Синтаксис COM:

HRESULT Convert ([in] BSTR inputFile,
[in] BSTR outfile,
[in] long command,

[in] VARIANT_BOOL showParam,
[out, retval] long* Result);

Входные параметры:

inputFile	- имя исходного файла документа,
outfile	- новое имя файла документа,
command	- номер команды,
showParam	- TRUE - выдавать диалог параметров конвертации,
	- FALSE - не выдавать диалог параметров конвертации.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если имя исходного файла документа не задано, то берётся текущий документ, если новое имя файла документа не задано, то конвертация будет происходить в новый документ системы КОМПАС.

VIisualEditConvertParam – Запустить визуальное редактирование параметров конвертации

Интерфейс...

Синтаксис Automation:

BOOL * VisualEditConvertParam (OLE_HANDLE parentHwnd, long command);

Синтаксис COM:

HRESULT VisualEditConvertParam ([in] OLE_HANDLE parentHwnd,
[in] long command,
[out, retval] VARIANT_BOOL * val);

Входные параметры:

parentHwnd	- дескриптор окна родителя,
command	- номер команды.

Возвращаемое значение:

TRUE	- выход из диалога по кнопке **OK**,
FALSE	- выход из диалога по кнопке **Отмена**.

Приложение

Интерфейс IKompasAPIObject

Базовый объект КОМПАС API (скрытый).

Примечание:

Данный интерфейс является базовым для всех интерфейсов КОМПАС API, кроме интерфейсов событий и некоторых вспомогательных интерфейсов.

IKompasAPIObject – свойства

Application – Ссылка на приложение

Интерфейс...

Тип данных: указатель на интерфейс IApplication.

Синтаксис Automation:

```
Application = iObject.Application          Получить свойство(* )
Application = iObject.GetApplication()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Application (&Application)    Получить свойство.
```

Примечание:

Свойство доступно только для чтения. Возвращает указатель на интерфейс приложения КОМПАС.

Parent – Ссылка на владельца

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject.

Синтаксис Automation:

```
Parent = iObject.Parent                    Получить свойство(* )
Parent = iObject.GetParent()               Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Parent (&Parent)              Получить свойство
```

Примечание:

Свойство доступно только для чтения. Возвращает указатель на интерфейс, у которого был взят данный объект.

Reference – Уникальный идентификатор объекта

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Reference = iObject.Reference                Получить свойство(* )
Reference = iObject.GetReference()           Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Reference (&Reference)          Получить свойство.
```

Примечание:

Свойство доступно только для чтения. Является скрытым; недоступно в таких языках как VB и т.п.

Type – Тип объекта

Интерфейс...

Тип данных: тип объекта из перечисления KompasAPIObjectTypeEnum.

Синтаксис Automation:

```
Type = iObject.Type                          Получить свойство(* )
Type = iObject.GetType()                     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Type (&Type)                    Получить свойство
```

Примечание:

Свойство доступно только для чтения. Позволяет определить тип объекта и его интерфейс.

Интерфейс IKompasCollection

Базовая коллекция КОМПАС API (скрытый).

Иерархия:

IKompasAPIObject

    IKompasCollection

Примечание:

Данный интерфейс является базовым для всех интерфейсов коллекций КОМПАС API.

Индекс первого элемента коллекции 0.

IKompasCollection – свойства

_NewEnum – Возвращает коллекцию (в VB циклы: For Each)

Интерфейс...

Тип данных: указатель на интерфейс IUnknown объекта-перечислителя.

Примечание:

Данное свойство позволяет реализовывать циклический проход по коллекции с помощью специальных операторов цикла, таких как оператор For Each в Visual Basic.

Count – Количество элементов в коллекции

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Count = iObject.Count                Получить свойство(* )
Count = iObject.GetCount()           Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Count (&Count)          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Интерфейс IApplication

Интерфейс приложения КОМПАС-3D.

Иерархия:

IKompasAPIObject

IApplication

IAttrTypeMng

IPropertyMng

Примечание:

1. Приложение КОМПАС-3D регистрируется с ProgID «KOMPAS.Application.7».
2. После запуска приложения из-под внешнего контроллера оно работает в слепом режиме. Чтобы оно стало видимым, нужно установить свойству IApplication::Visible значение TRUE.
3. Данный интерфейс приложения можно получить у любого объекта с помощью свойства IKompasAPIObject::Application.

4. Интерфейс приложения может быть создан следующими способами:

- в C с помощью функций HINSTANCE LoadLibrary(LibFileName), GetProcAddress, CreateKompasApplication(), пример создания интерфейса реализован в функции GetNewKompasAPI() - SDK\VisualC: Gayka1, Step12, Cube;
- в Delphi с помощью функции CreateKompasApplication, пример создания интерфейса реализован в функции GetNewKompasAPI() - SDK\Pascal\Delphi: Gayka1, Step12, Cube;
- в VBasic с помощью функции ksGetApplication7, пример создания интерфейса реализован в SDK\Basic: Gayka1, events.

5. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы IAttrTypeMng, IPropertyMng и IApplicationLicenseManager.

IApplication – свойства

ActiveDocument – Получить текущий активный документ

Интерфейс...

Тип данных: указатель на интерфейс IKompasDocument.

Синтаксис Automation:

```
ActiveDocument = iObject.ActiveDocument                 Получить свойство (* )
iObject.ActiveDocument = ActiveDocument                 Установить свойство (* )
ActiveDocument = iObject.GetActiveDocument()            Получить свойство(**)
iObject.SetActiveDocument (ActiveDocument)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ActiveDocument (&ActiveDocument)           Получить свойство
iObject->put_ActiveDocument (ActiveDocument)            Установить свойство
```

Примечание:

Свойство позволяет получить или установить текущий документ, если ни один документ не открыт.

ApplicationName – Имя приложения

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

```
ApplicationName = Object.ApplicationName( FullName )    Получить свойство(* )
ApplicationName = Object.GetApplicationName( FullName ) Получить свойство(**)
```

Синтаксис COM:

Object.get_ApplicationName(FullName, &ApplicationName) Получить свойство,

Входные параметры:

BOOL - FullName- полное имя,
TRUE - полное отображаемое имя приложения в заголовке окна Компас,
FALSE - нелокализованное английское имя.

CheckSum – Интерфейс контрольной суммы

Интерфейс...

Тип данных: Указатель на интерфейс ICheckSum.

Синтаксис Automation:

ICheckSum * CheckSum = Application.CheckSum Получить свойство(*)
ICheckSum * CheckSum = Application.GetCheckSum() Получить свойство(**)

Синтаксис COM:

HRESULT get_CheckSum (ICheckSum ** pRes) Получить свойство,

Примечание:

Свойство доступно только для чтения.

Converter – Конвертация документов КОМПАС

Интерфейс...

Тип данных: указатель на интерфейс IConverter конвертера файлов КОМПАС.

Синтаксис Automation:

Converter = iObject.Converter (Library) Получить свойство(*)
Converter = iObject.GetConverter (Library) Получить свойство(**)

Синтаксис COM:

iObject->get_Converter (Library, &Converter) Получить свойство,

Входные параметры:

Library - полный путь к библиотеке, тип VARIANT.

Примечание:

Свойство доступно только для чтения.

CurrentDirectory – Текущий каталог

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

CurrentDirectory = Object.CurrentDirectory	Получить свойство (*)
Object.CurrentDirectory = CurrentDirectory	Установить свойство (*)
CurrentDirectory = Object.GetCurrentDirectory()	Получить свойство(**)
Object.SetCurrentDirectory(CurrentDirectory)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentDirectory(&CurrentDirectory)	Получить свойство,
Object.put_CurrentDirectory(CurrentDirectory)	Установить свойство.

Примечание

При работе функций КОМПАС может изменяться текущий рабочий каталог приложения.

Свойство позволяет получить/установить текущий рабочий каталог.

Documents – Коллекция открытых документов в приложении

Интерфейс...

Тип данных: указатель на интерфейс коллекции документов IDocuments.

Синтаксис Automation:

Documents = iObject.Documents	Получить свойство(*)
Documents = iObject.GetDocuments()	Получить свойство(**)

Синтаксис COM:

iObject->get_Documents (&Documents)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

HideMessage – Скрывать/показывать сообщения

Интерфейс...

Тип данных: константа из перечисления ksHideMessageEnum.

Синтаксис Automation:

ksHideMessageEnum HideMessage = Application.HideMessage	Получить свойство (*)
Application.HideMessage = HideMessage	Установить свойство (*)

ksHideMessageEnum HideMessage = Application.GetHideMessage()	Получить свойство(**)
Application.SetHideMessage (HideMessage)	Установить свойство (**)

Синтаксис COM:

HRESULT get_HideMessage (ksHideMessageEnum * pRes)	Получить свойство,
Application->put_HideMessage (HideMessage)	Установить свойство.

Примечание:

1. После использования флагов ksHideMessageYes или ksHideMessageNo в прикладной библиотеке необходимо предусматривать взведение флага ksShowMessage по завершении работы библиотеки или команды библиотеки. В противном случае сообщения останутся подавленными в визуальном режиме работы с системой КОМПАС.
2. Константа ksHideMessageYes позволяет выполнять перестроения документа, скрывая все запросы на перестроение.
3. Константа ksHideMessageNo позволяет не выполнять перестроения документа, скрывая все запросы на перестроение.
4. Для всех сообщений с одной кнопкой **ОК** константы ksHideMessageYes и ksHideMessageNo одинаковым образом скрывают сообщения с обработкой нажатия **ОК**.
5. Для всех запросов с выбором **ДА** и **НЕТ** константа ksHideMessageYes позволяет скрыть запрос с обработкой выбора варианта **ДА**, константа ksHideMessageNo позволяет скрыть запрос с обработкой выбора варианта **НЕТ**.
6. При открытии спецификации, когда отсутствует файл graphic.lyt, константы ksHideMessageYes и ksHideMessageNo позволяют скрыть запрос на выбор нового стиля с отказом от выбора нового стиля.
7. При открытии спецификации, когда имеется рассогласование с объектами, константа ksHideMessageYes позволяет скрыть запрос **Обнаружено рассогласование между спецификацией и сборкой** с выбором варианта **Взять объект из сборки** и скрыть запрос **В сборке отсутствует объект** с выбором варианта **Восстановить в сборке**. Константа ksHideMessageNo позволяет скрыть запрос **Обнаружено рассогласование между спецификацией и сборкой** с выбором варианта **Взять объект из спецификации**, и скрыть запрос **В сборке отсутствует объект** с выбором варианта **Удалить из спецификации**.
8. При открытии спецификации, когда отсутствует файл чертежа, связанного со спецификацией, константа ksHideMessageYes позволяет скрыть сообщение с удалением чертежа из спецификации, константа ksHideMessageNo позволяет скрыть сообщение без удаления чертежа из спецификации.
9. При открытии чертежа со вставкой внешних фрагментов, когда фрагменты отсутствуют, константа ksHideMessageYes позволяет скрыть запрос с обработкой выбора варианта удаления из чертежа отсутствующих фрагментов, константа ksHideMessageNo позволяет скрыть запрос с обработкой выбора варианта игнорировать отсутствующие фрагменты.
10. При открытии ассоциативного чертежа, когда файл модели, связанной с чертежом, отсутствует, константа ksHideMessageYes позволяет скрыть запрос с обработкой выбора варианта удаления из чертежа видов, связанных с отсутствующими моделями, константа

ksHideMessageNo позволяет скрыть запрос с обработкой выбора варианта игнорировать отсутствующие модели.

KompasError – Информация об ошибке системы КОМПАС

Интерфейс...

Тип данных: указатель на интерфейс IKompasError информации о ошибке системы КОМПАС.

Синтаксис Automation:

```
KompasError = iObject.KompasError                Получить свойство(* )
KompasError = iObject.GetKompasError()           Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_KompasError (&KompasError)          Получить свойство,
```

Примечание:

Свойство доступно только для чтения.

LibraryManager – Менеджер библиотек

Интерфейс...

Тип данных: указатель на интерфейс ILibraryManager менеджера библиотек.

Синтаксис Automation:

```
LibraryManager = iObject.LibraryManager          Получить свойство(* )
LibraryManager = iObject.GetLibraryManager()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_LibraryManager (&LibraryManager)    Получить свойство,
```

Примечание:

Свойство доступно только для чтения.

LibraryStyles – Стили из библиотеки

Интерфейс...

Тип данных: Указатель на интерфейс IStyles

Синтаксис Automation:

```
LibraryStyles = Object.LibraryStyles( Path, StylesType )      Получить свойство(* )
LibraryStyles = Object.GetLibraryStyles( Path, StylesType )   Получить свойство(**)
```

Синтаксис COM:

Object.get_LibraryStyles(Path, StylesType, &LibraryStyles) Получить свойство,

Примечание:

Свойство доступно только для чтения.

Math2D – Интерфейс 2D математики

Интерфейс...

Тип данных:указатель на интерфейс IMath2D.

Синтаксис Automation:

```
Math2D = Object.Math2D                Получить свойство(* )
Math2D = Object.GetMath2D()           Получить свойство(**)
```

Синтаксис COM:

```
Object.get_Math2D( &Math2D )          Получить свойство,
```

Примечание:

Свойство доступно только для чтения.

PrintJob - Интерфейс задания на печать

Интерфейс...

Тип данных:указатель на интерфейс IPrintJob

Синтаксис Automation:

```
PrintJob = Object.PrintJob            Получить свойство(* )
PrintJob = Object.GetPrintJob()       Получить свойство(**)
```

Синтаксис COM:

```
Object.get_PrintJob( &PrintJob )      Получить свойство,
```

Свойство позволяет получать интерфейс задания на печать.

Примечание:

Свойство доступно только для чтения.

ProgressBarIndicator – Интерфейс индикатора прогресса

Интерфейс...

Тип данных: указатель на интерфейс IProgressBarIndicator.

Синтаксис Automation:

IProgressBarIndicator * ProgressBarIndicator = Application.ProgressBarIndicator — Получить свойство(*)
IProgressBarIndicator * ProgressBarIndicator = Application.GetProgressBarIndicator() — Получить свойство(**)

Синтаксис COM:

HRESULT get_ProgressBarIndicator (IProgressBarIndicator ** pRes) — Получить свойство

SystemSettings – Интерфейс параметров системы

Интерфейс...

Тип данных: указатель на интерфейс ISystemSettings.

Синтаксис Automation:

ISystemSettings * SystemSettings = Application.SystemSettings — Получить свойство(*)
ISystemSettings * SystemSettings = Application.GetSystemSettings() — Получить свойство(**)

Синтаксис COM:

HRESULT get_SystemSettings (ISystemSettings ** pRes) — Получить свойство

Примечание:

Свойство доступно только для чтения.

Visible – Видимость приложения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible — Получить свойство (*)
iObject.Visible = Visible — Установить свойство (*)
Visible = iObject.GetVisible() — Получить свойство(**)
iObject.SetVisible (Visible) — Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible) — Получить свойство,
iObject->put_Visible (Visible) — Установить свойство.

Примечание:

Позволяет получить и установить свойство видимости приложения КОМПАС-3D.

IApplication – методы

CreateProcessParam – Получить интерфейс параметров процесса

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateProcessParam();

Синтаксис COM:

HRESULT CreateProcessParam([out, retval] IProcessParam** processParam);

Возвращаемое значение:

указатель на интерфейс IProcessParam - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Данный метод создает новые параметры процесса и возвращает их. Параметры будут удалены после их освобождения библиотекой, поэтому библиотека должна держать ссылку на параметры все время от начала до завершения их использования. При повторном вызове этого метода будут созданы новые параметры.

CreatePropertyManager – Создать Панель свойств

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreatePropertyManager (BOOL newManager);

Синтаксис COM:

HRESULT CreatePropertyManager([in, defaultvalue(FALSE)] VARIANT_BOOL newManager, [out, retval] IPropertyManager** propertyManager);

Входные параметры:

newManager - тип создаваемой Панели свойств:
TRUE - библиотечная (создаваемая библиотекой) панель свойств,
FALSE - панель свойств процесса системы КОМПАС-3D.

Возвращаемое значение:

указатель на интерфейс IPropertyManager Панели свойств.

Примечание:

1. При получении интерфейса доступа к панели свойств процесса системы КОМПАС-3D будет возможно добавление собственных вкладок с элементами управления. Все остальные вкладки и их элементы управления будут недоступны.
2. Библиотечная (создаваемая библиотекой) панель свойств и/или создаваемые библиотекой вкладки на панели свойств процесса системы КОМПАС-3D будут удалены после освобождения библиотекой их интерфейса, поэтому библиотека должна держать ссылку на интерфейс панели свойств все время от начала до завершения ее использования.
3. При повторном вызове этого метода будет будет создана новая библиотечная (создаваемая библиотекой) панель свойств и/или создаваемые библиотекой вкладки на панели свойств процесса системы КОМПАС-3D.

ExecuteKompasCommand – Выполнить команду системы КОМПАС

Интерфейс...

Синтаксис Automation:

BOOL ExecuteKompasCommand (long commandID, BOOL post);

Синтаксис COM:

HRESULT ExecuteKompasCommand (long commandID, BOOL post, BOOL * retval);

Входные параметры:

commandID	- константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.
post	- true - запуск команды через PostMessage, - false - через SendMessage.

Возвращаемое значение:

TRUE	- в случае удачи,
FALSE	- в случае ошибки.

Примечание:

1. Проверить доступность команды можно с помощью функции IApplication::IsKompasCommandEnable.
2. Проверить, нажата ли в данный момент кнопка команды, можно с помощью IApplication::IsKompasCommandCheck.

IsKompasCommandEnable – Проверить доступность выполнения команды

Интерфейс...

Синтаксис Automation:

BOOL IsKompasCommandEnable (long commandID);

Синтаксис COM:

HRESULT IsKompasCommandEnable (long commandID, BOOL * retVal);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

TRUE - команда доступна,
FALSE - команда недоступна.

IsKompasCommandCheck – Проверить, нажата ли кнопка команды

Интерфейс...

Синтаксис Automation:

long IsKompasCommandCheck (long commandID);

Синтаксис COM:

HRESULT IsKompasCommandCheck (long commandID, long * retVal);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

1 - кнопка нажата,
0 - кнопка отжата.

MessageBoxEx – Выдать всплывающее сообщение

Интерфейс...

Синтаксис Automation:

long MessageBoxEx(BSTR Text, BSTR Caption, long Flags)

Синтаксис COM:

HRESULT MessageBoxEx(BSTR Text, BSTR Caption, long Flags, long * Result);

Входные параметры:

Text - текст сообщения,
Caption - заголовок сообщения,
Flags - флаги.

Примечание:

Параметр Flags и возвращаемое значение соответствует аналогичной функции WinApi MessageBox.

Дополнительно функция позволяет выдать всплывающее сообщение если в настройках Компас установлен признак **Сервис->Параметры->Система->Общее->Всплывающие сообщения->Использовать всплывающие сообщения вместо информационных диалогов**.

Сообщение выдается всплывающим, если используется комбинация флагов MB_OK | MB_ICONINFORMATION или MB_OK | MB_ICONWARNING.

Если установлены какие-либо другие флаги, то сообщение выдается как обычное сообщение типа MessageBox.

Если не задан заголовок сообщения, выдается умолчательный заголовок - «Сообщение библиотеки».

Если не задан текст сообщения, выдается сообщение о текущей ошибке аналогично функции MessageBoxResult.

MessageDlg – Выдать модельное сообщение

Интерфейс...

Синтаксис Automation:

long MessageDlg(OLE_HANDLE Parent, BSTR Text, BSTR Caption, BSTR Explanation, long Flags, BSTR PositiveButton, BSTR NegativeButton, BSTR CancelButton, long HelpId, BSTR HelpFileName);

Синтаксис COM:

HRESULT MessageDlg(OLE_HANDLE Parent, BSTR Text, BSTR Caption, BSTR Explanation, long Flags, BSTR PositiveButton, BSTR NegativeButton, BSTR CancelButton, long HelpId, BSTR HelpFileName, long * Result);

Возвращаемое значение:

- идентификатор нажатой в диалоге кнопки.

Входные параметры:

Parent	- дескриптор родительского окна,
Text	- текст сообщения,
Caption	- заголовок,
Explanation	- дополнительный текст сообщения,
Flags	- флаги,
PositiveButton	- текст кнопки для позитивного результата,
NegativeButton	- текст кнопки для негативного результата,
CancelButton	- текст кнопки для отмены,
HelpId	- идентификатор справки,
HelpFileName	- имя файла справки.

Примечание:

1. Параметр Flags и возвращаемое значение соответствует аналогичной функции WinApi MessageBox.
2. Тексты кнопок не обязательные параметры.
3. Если тексты не заданы, используются названия кнопок по умолчанию.
4. Наличие кнопок в диалоге зависит от параметра Flags.

Quit – Закрыть приложение

Интерфейс...

Синтаксис Automation:

void Quit();

Синтаксис COM:

HRESULT Quit();

Примечание:

Используется при работе из-под внешнего контроллера.

StopCurrentProcess – Остановить текущий процесс в документе

Интерфейс...

Синтаксис Automation:

void StopCurrentProcess (BOOL postMessage, LPDISPATCH pDoc);

Синтаксис COM:

HRESULT StopCurrentProcess([in, defaultvalue(FALSE)]VARIANT_BOOL postMessage, [in, defaultvalue(0)] IKompasDocument* pDoc);

Входные параметры:

postMessage	- способ остановки процесса,
pDoc	- указатель на интерфейс IKompasDocument, в котором требуется остановить процесс.

Примечание:

1. Если процесс требуется остановить немедленно то параметр postMessage должен иметь значение FALSE, однако такая остановка не всегда возможна, например, остановка из обработчика события. В этом случае результат обработки события должен быть обработан процессом; в этом случае процессу надо сообщить чтобы он остановился после обработки события. Для этого параметру postMessage нужно установить значение TRUE.
2. Если параметру pDoc установлено значение 0, то процесс будет остановлен в текущем документе.

IApplication – события

См. Интерфейс ksKompasObjectNotify

Менеджер лицензий

Интерфейс IApplicationLicenseManager

Менеджер лицензий

Иерархия:

IDispatch

IApplicationLicenseManager

Примечание:

Интерфейс является дополнительным к интерфейсу приложения IApplication. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

Версия: Компас v19

IApplicationLicenseManager – свойства

KompasModuleActive – Доступность компонентов КОМПАС

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

KompasModuleActive = Object.KompasModuleActive(Module)	Получить свойство (*)
Object.KompasModuleActive(Module) = KompasModuleActive	Установить свойство (*)
KompasModuleActive = Object.GetKompasModuleActive(Module)	Получить свойство (**)
Object.SetKompasModuleActive(Module, KompasModuleActive)	Установить свойство (*)

Синтаксис COM:

Object.get_KompasModuleActive(Module, &KompasModuleActive)	Получить свойство
Object.put_KompasModuleActive(Module, KompasModuleActive)	Установить свойство

Входные параметры:

ksKompasModuleEnum - Module - Модули Компас.

KompasVariant – Получение типа установленного дистрибутива в виде комбинации флагов из ksKompasVariantEnum

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
KompasVariant = Object.KompasVariant                    Получить свойство (* )
KompasVariant = Object.GetKompasVariant()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_KompasVariant( &KompasVariant )              Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

LibraryActive – Доступность продукта по номеру

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
LibraryActive = Object.LibraryActive( ProductNumber )          Получить свойство (* )
Object.LibraryActive( ProductNumber ) = LibraryActive          Установить свойство (* )
LibraryActive = Object.GetLibraryActive( ProductNumber )       Получить свойство (**)
Object.SetLibraryActive( ProductNumber, LibraryActive )        Установить свойство (* )
```

Синтаксис COM:

```
Object.get_LibraryActive( ProductNumber, &LibraryActive )   Получить свойство
Object.put_LibraryActive(  ProductNumber, LibraryActive )   Установить свойство
```

Входные параметры:

long ProductNumber - номер продукта.

LibraryLocalStatus – Признак локальный/ сетевой продукт

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LibraryLocalStatus = Object.LibraryLocalStatus(ProductNumber) Получить свойство (*)
LibraryLocalStatus = Object.GetLibraryLocalStatus(ProductNumber) Получить свойство (**)

Синтаксис COM:

Object.get_LibraryLocalStatus(ProductNumber, &LibraryLocalStatus) Получить свойство (*)

Входные параметры:

long ProductNumber - номер продукта.

Примечание:

Свойство доступно только для чтения.

LibraryStatus – Получить текущий статус продукта по номеру

Интерфейс...

Тип данных: из перечисления ksProtectProductStatusEnum.

Синтаксис Automation:

LibraryStatus = Object.LibraryStatus(ProductNumber) Получить свойство (*)
LibraryStatus = Object.GetLibraryStatus(ProductNumber) Получить свойство (**)

Синтаксис COM:

Object.get_LibraryStatus(ProductNumber, &LibraryStatus) Получить свойство (*)

Входные параметры:

long ProductNumber - номер продукта.

Примечание:

Свойство доступно только для чтения.

LibraryTrialStatus – Ознакомительный период

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LibraryTrialStatus = Object.LibraryTrialStatus(ProductNumber) Получить свойство (*)
LibraryTrialStatus = Object.GetLibraryTrialStatus(ProductNumber) Получить свойство (**)

Синтаксис COM:

Object.get_LibraryTrialStatus(ProductNumber, &LibraryTrialStatus) Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

LibraryProductKeyInfo – Информация о текущей сессии

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LibraryProductKeyInfo = Object.LibraryProductKeyInfo(ProductNumber) Получить свойство (*)
LibraryProductKeyInfo = Object.GetLibraryProductKeyInfo(ProductNumber) Получить свойство (**)

Синтаксис COM:

Object.get_LibraryProductKeyInfo(ProductNumber, &LibraryProductKeyInfo) Получить свойство (*)

Входные параметры:

long ProductNumber - номер продукта.

Примечание:

Свойство доступно только для чтения.

LibraryProductName – Получить название продукта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LibraryProductName = Object.LibraryProductName(ProductNumber) Получить свойство (*)
LibraryProductName = Object.GetLibraryProductName(ProductNumber) Получить свойство (**)

Синтаксис COM:

Object.get_LibraryProductName(ProductNumber, &LibraryProductName) Получить свойство (*)

Входные параметры:

long ProductNumber - номер продукта.

Примечание:

Свойство доступно только для чтения.

IApplicationLicenseManager – методы

EnableKompasInvisible – Установить ключ для КОМПАС– Invisible

Интерфейс...

Синтаксис Automation:

BOOL EnableKompasInvisible(BSTR Key, BSTR Signature);

Синтаксис COM:

HRESULT EnableKompasInvisible(BSTR Key, BSTR Signature, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

Key - имя (передается из внешнего приложения),
Signature - лицензия для проверки подлинности (передается из внешнего приложения).

RegisterLibraryNumber – Зарегистрировать номер продукта на сервере лицензий Компас

Интерфейс...

Синтаксис Automation:

long RegisterLibraryNumber(long ProductNumber);

Синтаксис COM:

HRESULT RegisterLibraryNumber(long ProductNumber, long * ProductNumbUnicueId);

Возвращаемое значение:

Уникальный идентификатор продукта.

Входные параметры:

long ProductNumber - номер продукта.

Примечание

Уникальный идентификатор продукта используется для разрегистрации продукта функцией IApplicationLicenseManager::UnRegisterLibraryNumber.

UnRegisterLibraryNumber – Разрегистрировать номер продукта на сервере лицензий Компас

Интерфейс...

Синтаксис Automation:

long UnRegisterLibraryNumber(long ProductNumbUnicueId);

Синтаксис COM:

HRESULT UnRegisterLibraryNumber(long ProductNumbUnicueId, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

long ProductNumber - номер продукта.

Интерфейс IPLMObject

Интерфейс объекта системы версионирования.

Иерархия:

IDispatch

IPLMObject

Версия KOMPAS v19

IPLMObject – свойства

PLMStatus – Статус в системе версионирования

Интерфейс...

Тип данных: из перечисления ksPLMStatusEnum

Синтаксис Automation:

PLMStatus = Object.PLMStatus	Получить свойство (*)
Object.PLMStatus = PLMStatus	Установить свойство (*)
PLMStatus = Object.GetPLMStatus()	Получить свойство (**)
Object.SetPLMStatus(PLMStatus)	Установить свойство (*)

Синтаксис COM:

Object.get_PLMStatus(&PLMStatus)	Получить свойство
Object.put_PLMStatus(PLMStatus)	Установить свойство

PLMChange – Отличие в системе версионирования

Интерфейс...

Тип данных: из перечисления ksPLMChangesEnum.

Синтаксис Automation:

PLMChange = Object.PLMChange	Получить свойство (*)
Object.PLMChange = PLMChange	Установить свойство (*)
PLMChange = Object.GetPLMChange()	Получить свойство (**)
Object.SetPLMChange(PLMChange)	Установить свойство (*)

Синтаксис COM:

Object.get_PLMChange(&PLMChange)	Получить свойство
Object.put_PLMChange(PLMChange)	Установить свойство

Интерфейс IPLMObjectsManager

Менеджер объектов системы версионирования.

Иерархия:

IDispatch

IPLMObjectsManager

Версия KOMPAS v19

IPLMObjectsManager– свойства

PLMChange – Отличие в системе версионирования

Интерфейс...

Тип данных: из перечисления ksPLMChangesEnum.

Синтаксис Automation:

PLMChange = Object.PLMChange(FileName)	Получить свойство (*)
Object.PLMChange(FileName) = PLMChange	Установить свойство (*)
PLMChange = Object.GetPLMChange(FileName)	Получить свойство (**)
Object.SetPLMChange(FileName, PLMChange)	Установить свойство (*)

Синтаксис COM:

Object.get_PLMChange(FileName, &PLMChange)	Получить свойство
Object.put_PLMChange(FileName, PLMChange)	Установить свойство

Входные параметры:

BSTR	- FileName - имя файла.

Примечание

Свойство позволяет получить/установить признак отличия объектов версионирования по имени файла.

PLMStatus – Статус в системе версионирования

Интерфейс...

Тип данных: из перечисления ksPLMStatusEnum.

Синтаксис Automation:

PLMStatus = Object.PLMStatus(FileName)	Получить свойство (*)
Object.PLMStatus(FileName) = PLMStatus	Установить свойство (*)
PLMStatus = Object.GetPLMStatus(FileName)	Получить свойство (**)
Object.SetPLMStatus(FileName, PLMStatus)	Установить свойство (*)

Синтаксис COM:

Object.get_PLMStatus(FileName, &PLMStatus)	Получить свойство
Object.put_PLMStatus(FileName, PLMStatus)	Установить свойство

Входные параметры:

BSTR - FileName - имя файла.

Примечание:

Свойство позволяет получить/установить статус объектов версионирования по имени файла.

IPLMObjectsManager – методы

SetPLMStatusAttrAvailability – Установить доступность в настройках дерева документа пункта меню "ЛОЦМАН Статус"

Интерфейс...

Синтаксис Automation:

BOOL SetPLMStatusAttrAvailability(BOOL Available, BOOL Enabled);

Синтаксис COM :

HRESULT SetPLMStatusAttrAvailability(BOOL Available, BOOL Enabled, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

Available - доступность атрибута “ЛОЦМАН Статус”
в контекстном меню настроек Дерева документа,
Enabled - видимость атрибутов “ЛОЦМАН Статус” в Дереве.

SetPLMChangesAttrAvailability – Установить доступность в настройках Дерева документа пункта меню “ЛОЦМАН Отличия”

Интерфейс...

Синтаксис Automation :

BOOL SetPLMChangesAttrAvailability(BOOL Available, BOOL Enabled);

Синтаксис COM :

HRESULT SetPLMChangesAttrAvailability(BOOL Available, BOOL Enabled, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

Available - доступность атрибута “ЛОЦМАН Отличия” в контекстном меню настроек Дерева документа,
Enabled - видимость в атрибутов “ЛОЦМАН Отличия” в Дереве.

Коллекция документов

Интерфейс IDocuments

Коллекция документов, открытых в приложении КОМПАС-3D.

Иерархия:

IKompasAPIObject
IKompasCollection
IDocuments

Примечание:

Данный интерфейс может быть получен от интерфейса приложения IApplication с помощью свойства IApplication::Documents.

IDocuments – свойства

DocumentSynchronize – Синхронизировать с зависимыми документами

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
DocumentSynchronize = Object.DocumentSynchronize                Получить свойство (* )
Object.DocumentSynchronize = DocumentSynchronize                Установить свойство (* )
DocumentSynchronize = Object.GetDocumentSynchronize()           Получить свойство(**)
Object.SetDocumentSynchronize( DocumentSynchronize )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DocumentSynchronize( &DocumentSynchronize )          Получить свойство
Object.put_DocumentSynchronize(  DocumentSynchronize )          Установить свойство
```

Свойство позволяет устанавливать и получать признак синхронизации с зависимыми документами. Реализовано только для спецификации.

Item – Документ, заданный по имени, ссылке или по индексу

Интерфейс...

Тип данных: указатель на интерфейс документа IKompasDocument

Синтаксис Automation:

```
document = iObject.Item (Index)                 Получить свойство(* )
document = iObject.GetItem (Index)              Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_IItem ( Index, &Document )         Получить свойство
```

Входные параметры:

Index - Тип VARIANT, полное имя PathName документа, ссылка на документ или индекс в коллекции.

Примечание:

1. Свойство доступно только для чтения.
2. Полное имя файла документа состоит из пути, имени и расширения файла.

RecoverError – Признак ошибки после открытия файла с восстановлением

Интерфейс...

Тип данных: ksRecoverErrorEnum

Синтаксис Automation:

```
RecoverError = Object.RecoverError                  Получить свойство (* )
Object.RecoverError = RecoverError                  Установить свойство (* )
RecoverError = Object.GetRecoverError()             Получить свойство(**)
Object.SetRecoverError( RecoverError )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_RecoverError( &RecoverError )            Получить свойство
Object.put_RecoverError(  RecoverError )            Установить свойство
```

RecoverMode – Признак открытия файлов в режиме восстановления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
RecoverMode = Object.RecoverMode                    Получить свойство (* )
Object.RecoverMode = RecoverMode                    Установить свойство (* )
RecoverMode = Object.GetRecoverMode()               Получить свойство(**)
Object.SetRecoverMode( RecoverMode )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_RecoverMode( &RecoverMode )              Получить свойство
Object.put_RecoverMode(  RecoverMode )              Установить свойство
```

RecoverModeErrorList – Список ошибок, исправленных при открытии файла с восстановлением

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
RecoverModeErrorList = Object.RecoverModeErrorList( )      Получить свойство(* )
RecoverModeErrorList = Object.GetRecoverModeErrorList()    Получить свойство(**)
```

Синтаксис COM:

```
Object.get_RecoverModeErrorList( &RecoverModeErrorList )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Список ошибок возвращается в виде массива VT_ARRAY | VT_I4.

IDocuments – методы

Add – Создать новый документ и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (long Type, BOOL Visible);

Синтаксис COM:

HRESULT Add([in] DocumentTypeEnum Type, [in, defaultvalue(TRUE)] VARIANT_BOOL Visible, [out, retval] IKompasDocument** Result);

Входные параметры:

Type - тип документа из перечисления DocumentTypeEnum,
Visible - видимость документа:
TRUE создавать документ в видимом режиме,
FALSE в слепом.

Возвращаемое значение:

указатель на интерфейс IKompasDocument

AddCustomDocument – Создать новый документ по идентификатору и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddCustomDocument(BSTR DocumentTypeId);

Синтаксис COM:

HRESULT AddCustomDocument(BSTR DocumentTypeId, IKompasDocument** Result);

Возвращаемое значение :

указатель на интерфейс IKompasDocument

AddNewDocumentFromTemplate – Создать новый документ по шаблону и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddNewDocumentFromTemplate(BSTR TemplateFileName);

Синтаксис COM:

HRESULT AddNewDocumentFromTemplate(BSTR TemplateFileName, IKompasDocument** Result);

Возвращаемое значение :

указатель на интерфейс IKompasDocument

AddWithDefaultSettings – Создать новый документ с параметрами из настроек для новых документов

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddWithDefaultSettings(DocumentTypeEnum Type, BOOL Visible);

Синтаксис COM:

HRESULT AddWithDefaultSettings(DocumentTypeEnum Type, BOOL Visible, IKompasDocument** Result);

Входные параметры:

Type - тип документа DocumentTypeEnum,
Visible - TRUE видимый,
- FALSE - невидимый.

Возвращаемое значение:

индекс линии разрыва,
указатель на интерфейс IKompasDocument.

GetEmbodimentsTree – Дерево исполнений

Интерфейс...

Синтаксис Automation:

VARIANT GetEmbodimentsTree(BSTR FileName, ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer, BOOL AddSpaces, long * CurrentEmbodiment);

Синтаксис COM:

HRESULT GetEmbodimentsTree(BSTR FileName, ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer, BOOL AddSpaces, long * CurrentEmbodiment, VARIANT * Tree);

Входные параметры:

FileName - Имя файла,
MarkingType - Формат возвращаемых обозначений ksVariantMarkingTypeEnum,
AddSystemDelimer - Возвращать системные разделители в обозначении,
AddSpaces - Добавить пробелы для определения уровня исполнения в дереве исполнений,
CurrentEmbodiment - Индекс текущего исполнения в списке исполнений.

Возвращаемое значение:

- Дерево исполнений. Массив узлов дерева исполнений. SAFEARRAY BSTR - (VT_ARRAY | VT_BSTR)

GetLoadCombinations – Получить массив типов загрузки в виде массива SAFEARRAY BSTR – (VT_ARRAY | VT_BSTR)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1072_116_2_Tipy_zagruzki_sborki.htm

Синтаксис Automation:

VARIANT GetLoadCombinations (LPCTSTR PathName, long * CurrentIndex);

Синтаксис COM:

HRESULT GetLoadCombinations([in] BSTR PathName, [out] long * CurrentIndex, [out, retval] VARIANT * Value);

Входные параметры:

PathName - полное имя файла документа, состоящее из пути, имени и расширения файла.

Выходные параметры:

CurrentIndex - текущий индекс типа загрузки, по которому загружается документ по умолчанию.

Возвращаемое значение:

тип значения VARIANT - массив типов загрузки в виде массива SAFEARRAY BSTR - (VT_ARRAY | VT_BSTR)

Примечание:

Метод позволяет получить массив типов загрузки документа до открытия документа.

Метод в настоящее время используется только для сборки 3D. У сборки 3D есть три умолчательных способа загрузки, индексы которых соответствуют перечислению ksLoadStateEnum:

Полная	0	- полная загрузка,
Пустая	1	- все компоненты не загружены,
Упрощенная	2	- все компоненты загружены частично.

Кроме этого, могут быть еще и пользовательские типы загрузки.

GetLoadCombinationsParam – Получить интерфейс типов загрузки документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetLoadCombinationsParam (BSTR PathName);

Синтаксис COM:

HRESULT GetLoadCombinationsParam (BSTR PathName, ILoadCombinationsParam ** Result);

Входные параметры:

PathName - имя документа

Возвращаемое значение:

Указатель на интерфейс ILoadCombinationsParam

GetOpenDocumentParam – Получить интерфейс параметров открытия документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetOpenDocumentParam();

Синтаксис COM:

HRESULT GetOpenDocumentParam(IOpenDocumentParam ** Result);

Возвращаемое значение:

Указатель на интерфейс IOpenDocumentParam

Open – Открыть документ

Интерфейс...

Синтаксис Automation:

LPDISPATCH Open (LPCTSTR PathName, BOOL Visible, BOOL ReadOnly);

Синтаксис COM:

HRESULT Open([in] BSTR PathName, [in, defaultvalue(TRUE)] VARIANT_BOOL Visible, [in, defaultvalue(FALSE)] VARIANT_BOOL ReadOnly, [out, retval] IKompasDocument** Result);

Входные параметры:

PathName	- полное имя файла документа, состоящее из пути, имени и расширения файла,
Visible	- видимость документа: TRUE создавать документ в видимом режиме, FALSE в невидимом,
ReadOnly	- TRUE - открыть документ только для чтения.

Возвращаемое значение:

указатель на интерфейс IKompasDocument	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Открывает существующий документ и добавляет его в коллекцию.

OpenEx – Открыть документ

Интерфейс...

Синтаксис Automation:

LPDISPATCH OpenEx (LPCTSTR PathName, BOOL Visible, BOOL ReadOnly, VARIANT LoadCombinationIndex);

Синтаксис COM:

HRESULT OpenEx([in] BSTR PathName, [in, defaultvalue(TRUE)] VARIANT_BOOL Visible, [in, defaultvalue(FALSE)] VARIANT_BOOL ReadOnly, [in]VARIANT LoadCombinationIndex, [out, retval] IKompasDocument** Result);

Входные параметры:

PathName	- полное имя файла документа, состоящее из пути, имени и расширения файла,
Visible	- видимость документа: TRUE создавать документ в видимом режиме, FALSE в невидимом,
ReadOnly	- TRUE - открыть документ только для чтения,
LoadCombinationIndex	- тип VARIANT - индекс загружаемого набора или имя набора.

Возвращаемое значение:

указатель на интерфейс IKompasDocument	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

1. Открывает существующий документ и добавляет его в коллекцию.
2. Для сборки 3D можно задать индекс набора для загрузки. Для других типов документов индекс загружаемого набора игнорируется.

OpenDocument – Открыть документ с заданными параметрами открытия документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH OpenDocument(BSTR FileName, IOpenDocumentParam * Param)

Синтаксис COM:

HRESULT OpenDocument(BSTR FileName, IOpenDocumentParam * Param, IKompasDocument ** Result);

Возвращаемое значение:

Указатель на интерфейс документа IKompasDocument

Входные параметры:

FileName - имя файла документа
Param - указатель на интерфейс параметров открытия документа IOpenDocumentParam

Информация об ошибках

Интерфейс IKompasError

Интерфейс информации об ошибке системы КОМПАС.

Иерархия:

IKompasAPIObject

IKompasError

Описание:

1. Позволяет получить информацию об ошибке, возникшей при работе API системы КОМПАС.
2. Системные ошибки описаны двумя перечислениями: Ошибки API ErrorType и Ошибки 3D API ErrorType3d

   В случае возникновения фатальных ошибок большинство методов интерфейсов и экспортных функций API КОМПАС выполняться не будут.

   Если в результате работы библиотеки возникает ошибка, то ее можно снять вызовом одного из следующих свойств и методов: Description, Report или Clear. Если указанные свойства и методы не вызывались, сообщение об ошибке будет автоматически выведено после завершения выполнения команды библиотеки.

   Примечание:

   Данный интерфейс может быть получен от интерфейса приложения IApplication с помощью свойства IApplication::KompasError.

IKompasError– свойства

Code – Код ошибки

Интерфейс...

Тип данных: из перечисления ErrorType и ErrorType3d.

Синтаксис Automation:

Code = iObject.Code Получить свойство (*)

Code = iObject.GetCode() Получить свойство (**)

Синтаксис COM:

iObject->get_Code (&Code) Получить свойство

Примечание:

Свойство доступно только для чтения. Так как коды ошибок из ErrorType и ErrorType3d пересекаются, то для определения из какого перечисления взведена ошибка, нужно воспользоваться свойством Error3D.

Description – Описание ошибки

Интерфейс...

Тип данных:BSTR (строка).

Синтаксис Automation:

Description = iObject.Description Получить свойство (*)
Description = iObject.GetDescription() Получить свойство(**)

Синтаксис COM:

iObject->get_Description (&Description) Получить свойство

Примечание:

Свойство доступно только для чтения. Позволяет получить символьное описание возникшей ошибки в системе КОМПАС. После выдачи описания ошибки она сбрасывается, т.е. устанавливается в etSuccess (Успешное завершение).

Error3D – Ошибка для 3D

Интерфейс...

Тип данных:BOOL.

Синтаксис Automation:

Error3D = iObject.Error3D Получить свойство (*)
Error3D = iObject.GetError3D() Получить свойство(**)

Синтаксис COM:

iObject->get_Error3D (&Error3D) Получить свойство

Значения свойства:

TRUE - ошибка возникла в 3D API (перечисление ErrorType3d),
FALSE - ошибка из перечисления ErrorType.

Примечание:

Свойство доступно только для чтения. Позволяет определить, какому из двух перечислений соответствует код ошибки, полученный через свойство Code

IKompasError- методы

Clear- Сбросить ошибку

Интерфейс...

Синтаксис Automation:

void Clear();

Синтаксис COM:

HRESULT Clear();

Примечание:

В результате выполнения метода код ошибки устанавливается в etSuccess (Успешное завершение).

Report – Вывести сообщение о ошибке

Интерфейс...

Синтаксис Automation:

void Report();

Синтаксис COM:

HRESULT Report();

Примечание:

В результате выполнения этого метода на экран выводится диалог с описанием возникшей ошибки. После выдачи сообщения о ошибке она сбрасывается, то есть устанавливается в etSuccess (Успешное завершение).

Параметры процесса

Интерфейс IProcessParam

Интерфейс событий...

Интерфейс параметров процесса.

Иерархия:

IKompasAPIObject

IProcessParam

Примечание:

1. Интерфейс позволяет управлять параметрами процесса.
2. Данный интерфейс может быть получен от интерфейса приложения IApplication с помощью свойства IApplication::CreateProcessParam.

IProcessParam – свойства

AutoReduce – Завершение процесса автоматически после задания всех параметров

Интерфейс...

Тип данных:BOOL

Синтаксис Automation:

AutoReduce = iObject.AutoReduce	Получить свойство (*)
iObject.AutoReduce = AutoReduce	Установить свойство (*)
AutoReduce = iObject.GetAutoReduce()	Получить свойство(**)
iObject.SetAutoReduce (AutoReduce)	Установить свойство (**)

Синтаксис COM:

iObject->get_AutoReduce (&AutoReduce)	Получить свойство
iObject->put_AutoReduce (AutoReduce)	Установить свойство

Примечание:

С помощью данного свойства можно получить и изменить завершение процесса после задания всех параметров.

BmpBeginId – Начальный диапазон для иконок специальной панели

Интерфейс...

Тип данных long.

Синтаксис Automation:

BmpBeginId(Size) = iObject.BmpBeginId	Получить свойство (*)
iObject.BmpBeginId(Size) = BmpBeginId	Установить свойство (*)
BmpBeginId = iObject.GetBmpBeginId(Size)	Получить свойство(**)
iObject.SetBmpBeginId(Size, BmpBeginId)	Установить свойство (**)

Синтаксис COM:

iObject->get_BmpBeginId(Size,&BmpBeginId)	Получить свойство
iObject->put_BmpBeginId(Size, BmpBeginId)	Установить свойство

Входные параметры:

Size - размер битмапов

Свойство BmpBeginId используется совместно со свойствами ResModuleI и SpecToolbarEx.

Caption – Заголовок процесса

Интерфейс...

Тип данных:BSTR (строка).

Синтаксис Automation:

```
Caption = iObject.Caption                    Получить свойство (* )
iObject.Caption = Caption                    Установить свойство (* )
Caption = iObject.GetCaption()               Получить свойство(**)
iObject.SetCaption (Caption)                 Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Caption (&Caption)              Получить свойство (* )
iObject->put_Caption (Caption)               Установить свойство (* )
```

Примечание:

Позволяет получить и установить заголовок процесса.

DefaultControlFix – Состояние фиксированности для умолчательных элементов управления Панели свойств

Интерфейс...

Тип данных: состояние фиксированности для умолчательных элементов управления панели свойств из перечисления DefaultFixTypeEnum.

Синтаксис Automation:

```
DefaultControlFix = iObject.DefaultControlFix                Получить свойство (* )
iObject.DefaultControlFix = DefaultControlFix                Установить свойство (* )
DefaultControlFix = iObject.GetDefaultControlFix()           Получить свойство(**)
iObject.SetDefaultControlFix (DefaultControlFix)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DefaultControlFix (&DefaultControlFix)          Получить свойство (* )
iObject->put_DefaultControlFix (DefaultControlFix)           Установить свойство (* )
```

Примечание:

С помощью данного свойства можно получить и изменить состояние фиксированности для умолчательных элементов управления Панели свойств.

EnableUndoRedo – Признак обработки процессом Undo/Redo команд

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableUndoRedo = Object.EnableUndoRedo                  Получить свойство (* )
Object.EnableUndoRedo = EnableUndoRedo                  Установить свойство (* )
EnableUndoRedo = Object.GetEnableUndoRedo()             Получить свойство(**)
Object.SetEnableUndoRedo( EnableUndoRedo )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableUndoRedo( &EnableUndoRedo )            Получить свойство (* )
Object.put_EnableUndoRedo(  EnableUndoRedo )            Установить свойство (* )
```

EnterButtonIconType – Тип иконки для кнопки Создать

Интерфейс...

Тип данных: из перечисления ksEnterButtonIconTypeEnum

Синтаксис Automation:

```
EnterButtonIconType = Object.EnterButtonIconType        Получить свойство (* )
Object.EnterButtonIconType = EnterButtonIconType        Установить свойство (* )
EnterButtonIconType = Object.GetEnterButtonIconType()   Получить свойство(**)
Object.SetEnterButtonIconType( EnterButtonIconType )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnterButtonIconType( &EnterButtonIconType )  Получить свойство (* )
Object.put_EnterButtonIconType(  EnterButtonIconType )  Установить свойство (* )
```

Layout – Положение панели свойств (вверху, внизу, слева, справа, плавает)

Функция не поддерживается

Интерфейс...

Тип данных: положение Панели свойств из перечисления PropertyManagerLayout.

Синтаксис Automation:

```
Layout = iObject.Layout()                               Получить свойство (* )
Layout = iObject.GetLayout()                            Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_Layout( &Layout )                          Получить свойство
```

Примечание:

С помощью данного свойства можно получить положение Панели свойств.

Свойство доступно только для чтения.

PropertyTabs – Коллекция вкладок Панели свойств

Интерфейс...

Тип данных:указатель на интерфейс IPropertyTabs.

Синтаксис Automation:

PropertyTabs = iObject.PropertyTabs	Получить свойство (*)
PropertyTabs = iObject.GetPropertyTabs()	Получить свойство(**)

Синтаксис COM:

iObject->get_PropertyTabs (&PropertyTabs)	Получить свойство

Примечание:

1. Свойство только для чтения.
2. Позволяет получить доступ к вкладкам Панели свойств.

ResModule – Модуль с описанием пользовательской спецпанели

Интерфейс...

Тип данных:VARIANT (long или BSTR).

Синтаксис Automation:

ResModule = iObject.ResModule	Получить свойство (*)
iObject.ResModule = ResModule	Установить свойство (*)
ResModule = iObject.GetResModule()	Получить свойство(**)
iObject.SetResModule(ResModule)	Установить свойство (**)

Синтаксис COM:

iObject->get_ResModule(&ResModule)	Получить свойство
iObject->put_ResModule(ResModule)	Установить свойство

Значение свойства:

HINSTANCE (тип значения long)

- полный путь к файлу (тип значения BSTR) dll-модуля, в котором находятся все необходимые ресурсы.

Примечание:

Данное свойство позволяет установить или получить dll-модуль, в котором находится описание пользовательской спецпанели, рисунки для пользовательских кнопок (см. IProcessParam::SpecToolbarEx и IProcessParam::BmpBeginID).

ShowCommandWindow – Показывать командное окно

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowCommandWindow = Object.ShowCommandWindow	Получить свойство (*)
Object.ShowCommandWindow = ShowCommandWindow	Установить свойство (*)
ShowCommandWindow = Object.GetShowCommandWindow()	Получить свойство(**)
Object.SetShowCommandWindow(ShowCommandWindow)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowCommandWindow(&ShowCommandWindow)	Получить свойство
Object.put_ShowCommandWindow(ShowCommandWindow)	Установить свойство

ShowContextMenuOfGeomCalculator – Наличие кнопки Геометрический калькулятор в контекстном меню процесса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowContextMenuOfGeomCalculator = Object.ShowContextMenuOfGeomCalculator	Получить свойство (*)
Object.ShowContextMenuOfGeomCalculator = ShowContextMenuOfGeomCalculator	Установить свойство (*)
ShowContextMenuOfGeomCalculator = Object.GetShowContextMenuOfGeomCalculator()	Получить свойство(**)
Object.SetShowContextMenuOfGeomCalculator(ShowContextMenuOfGeomCalculator)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowContextMenuOfGeomCalculator (&ShowContextMenuOfGeomCalculator)	Получить свойство
Object.put_ShowContextMenuOfGeomCalculator (ShowContextMenuOfGeomCalculator)	Установить свойство

Значение свойства:

TRUE	- команда **Геометрический калькулятор** присутствует в контекстном меню процесса,
FALSE	- команда **Геометрический калькулятор** отсутствует в контекстном меню процесса.

ShowContextMenuOfSnap – Наличие кнопки Привязки в контекстном меню процесса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowContextMenuOfSnap = Object.ShowContextMenuOfSnap	Получить свойство (*)
Object.ShowContextMenuOfSnap = ShowContextMenuOfSnap	Установить свойство (*)
ShowContextMenuOfSnap = Object.GetShowContextMenuOfSnap()	Получить свойство(**)
Object.SetShowContextMenuOfSnap(ShowContextMenuOfSnap)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowContextMenuOfSnap(&ShowContextMenuOfSnap)	Получить свойство
Object.put_ShowContextMenuOfSnap(ShowContextMenuOfSnap)	Установить свойство

Значение свойства:

TRUE	- команда **Привязки** присутствует в контекстном меню процесса,
FALSE	- команда **Привязки** отсутствует в контекстном меню процесса.

SpecToolbar – Специальная панель

Интерфейс...

Тип данных: тип предопределенной спецпанели для Панели свойств из перечисления SpecPropertyToolBarEnum.

Синтаксис Automation:

SpecToolbar = iObject.SpecToolbar	Получить свойство (*)
iObject.SpecToolbar = SpecToolbar	Установить свойство (*)
SpecToolbar = iObject.GetSpecToolbar()	Получить свойство(**)
iObject.SetSpecToolbar (SpecToolbar)	Установить свойство (**)

Синтаксис COM:

iObject->get_SpecToolbar (&SpecToolbar)	Получить свойство
iObject->put_SpecToolbar (SpecToolbar)	Установить свойство

Примечание:

С помощью данного свойства можно получить и изменить тип предопределенной спецпанели для Панели свойств.

SpecToolbarEx – Специальная панель

Интерфейс...

Тип данных long.

Синтаксис Automation:

SpecToolbarEx = iObject.SpecToolbarEx	Получить свойство (*)
iObject.SpecToolbarEx = SpecToolbarEx	Установить свойство (*)
SpecToolbarEx = iObject.GetSpecToolbarEx()	Получить свойство(**)
iObject.SetSpecToolbarEx(SpecToolbarEx)	Установить свойство (**)

Синтаксис COM:

iObject->get_SpecToolbarEx(&SpecToolbarEx)	Получить свойство
iObject->put_SpecToolbarEx(SpecToolbarEx)	Установить свойство

Значение свойства:

long идентификатор спецпанели в Панели свойств.

Примечание:

Данное свойство является расширением свойства SpecToolbar.

Свойство позволяет установить или получить идентификатор спецпанели. Может быть установлен идентификатор из перечисления SpecPropertyToolBarEnum - **Предопределенные спецпанели для панели свойств** либо задан идентификатор ресурса пользовательской спецпанели, уникальный в рамках библиотеки.

Свойство SpecToolbarEx используется совместно со свойствами ResModule и BmpBeginId.

Чтобы описать пользовательскую спецпанель, нужно:

1. Объявить уникальный идентификатор спецпанели в h файле

   ```
   #define ITB_SPEC_TOOLBAR            3000
   #define END_OF_RESOURCE_TABLE       0xffff
   ```

2. В ресурсном файле описать RCDATA панели и пользовательские кнопки

   rc2-файл:

   ```
   ITB_SPEC_TOOLBAR  RCDATA
   {
    1 //pbEnter
    2 //pbEsc
    3 //pbHelpсправка

    20   //пользовательская кнопка
    END_OF_RESOURCE_TABLE
   ```

}

В примере для пользовательской спецпанели определено 4 кнопки.

1, 2, 3 - предопределенные кнопки из перечисления SpecPropertyToolBarEnum

20 - пользовательская кнопка

//битмап для пользовательской кнопки спецпанели

20 BITMAP "res\\st_reduc.bmp" // CREATE процесса настройки фильтра для выбора объектов отчета

//tips\hint для пользовательской кнопки спецпанели

STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE

BEGIN

20 "Для тестирования пользовательской кнопки\nПользовательская кнопка"

END

IProcessParam – методы

AddSetupMenuCommand – Добавить пункт в меню настроек процесса

Интерфейс...

Синтаксис Automation:

BOOL AddSetupMenuCommand(BSTR Title, long Command, BOOL Checable);

Синтаксис COM:

HRESULT AddSetupMenuCommand(BSTR Title, long Command, BOOL Checable, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Title - заголовок,
Command - идентификатор команды,
Checable - FALSE - выполняемая команда,
TRUE - флаг состояния.

AddSpecToolbarButton – Добавить кнопку в спецпанель

Интерфейс...

Синтаксис Automation:

BOOL AddSpecToolbarButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont);

Синтаксис COM:

HRESULT AddSpecToolbarButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BtnID	- идентификатор кнопки,
Bmp	- идентификатор картинки или символа шрифта или путь к файлу,
ips	- имя кнопки,
IconFont	- имя шрифта.

Примечание:

1. Для использования иконок из шрифта Компас нужно передать параметр IconFont == пустой строке.
2. Для использования шрифта библиотеки, описанного в манифесте библиотеки, нужно передать строчку полученную методом IProceduresLibrary::IconsFont.
3. Для использования картинок из ресурсов требуется установить свойство IProcessParam::ResModule. Для использования шрифтовых иконок свойство IProcessParam::ResModule устанавливаться не должно.

ClearSpecToolbar – Очистить спецпанель

Интерфейс...

Синтаксис Automation:

BOOL ClearSpecToolbar( );

Синтаксис COM:

HRESULT ClearSpecToolbar(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetGabaritRect – Получить габаритный прямоугольник Панели свойств

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

void GetGabaritRect(long * left,

long * top,
long * right,
long * bottom);

Синтаксис COM:

HRESULT GetGabaritRect([out]long * left,
[out]long * top,
[out]long * right,
[out]long * bottom);

Выходные параметры:

left, top	- координата верхней левой точки,
right, bottom	- координата нижней правой точки.

Примечание:

Метод позволяет получить габаритный прямоугольник Панели свойств. Точки задаются в относительных координатах окна КОМПАС.

PopProcessName – Убрать имя подпроцесса

Интерфейс...

Синтаксис Automation:

BOOL PopProcessName();

Синтаксис COM:

HRESULT PopProcessName(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

PushProcessName – Установить имя подпроцесса

Интерфейс...

Синтаксис Automation:

BOOL PushProcessName(BSTR Name);

Синтаксис COM:

HRESULT PushProcessName(BSTR Name, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Name - добавочное имя подпроцесса.

SetSetupMenuCommandState – Установить состояние команды меню настроек процесса

Интерфейс...

Синтаксис Automation:

BOOL SetSetupMenuCommandState(long Command, BOOL Visible, BOOL Enable, BOOL Checked);

Синтаксис COM:

HRESULT SetSetupMenuCommandState(long Command, BOOL Visible, BOOL Enable, BOOL Checked, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Command - идентификатор команды,
Visible - видимость,
Enable - доступность,
Checked - признак выбора.

Интерфейс IProcess

Интерфейс процесса.

Иерархия:

IDispatch

IKompasAPIObject

IProcess

Примечание:

Интерфейс можно получить с помощью методов:

IKompasDocument2D1::LibProcess

IKompasDocument3D1::LibProcess

IProcess – свойства

Caption – Заголовок процесса

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Caption= Object.CursorId                    Получить свойство (* )
Object.Caption= CursorId                    Установить свойство (* )
Caption= Object.GetCursorId()               Получить свойство(**)
Object.SetCursorId( Caption)                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Caption( &Caption )              Получить свойство
Object.put_Caption(  Caption )              Установить свойство
```

Critical – Признак критического процесса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Critical = Object.Critical                  Получить свойство (* )
Object.Critical = Critical                  Установить свойство (* )
Critical = Object.GetCritical()             Получить свойство(**)
Object.SetCritical( Critical )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Critical( &Critical )            Получить свойство
Object.put_Critical(  Critical )            Установить свойство
```

CursorId – Строка с именем стандартного курсора или идентификатор

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
CursorId = Object.CursorId                  Получить свойство (* )
Object.CursorId = CursorId                  Установить свойство (* )
CursorId = Object.GetCursorId()             Получить свойство(**)
Object.SetCursorId( CursorId )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CursorId( &CursorId )            Получить свойство
Object.put_CursorId(  CursorId )            Установить свойство
```

Dynamic – Признак динамического запроса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Dynamic = Object.Dynamic                Получить свойство (* )
Object.Dynamic = Dynamic                Установить свойство (* )
Dynamic = Object.GetDynamic()           Получить свойство(**)
Object.SetDynamic( Dynamic )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Dynamic( &Dynamic )          Получить свойство
Object.put_Dynamic(  Dynamic )          Установить свойство
```

Menu – Меню

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Menu = Object.Menu                      Получить свойство (* )
Object.Menu = Menu                      Установить свойство (* )
Menu = Object.GetMenu()                 Получить свойство(**)
Object.SetMenu( Menu )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Menu( &Menu )                Получить свойство
Object.put_Menu(  Menu )                Установить свойство
```

ProcessParam – Параметры процесса

Интерфейс...

Тип данных: Указатель на интерфейс IProcessParam

Синтаксис Automation:

```
ProcessParam = Object.ProcessParam              Получить свойство (* )
Object.ProcessParam = ProcessParam              Установить свойство (* )
ProcessParam = Object.GetProcessParam()         Получить свойство(**)
Object.SetProcessParam( ProcessParam )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ProcessParam( &ProcessParam )        Получить свойство
Object.put_ProcessParam(  ProcessParam )        Установить свойство
```

Prompt – Строка приглашения

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Prompt= Object.Prompt	Получить свойство (*)
Object.Prompt= Prompt	Установить свойство (*)
Prompt= Object.GetPrompt()	Получить свойство(**)
Object.SetPrompt(Prompt)	Установить свойство (**)

Синтаксис COM:

Object.get_Prompt(&Prompt)	Получить свойство
Object.put_Prompt(Prompt)	Установить свойство

ResModule – Модуль с описанием пользовательской спецпанели

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ResModule = Object.ResModule	Получить свойство (*)
Object.ResModule = ResModule	Установить свойство (*)
ResModule = Object.GetResModule()	Получить свойство(**)
Object.SetResModule(ResModule)	Установить свойство (**)

Синтаксис COM:

Object.get_ResModule(&ResModule)	Получить свойство
Object.put_ResModule(ResModule)	Установить свойство

IProcess – методы

Run – Запустить процесс

Интерфейс...

Синтаксис Automation:

BOOL Run(BOOL Modal, BOOL PostMessage);

Синтаксис COM:

HRESULT Run(BOOL Modal, BOOL PostMessage, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Входные параметры:

Modal	TRUE - модальный процесс, FALSE - немодальный процесс,
PostMessage	TRUE - отложенный запуск для немодального процесса.

Примечание:

- ▼ При запуске модального процесса Функция ожидает завершение работы процесса.
- ▼ При запуске немодального процесса процесс начнет работу позже.
- ▼ Модальный процесс может быть запущен только один.
- ▼ Немодальные процессы можно запустить в разных документах.

SetCursorText – Установить текст курсора

Интерфейс...

Синтаксис Automation:

BOOL SetCursorText(BSTR PVal);

Синтаксис COM:

HRESULT SetCursorText(BSTR PVal, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

PVal - Строка для текста курсора.

Stop – Остановить процесс

Интерфейс...

Синтаксис Automation:

BOOL Stop();

Синтаксис COM:

HRESULT Stop(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Update – Обновить параметры процесса

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IProcess2D

Интерфейс процесса 2D.

Иерархия:

IDispatch

IProcess

IProcess2D

Примечание:

Интерфейс является дополнительным для IProcess.

Интерфейс можно получить с помощью метода: IKompasDocument2D1::LibProcess

IProcess2D – свойства

Angle – Угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Phantom2D – Параметры фантома

Интерфейс...

Тип данных: Указатель на интерфейс IPhantom2D

Phantom2D = Object.Phantom2D	Получить свойство(*)
Phantom2D = Object.GetPhantom2D()	Получить свойство (**)

Синтаксис COM:

Object.get_Phantom2D(&Phantom2D))	Получить свойство

Примечание:

Свойство доступно только для чтения.

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX())	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY())	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Интерфейс IProcess3D

Интерфейс процесса 3D.

Иерархия:

IDispatch

IProcess

IProcess3D

Примечание:

Интерфейс является дополнительным для IProcess.

Интерфейс можно получить с помощью метода: IKompasDocument3D1::LibProcess

IProcess3D – свойства

ClearProcessUndo – Очистить стэк команд Undo процесса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClearProcessUndo = Object.ClearProcessUndo	Получить свойство (*)
Object.ClearProcessUndo = ClearProcessUndo	Установить свойство (*)
ClearProcessUndo = Object.GetClearProcessUndo()	Получить свойство (**)
Object.SetClearProcessUndo(ClearProcessUndo)	Установить свойство (**)

Синтаксис COM:

Object.get_ClearProcessUndo(&ClearProcessUndo)	Получить свойство
Object.put_ClearProcessUndo(ClearProcessUndo)	Установить свойство

Manipulators – Коллекция манипуляторов

Интерфейс...

Тип данных: Указатель на интерфейс IManipulators

Синтаксис Automation:

Manipulators = Object.Manipulators	Получить свойство (*)
Manipulators = Object.GetManipulators()	Получить свойство (**)

Синтаксис COM:

Object.get_Manipulators(&Manipulators)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MateConstraints – Коллекция сопряжений

Интерфейс...

Тип данных: Указатель на интерфейс IMateConstraints3D

Синтаксис Automation:

MateConstraints = Object.MateConstraints ;	Получить свойство (*)

MateConstraints = Object.GetMateConstraints() Получить свойство (**)

Синтаксис COM:

Object.get_MateConstraints(&MateConstraints); Получить свойство

Примечание:

Свойство доступно только для чтения.

MateConstraintsObjects – Выбранные объекты для сопряжений

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

MateConstraintsObjects = Object.MateConstraintsObjects Получить свойство(*)
Object.MateConstraintsObjects = MateConstraintsObjects Установить свойство (*)
MateConstraintsObjects = Object.GetMateConstraintsObjects() Получить свойство (**)
Object.SetMateConstraintsObjects(MateConstraintsObjects) Установить свойство (**)

Синтаксис COM:

Object.get_MateConstraintsObjects(&MateConstraintsObjects); Получить свойство
Object.put_MateConstraintsObjects(MateConstraintsObjects); Установить свойство

ObjectsFilter3D – Способ фильтрации 3D объектов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ObjectsFilter3D = Object.ObjectsFilter3D(Type) Получить свойство (*)
Object.ObjectsFilter3D(Type) = ObjectsFilter3D Установить свойство (*)
ObjectsFilter3D = Object.GetObjectsFilter3D(Type) Получить свойство (**)
Object.SetObjectsFilter3D(Type, ObjectsFilter3D) Установить свойство (**)

Синтаксис COM:

Object.get_ObjectsFilter3D(Type, &ObjectsFilter3D) Получить свойство
Object.put_ObjectsFilter3D(Type, ObjectsFilter3D) Установить свойство

Входные параметры:

ksProcessObjectsFilter3DEnum - Type - Режим использования прямоугольной рамки для выделения объектов.

PhantomObject – Фантом

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
PhantomObject = Object.PhantomObject                Получить свойство(* )
Object.PhantomObject = PhantomObject                Установить свойство (* )
PhantomObject = Object.GetPhantomObject()           Получить свойство (**)
Object.SetPhantomObject( PhantomObject )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PhantomObject( &PhantomObject );         Получить свойство
Object.put_PhantomObject(  PhantomObject );         Установить свойство
```

Placement – Интерфейс локальной системы координат (положение объекта)

Интерфейс...

Тип данных: Указатель на интерфейс IPlacement3D

Синтаксис Automation:

```
Placement = Object.Placement;                       Получить свойство (* )
Placement = Object.GetPlacement()                   Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Placement( &Placement );                 Получить свойство
```

Примечание:

Свойство доступно только для чтения.

SelectionBandMode – Режим использования прямоугольной рамки для выделения объектов

Интерфейс...

Тип данных: из перечисления ksSelectionBandMode

Синтаксис Automation:

```
SelectionBandMode = Object.SelectionBandMode        Получить свойство (* )
Object.SelectionBandMode = SelectionBandMode        Установить свойство (* )
SelectionBandMode = Object.GetSelectionBandMode()   Получить свойство (**)
Object.SetSelectionBandMode( SelectionBandMode )    Установить свойство (**)
```

Синтаксис COM:

Object.get_SelectionBandMode(&SelectionBandMode) Получить свойство
Object.put_SelectionBandMode(SelectionBandMode) Установить свойство

Примечание:

Свойство позволяет включить режим группового выделения объектов рамкой в процессе.

При групповом селектировании рамкой будет посылаться событие ksProcess3DNotify::ProcessingGroupObjects.

TakeProcessObject – Объект, создаваемый в подпроцессе

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

TakeProcessObject = Object.TakeProcessObject Получить свойство(*)
Object.TakeProcessObject = TakeProcessObject Установить свойство (*)
TakeProcessObject = Object.GetTakeProcessObject() Получить свойство (**)
Object.SetTakeProcessObject(TakeProcessObject) Установить свойство (**)

Синтаксис COM:

Object.get_TakeProcessObject(&PhantomObject); Получить свойство
Object.put_TakeProcessObject(TakeProcessObject); Установить свойство

IProcess3D – методы

RunTakeCreateObjectProcess – Запустить подчиненный режим создания объектов

Интерфейс...

Синтаксис Automation:

BOOL RunTakeCreateObjectProcess(ProcessTypeEnum ProcessType, LPDISPATCH TakeObject, BOOL NeedCreateTakeObj, BOOL LostTakeObj);

Синтаксис COM:

HRESULT RunTakeCreateObjectProcess(ProcessTypeEnum ProcessType, IModelObject * TakeObject, BOOL NeedCreateTakeObj, BOOL LostTakeObj, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Выходные параметры:

ProcessType - тип подпроцесса,
TakeObject - указатель на интерфейс редактируемого в подпроцессе объекта,
NeedCreateTakeObj - необходимость создания объекта в подпроцессе,
LostTakeObj - фиксировать параметры создаваемого объекта.

Функция позволяет запустить следующие подпроцессы:

prPoint3D	Точка 3D
prAxisByDirection	Ось через вершину по объекту
prCAxis2Points	Ось по двум точкам
prCAxis2Planes	Ось по двум плоскостям
prCAxisConeface	Ось конической грани
prCAxisEdge	Ось проходящая через ребро
prContour3D	Контур 3D
prLocalCoordinateSystem	Локальная система координат
prIsoparamCurve	Изопараметрическая кривая
prEquidistant3D	Эквидистанта 3D
prCPlaneOffset	Смещённая плоскость
prCPlane3Points	Плоскость по 3-м точкам
prCPlaneAngle	Плоскость под углом
prCPlaneEdgePoint	Плоскость через ребро и вершину
prCPlaneParallel	Плоскость через вершину параллельно другой плоскости
prCPlanePerpendicular	Плоскость через вершину перпендикулярно ребру
prCPlaneNormalToSurface	Нормальная плоскость
prCPlaneTangentToSurface	Касательная плоскость
prCPlaneLineToEdge	Плоскость через ребро параллельно/ перпендикулярно другому ребру

prCPlaneLineToFlat	Плоскость через ребро параллельно/ перпендикулярно грани
prCPlaneMiddle	Средняя плоскость
prCPlaneTangentAtPoint	Плоскость, касательная к грани в точке
prCPlaneAtCurve	Плоскость через плоскую кривую
prMateParallel	Сопряжения компонентов - Параллельность
prMatePerpendicular	Сопряжения компонентов - Перпендикулярность
prMateOnDistance	Сопряжения компонентов - На расстоянии
prMateOnAngle	Сопряжения компонентов - Под углом
prMateTangent	Сопряжения компонентов - Касание
prMateConcentric	Сопряжения компонентов - Соосность
prMateCoincident	Сопряжения компонентов - Совпадение
prMateSymmetry	Сопряжения компонентов - Симметрия
prMateDependent	Сопряжения компонентов - Зависимое положение

Интерфейс IProgressBarIndicator

Интерфейс индикатора прогресса.

Иерархия:

IKompasAPIObject

IProgressBarIndicator

Описание:

Позволяет запускать, устанавливать текущее значение, останавливать индикатор прогресса, устанавливать текст в строке состояния.

Примечание:

1. Данный интерфейс можно получить от интерфейса IApplication с помощью метода IApplication::ProgressBarIndicator.
2. Для работы с индикатором прогресса необходимо, получив указатель на интерфейс индикатора, запустить его командой IProgressBarIndicator::Star. Текущие значения и текст устанавливать командой IProgressBarIndicator::SetProgress. По завершении отображае-

мого процесса завершить работу с индикатором командой IProgressBarIndicator::Stop. Для отображения следующего процесса необходимо повторно запустить индикатор (при необходимости с новыми параметрами) от ранее полученного указателя на интерфейс.

3. Команда IProgressBarIndicator::SetText выполняется независимо от того, запущен индикатор прогресса или нет.
4. Если получены несколько указателей на интерфейс IProgressBarIndicator, и по одному из указателей индикатор запущен командой IProgressBarIndicator::Start, по любому другому указателю может быть выполнена команда IProgressBarIndicator::SetProgress или IProgressBarIndicator::Stop. Команда IProgressBarIndicator::Start повторно не выполняется до остановки индикатора.

IProgressBarIndicator – методы

SetProgress – Установить текущее значение индикатора

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetProgressBar

Синтаксис Automation:

void SetProgress (long currentVal, LPCTSTR newText, BOOL resetText);

Синтаксис COM:

HRESULT SetProgress (long currentVal, BSTR newText, VARIANT_BOOL resetText);

Входные параметры:

currentVal	- текущее значение значение индикатора,
newText	- текст в строке состояния,
resetText	- TRUE - обновить текст в строке состояния.

Примечание:

Метод выполняется, если индикатор прогресса запущен командой IProgressBarIndicator::Start.

SetText – Установить текст в строке состояния индикатора

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetProgressText.

Синтаксис Automation:

void SetText (LPCTSTR newText);

Синтаксис COM:

HRESULT SetText (BSTR newText);

Входные параметры:

newText	- текст в строке состояния,

Примечание:

1. Метод выполняется как при запущенном индикаторе прогресса командой IProgressBarIndicator::Start, так и при остановленном.
2. Если команда выполняется при запущенном индикаторе прогресса, устанавливаемый текст запоминается как последний текст перед запуском индикатора, и после остановки индикатора командой IProgressBarIndicator::Stop в строке состояния будет отображаться текст newText.

Start – Запустить индикатор прогресса

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksStartProgressBar.

Синтаксис Automation:

void Start (long minVal, long maxVal, LPCTSTR newText, BOOL resetText);

Синтаксис COM:

HRESULT Start (long minVal, long maxVal, BSTR newText, VARIANT_BOOL resetText);

Входные параметры:

minVal	- минимальное значение шкалы,
maxVal	- максимальное значение шкалы,
newText	- текст в строке состояния,
resetText	- TRUE - обновить текст в строке состояния.

Примечание:

После завершения работы индикатор прогресса должен быть остановлен командой IProgressBarIndicator::Stop.

Stop – Остановить индикатор прогресса

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksStopProgressBar.

Синтаксис Automation:

void Stop (LPCTSTR newText, BOOL resetTxt);

Синтаксис COM:

HRESULT Stop (BSTR newText, VARIANT_BOOL resetTxt);

Входные параметры:

newText	- текст в строке состояния,
resetText	- TRUE - обновить текст в строке состояния.

Примечание:

Метод выполняется, если индикатор прогресса запущен командой IProgressBarIndicator::Start.

Интерфейс IProcessInfoWindow

Интерфейс информационного окна процесса.

Иерархия:

IKompasAPIObject

IProcessInfoWindow

Примечание:

Интерфейс является дополнительным к интерфейсу IProcessParam и позволяет создавать информационное окно, принадлежащее процессу с возможностью чтения/записи текста и управления видимостью окна, а также дополнительными свойствами.

Данный интерфейс можно получить у интерфейса IProcessParam процесса, в котором необходимо показать окно, посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);

IProcessInfoWindow – свойства

CloseProcess – Закрытие процесса при закрытии окна

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE - закрывать процесс, которому принадлежит окно,
FALSE - не закрывать.

Синтаксис Automation:

```
CloseProcess = iObject.CloseProcess;          Получить свойство (* )
iObject.CloseProcess = CloseProcess;          Установить свойство (* )
CloseProcess = iObject.GetCloseProcess();     Получить свойство(**)
iObject.SetCloseProcess( CloseProcess );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CloseProcess( &CloseProcess );     Получить свойство
iObject->put_CloseProcess( CloseProcess );      Установить свойство
```

CreateWindow – Управление созданием окна в процессе

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - создавать окно в процессе,
FALSE - не создавать окно.

Синтаксис Automation:

```
CreateWindow = iObject.CreateWindow;                Получить свойство (* )
iObject.CreateWindow = CreateWindow;                Установить свойство (* )
CreateWindow = iObject.GetCreateWindow();           Получить свойство(**)
iObject.SetCreateWindow( CreateWindow );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CreateWindow( &CreateWindow );         Получить свойство
iObject->put_CreateWindow( CreateWindow );          Установить свойство
```

HelpId – Идентификатор справки

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
HelpId = Object.HelpId                              Получить свойство (* )
Object.HelpId = HelpId                              Установить свойство (* )
HelpId = Object.GetHelpId()                         Получить свойство(**)
Object.SetHelpId( HelpId )                          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HelpId( &HelpId )                        Получить свойство
Object.put_HelpId(  HelpId )                        Установить свойство
```

Text – Текст в окне

Интерфейс...

Тип данных: BSTR.

Синтаксис Automation:

```
Text = iObject.Text;                                Получить свойство (* )
iObject.Text = Text;                                Установить свойство (* )
Text = iObject.GetText();                           Получить свойство(**)
iObject.SetText( Text );                            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Text( &Text );                         Получить свойство
iObject->put_Text( Text );                          Установить свойство
```

Примечание:

Позволяет устанавливать и получать текст в окне.

Visible – Видимость окна

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE - окно видимое,
FALSE - окно скрытое.

Синтаксис Automation:

```
Visible = iObject.Visible;                  Получить свойство (* )
iObject.Visible = Visible;                  Установить свойство (* )
Visible = iObject.GetVisible();             Получить свойство(**)
iObject.SetVisible( Visible );              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Visible( &Visible );           Получить свойство
iObject->put_Visible( Visible );            Установить свойство
```

WindowCaption – Заголовок окна

Интерфейс...

Тип данных: BSTR.

Синтаксис Automation:

```
WindowCaption = iObject.WindowCaption;              Получить свойство (* )
iObject.WindowCaption = WindowCaption;              Установить свойство (* )
WindowCaption = iObject.GetWindowCaption();         Получить свойство(**)
iObject.SetWindowCaption( WindowCaption );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_WindowCaption( &WindowCaption );       Получить свойство
iObject->put_WindowCaption( WindowCaption );        Установить свойство
```

Примечание:

Позволяет устанавливать и получать заголовок окна.

Интерфейс IApplicationDialogs

Интерфейс доступа к диалогам Компас.

Иерархия:

IDispatch

IApplicationDialogs

Интерфейс является дополнительным для IApplication.

IApplicationDialogs – методы

GetDialogParam – Получить интерфейс параметров для вызова диалога

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDialogParam(KompasAPIObjectTypeEnum ParamType);

Синтаксис COM:

HRESULT GetDialogParam(KompasAPIObjectTypeEnum ParamType, IKompasAPIObject * * Result);

Возвращаемое значение:

- интерфейс параметров диалога в зависимости от параметра ParamType.

Входные параметры:

ParamType	- тип параметров диалога.
ksObjectThreadDialogParam	- параметры диалога Выбор стандарта резьбы IThreadDialogParam для функции IApplicationDialogs::SelectThread.

SelectThread – Выбор стандарта резьбы

Интерфейс...

Синтаксис Automation:

BOOL SelectThread(OLE_HANDLE ParentHwnd, IThreadDialogParam * DialogParam);

Синтаксис COM:

HRESULT SelectThread(OLE_HANDLE ParentHwnd, IThreadDialogParam * DialogParam, BOOL * Result);

Возвращаемое значение:

TRUE	- если пользователь закрывает диалог с выбором значения,
FALSE	- при отмене или ошибке.

Входные параметры:

TParentHwnd	- дескриптор родительского окна,
DialogParam	- параметры диалога.

ShowContentDialog – Диалог с произвольным наполнением

Интерфейс...

Синтаксис Automation:

long ShowContentDialog(OLE_HANDLE ParentHwnd, IContentDialogParam * DialogParam);

Синтаксис COM:

HRESULT ShowContentDialog(OLE_HANDLE ParentHwnd, IContentDialogParam * DialogParam, long * Result);

Возвращаемое значение:

- результат работы диалога.

Входные параметры:

TParentHwnd - дескриптор родительского окна,
DialogParam - параметры диалога IContentDialogParam.

WhatsWrongDlg – Диалог "Что неверно"

Интерфейс...

Синтаксис Automation:

long WhatsWrongDlg(VARIANT Objs);

Синтаксис COM:

HRESULT WhatsWrongDlg(VARIANT Objs, long * Result);

Возвращаемое значение:

- результат работы диалога.

Входные параметры:

Objs - массив объектов документа.

Интерфейс IContentDialogParam

Интерфейс параметров диалога с произвольным контентом.

Иерархия:

IDispatch
    IKompasAPIObject
        IContentDialogParam

Данный интерфейс можно получить с помощью метода IApplicationDialogs::GetDialogParam, используя константу ksObjectContentDialogParam.

Интерфейс событий...- ksContentDialogNotify

КОМПАС версия v18

IContentDialogParam – свойства

AdditionalButton – Текст дополнительной кнопки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
AdditionalButton = Object.AdditionalButton;                Получить свойство (* )
Object.AdditionalButton = AdditionalButton;                Установить свойство (* )
AdditionalButton = Object.GetAdditionalButton();           Получить свойство(**)
Object.SetAdditionalButton( AdditionalButton );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AdditionalButton( &AdditionalButton );          Получить свойство
Object.put_AdditionalButton(  AdditionalButton );          Установить свойство
```

Примечание:

Если свойство не задано, кнопка не отображается.

AddLeftSeparator – Добавить разделитель в левую панель

Интерфейс...

Синтаксис Automation:

BOOL AddLeftSeparator();

Синтаксис COM:

HRESULT AddLeftSeparator(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

CancelButton – Текст кнопки Отмена

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
CancelButton = Object.CancelButton;          Получить свойство (* )
Object.CancelButton = CancelButton;          Установить свойство (* )
CancelButton = Object.GetCancelButton();     Получить свойство(**)
Object.SetCancelButton( CancelButton );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CancelButton( &CancelButton );    Получить свойство
Object.put_CancelButton(  CancelButton );    Установить свойство
```

Примечание:

Если свойство не задано, кнопка не отображается.

CheckBoxChecked – Состояние чекбокса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CheckBoxChecked = Object.CheckBoxChecked;                Получить свойство (* )
Object.CheckBoxChecked = CheckBoxChecked;                Установить свойство (* )
CheckBoxChecked = Object.GetCheckBoxChecked();           Получить свойство(**)
Object.SetCheckBoxChecked( CheckBoxChecked );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CheckBoxChecked( &CheckBoxChecked );          Получить свойство
Object.put_CheckBoxChecked(  CheckBoxChecked );          Установить свойство
```

CheckBoxTitle – Текст чекбокса

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
CheckBoxTitle = Object.CheckBoxTitle;        Получить свойство (* )
Object.CheckBoxTitle = CheckBoxTitle;        Установить свойство (* )
CheckBoxTitle = Object.GetCheckBoxTitle();   Получить свойство(**)
Object.SetCheckBoxTitle( CheckBoxTitle );    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CheckBoxTitle( &CheckBoxTitle );      Получить свойство
Object.put_CheckBoxTitle(  CheckBoxTitle );      Установить свойство
```

Примечание:

Если свойство не задано, чекбокс не отображается.

Height – Высота диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Height = Object.Height;                 Получить свойство (* )
Object.Height = Height;                 Установить свойство (* )
Height = Object.GetHeight();            Получить свойство(**)
Object.SetHeight( Height );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Height( &Height );           Получить свойство
Object.put_Height(  Height );           Установить свойство
```

HelpId – Идентификатор справки

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
HelpId = Object.HelpId;                 Получить свойство (* )
Object.HelpId = HelpId;                 Установить свойство (* )
HelpId = Object.GetHelpId();            Получить свойство(**)
Object.SetHelpId( HelpId );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HelpId( &HelpId );           Получить свойство
Object.put_HelpId(  HelpId );           Установить свойство
```

Примечание:

Если идентификатор справки не задан, кнопка справки не отображается.

HelpFileName – Имя файла справки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
HelpFileName = Object.HelpFileName;             Получить свойство (* )
Object.HelpFileName = HelpFileName;             Установить свойство (* )
HelpFileName = Object.GetHelpFileName();        Получить свойство(**)
Object.SetHelpFileName( HelpFileName );         Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HelpFileName( &HelpFileName );   Получить свойство
Object.put_HelpFileName(  HelpFileName );   Установить свойство
```

MaxHeight – Максимальная высота диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MaxHeight = Object.MaxHeight;               Получить свойство (* )
Object.MaxHeight = MaxHeight;               Установить свойство (* )
MaxHeight = Object.GetMaxHeight();          Получить свойство(**)
Object.SetMaxHeight( MaxHeight );           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MaxHeight( &MaxHeight );         Получить свойство
Object.put_MaxHeight(  MaxHeight );         Установить свойство
```

MaxWidth – Максимальная ширина диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MaxWidth = Object.MaxWidth;                 Получить свойство (* )
Object.MaxWidth = MaxWidth;                 Установить свойство (* )
MaxWidth = Object.GetMaxWidth();            Получить свойство(**)
Object.SetMaxWidth( MaxWidth );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MaxWidth( &MaxWidth );           Получить свойство
Object.put_MaxWidth(  MaxWidth );           Установить свойство
```

MinHeight – Минимальная высота диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MinHeight = Object.MinHeight;               Получить свойство (* )
Object.MinHeight = MinHeight;               Установить свойство (* )
MinHeight = Object.GetMinHeight();          Получить свойство(**)
Object.SetMinHeight( MinHeight );           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MinHeight( &MinHeight );     Получить свойство
Object.put_MinHeight(  MinHeight );     Установить свойство
```

MinWidth – Минимальная ширина диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MinWidth = Object.MinWidth;                 Получить свойство (* )
Object.MinWidth = MinWidth;                 Установить свойство (* )
MinWidth = Object.GetMinWidth();            Получить свойство(**)
Object.SetMinWidth( MinWidth );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MinWidth( &MinWidth );       Получить свойство
Object.put_MinWidth(  MinWidth );       Установить свойство
```

NegativeButton – Текст кнопки с отрицательным результатом. Кнопка 'Нет'

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
NegativeButton = Object.NegativeButton;                 Получить свойство (* )
Object.NegativeButton = NegativeButton;                 Установить свойство (* )
NegativeButton = Object.GetNegativeButton();            Получить свойство(**)
Object.SetNegativeButton( NegativeButton );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_NegativeButton( &NegativeButton );       Получить свойство
Object.put_NegativeButton(  NegativeButton );       Установить свойство
```

Примечание:

Если свойство не задано, кнопка не отображается.

PositiveButton – Текст кнопки с положительным результатом. Кнопка 'Да'

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
PositiveButton = Object.PositiveButton;          Получить свойство (* )
Object.PositiveButton = PositiveButton;          Установить свойство (* )
PositiveButton = Object.GetPositiveButton();     Получить свойство(**)
Object.SetPositiveButton( PositiveButton );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PositiveButton( &PositiveButton );    Получить свойство
Object.put_PositiveButton(  PositiveButton );    Установить свойство
```

Примечание:

Если свойство не задано, кнопка не отображается.

Resizable – Доступно ли изменение размера

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Resizable = Object.Resizable;                    Получить свойство (* )
Object.Resizable = Resizable;                    Установить свойство (* )
Resizable = Object.GetResizable();               Получить свойство(**)
Object.SetResizable( Resizable );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Resizable( &Resizable );              Получить свойство
Object.put_Resizable(  Resizable );              Установить свойство
```

Title – Заголовок

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Title = Object.Title;                            Получить свойство (* )
Object.Title = Title;                            Установить свойство (* )
Title = Object.GetTitle();                       Получить свойство(**)
Object.SetTitle( Title );                        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Title( &Title );                      Получить свойство
Object.put_Title(  Title );                      Установить свойство
```

UpperCaseTitle – Заголовок диалога переводить в верхний регистр

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
UpperCaseTitle = Object.UpperCaseTitle;          Получить свойство (* )
Object.UpperCaseTitle = UpperCaseTitle;          Установить свойство (* )
UpperCaseTitle = Object.GetUpperCaseTitle();     Получить свойство(**)
Object.SetUpperCaseTitle( UpperCaseTitle );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_UpperCaseTitle( &UpperCaseTitle );    Получить свойство
Object.put_UpperCaseTitle(  UpperCaseTitle );    Установить свойство
```

Width – Ширина диалога

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width;                Получить свойство (* )
Object.Width = Width;                Установить свойство (* )
Width = Object.GetWidth();           Получить свойство(**)
Object.SetWidth( Width );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width );          Получить свойство
Object.put_Width(  Width );          Установить свойство
```

IContentDialogParam – методы

AddLeftToolButton – Добавить кнопку в левую панель

Интерфейс...

Синтаксис Automation:

BOOL AddLeftToolButton(BSTR Header, BOOL HeaderVisible, long CommandID, long IconId, BSTR IconFont, ButtonTypeEnum ButtonType);

Синтаксис COM:

HRESULT AddLeftToolButton(BSTR Header, BOOL HeaderVisible, long CommandID, long IconId, BSTR IconFont, ButtonTypeEnum ButtonType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Header - имя кнопки,
HeaderVisible - отображать имя кнопки на панели,
CommandID - идентификатор команды,
IconId - идентификатор иконки из шрифта,
IconFont - имя шрифта,
ButtonType - тип кнопки из перечисления ButtonTypeEnum.

AddRightToolButton – Добавить кнопку в правую панель

Интерфейс...

Синтаксис Automation:

BOOL AddRightToolButton(BSTR Header, BOOL HeaderVisible, long CommandID, long IconId, BSTR IconFont, ButtonTypeEnum ButtonType);

Синтаксис COM:

HRESULT AddRightToolButton(BSTR Header, BOOL HeaderVisible, long CommandID, long IconId, BSTR IconFont, ButtonTypeEnum ButtonType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Header - имя кнопки,
HeaderVisible - отображать имя кнопки на панели,
CommandID - идентификатор команды,
IconId - идентификатор иконки из шрифта,
IconFont - имя шрифта,
ButtonType - тип кнопки из перечисления ButtonTypeEnum.

AddRightSeparator – Добавить разделитель в правую панель

Интерфейс...

Синтаксис Automation:

BOOL AddRightSeparator();

Синтаксис COM:

HRESULT AddRightSeparator(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Интерфейс ISerializer

Интерфейс сериализации.

Иерархия:

IDispatch

ISerializer

ISerializer – свойства

XML – XML Сериализация/десериализация

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
XML = Object.XML;                  Получить свойство (* )
Object.XML = XML;                  Установить свойство (* )
XML = Object.GetXML();             Получить свойство(**)
Object.SetXML( XML );              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_XML( &XML );            Получить свойство
Object.put_XML(  XML );            Установить свойство
```

Интерфейс IManipulators

Интерфейс коллекции манипуляторов.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IManipulators

Событие ksProcess3DManipulatorsNotify

Примечание

Получить интерфейс коллекции манипуляторов можно, используя свойство процесса 3D IProcess3D::Manipulators.

KOMPAS v19

IManipulators – свойства

Item – Возвращает манипулятор, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IBaseManipulator

Синтаксис Automation:

```
Item = Object.Item( Index )          Получить свойство(* )
Item = Object.GetItem( Index )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Item( Index, &Item )      Получить свойство
```

Входные параметры:

```
Index          - Идентификатор манипулятора.
```

Примечание:

Свойство доступно только для чтения.

Manipulator – Возвращает манипулятор с заданным идентификатором

Интерфейс...

Тип данных: Указатель на интерфейс IBaseManipulator

Синтаксис Automation:

```
Manipulator = Object.Manipulator( Id )       Получить свойство(* )
Manipulator = Object.GetManipulator( Id )    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Manipulator( Id, &Manipulator )   Получить свойство
```

Входные параметры:

```
Id          - Идентификатор манипулятора.
```

Примечание:

Свойство доступно только для чтения.

IManipulators- методы

Add – Создать манипулятор (добавляет контрол в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksManipulatorTypeEnum Type);

Синтаксис COM :

HRESULT Add(ksManipulatorTypeEnum Type, IBaseManipulator * * Result);

Возвращаемое значение:

- Указатель на интерфейс IBaseManipulator .

Входные параметры:

Type - Способ разбиения зоны из перечисления ksManipulatorTypeEnum.

Примечания:

Метод позволяет создать новый базовый интерфейс манипулятора.

Интерфейс IBaseManipulator

Базовый интерфейс манипулятора.

Иерархия:

IDispatch

IBaseManipulator

Примечание:

Интерфейс можно получить с помощью метода коллекции манипуляторов IManipulators::Add

KOMPAS v19

IBaseManipulator – свойства

Active – Активный манипулятор

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Active = Object.Active	Получить свойство(*)
Object.Active = Active	Установить свойство (*)
Active = Object.GetActive()	Получить свойство (**)
Object.SetActive(Active)	Установить свойство (**)

Синтаксис COM:

Object.get_Active(&Active)	Получить свойство
Object.put_Active(Active)	Установить свойство

Id – Идентификатор манипулятора

Интерфейс...

Тип данных: long

Синтаксис Automation:

Id = Object.Id	Получить свойство(*)
Object.Id = Id	Установить свойство (*)
Id = Object.GetId()	Получить свойство (**)
Object.SetId(Id)	Установить свойство (**)

Синтаксис COM:

Object.get_Id(&Id)	Получить свойство
Object.put_Id(Id)	Установить свойство

Примечание:

Для корректной рассылки событий манипуляторов каждому манипулятору требуется задать уникальный ненулевой номер.

ManipulatorType – Тип манипулятора

Интерфейс...

Тип данных: из перечисления ksManipulatorTypeEnum

Синтаксис Automation:

ManipulatorType = Object.ManipulatorType	Получить свойство(*)
ManipulatorType = Object.GetManipulatorType()	Получить свойство (**)

Синтаксис COM:

Object.get_ManipulatorType(&ManipulatorType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Placement – Получить локальную систему координат

Интерфейс...

Тип данных: Указатель на интерфейс IPlacement3D

Синтаксис Automation:

Placement = Object.Placement	Получить свойство(*)
Placement = Object.GetPlacement()	Получить свойство (**)

Синтаксис COM:

Object.get_Placement(&Placement)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Visible – Свойство видимости манипулятора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = Object.Visible	Получить свойство(*)
Object.Visible = Visible	Установить свойство (*)
Visible = Object.GetVisible()	Получить свойство (**)
Object.SetVisible(Visible)	Установить свойство (**)

Синтаксис COM:

Object.get_Visible(&Visible)	Получить свойство
Object.put_Visible(Visible)	Установить свойство

IBaseManipulator – методы

Create – Создать манипулятор

Интерфейс...

Синтаксис Automation:

BOOL Create();

Синтаксис COM:

HRESULT Create(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Delete – Удалить манипулятор

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

UpdatePlacement – Установить локальную систему координат

Интерфейс...

Синтаксис Automation:

BOOL UpdatePlacement(BOOL Redraw);

Синтаксис COM:

HRESULT UpdatePlacement(BOOL Redraw, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Redraw	- признак перерисовки.

Интерфейс IEditDoubleManipulator

Интерфейс манипулятора – редактора вещественного значения.

Иерархия:

IDispatch

IBaseManipulator

IEditDoubleManipulator

Примечание

- Манипулятор позволяет вводить вещественное значение в 3D процессе.
- Интерфейс можно получить, добавив манипулятор через коллекцию манипуляторов IManipulators::Add.
- Интерфейс является дополнительным для IBaseManipulator.

Версия: KOMPAS v19

IEditDoubleManipulator – свойства

EditValue – Текущее значение в редакторе

Интерфейс...

Тип данных: double

Синтаксис Automation:

EditValue = Object.EditValue	Получить свойство (*)
Object.EditValue = EditValue	Установить свойство (*)
EditValue = Object.GetEditValue()	Получить свойство (**)
Object.SetEditValue(EditValue)	Установить свойство (**)

Синтаксис COM:

Object.get_EditValue(&EditValue)	Получить свойство
Object.put_EditValue(EditValue)	Установить свойство

IsEditCreated – Создан редактор с установленным фокусом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsEditCreated = Object.IsEditCreated	Получить свойство (*)
IsEditCreated = Object.GetIsEditCreated()	Получить свойство (**)

Синтаксис COM:

Object.get_IsEditCreated(&IsEditCreated)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEditDoubleManipulator – методы

SetValueRange – Установить новые ограничения на значение

Интерфейс...

Синтаксис Automation:

BOOL SetValueRange(double MinVal, double MaxVal);

Синтаксис COM :

HRESULT SetValueRange(double MinVal, double MaxVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

MinVal	- минимальное значение,
MaxVal	- максимальное значение.

Интерфейс IPlacement3DManipulator

Интерфейс манипулятора системы координат.

Иерархия:

IDispatch

IBaseManipulator

IPlacement3DManipulator

Примечание

Манипулятор позволяет указать положение при работе в 3D.

Интерфейс можно получить, добавив манипулятор через коллекцию манипуляторов IManipulators::Add.

Интерфейс является дополнительным для IBaseManipulator.

Версия: KOMPAS v19

IPlacement3DManipulator – свойства

EditValue – Текущее значение в редакторе

Интерфейс...

Тип данных: double

Синтаксис Automation:

EditValue = Object.EditValue	Получить свойство (*)
Object.EditValue = EditValue	Установить свойство (*)
EditValue = Object.GetEditValue()	Получить свойство (**)
Object.SetEditValue(EditValue)	Установить свойство (**)

Синтаксис COM:

Object.get_EditValue(&EditValue)	Получить свойство
Object.put_EditValue(EditValue)	Установить свойство

IsEditCreated – Создан редактор с установленным фокусом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsEditCreated = Object.IsEditCreated	Получить свойство (*)

IsEditCreated = Object.GetIsEditCreated() Получить свойство (**)

Синтаксис COM:

Object.get_IsEditCreated(&IsEditCreated) Получить свойство,

Примечание:

Свойство доступно только для чтения.

Mode – Режим работы манипулятора

Интерфейс...

Тип данных: long

Синтаксис Automation:

Mode = Object.Mode Получить свойство (*)
Object.Mode = Mode Установить свойство (*)
Mode = Object.GetMode() Получить свойство (**)
Object.SetMode(Mode) Установить свойство (**)

Синтаксис COM:

Object.get_Mode(&Mode) Получить свойство
Object.put_Mode(Mode) Установить свойство

PrimitiveDisabled – Запрет изменения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PrimitiveDisabled = Object.PrimitiveDisabled(Primitive) Получить свойство (*)
Object.PrimitiveDisabled(Primitive) = PrimitiveDisabled Установить свойство (*)
PrimitiveDisabled = Object.GetPrimitiveDisabled(Primitive) Получить свойство (**)
Object.SetPrimitiveDisabled(Primitive, PrimitiveDisabled) Установить свойство (**)

Синтаксис COM:

Object.get_PrimitiveDisabled(Primitive, &PrimitiveDisabled) Получить свойство
Object.put_PrimitiveDisabled(Primitive, PrimitiveDisabled) Установить свойство

Входные параметры:

ksManipulatorPrimitiveEnum Primitive - тип примитива манипулятора.

PrimitiveSelected – Селектированный элемент манипулятора

Интерфейс...

Тип данных: из перечисления ksManipulatorPrimitiveEnum

Синтаксис Automation:

PrimitiveSelected = Object.PrimitiveSelected	Получить свойство (*)
Object.PrimitiveSelected = PrimitiveSelected	Установить свойство (*)
PrimitiveSelected = Object.GetPrimitiveSelected	Получить свойство (**)
Object.SetPrimitiveSelected(PrimitiveSelected)	Установить свойство (**)

Синтаксис COM:

Object.get_PrimitiveSelected(&PrimitiveSelected)	Получить свойство
Object.put_PrimitiveSelected(PrimitiveSelected)	Установить свойство

PrimitiveVisible – Свойство видимости элементов манипулятора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PrimitiveVisible = Object.PrimitiveVisible(Primitive)	Получить свойство (*)
Object.PrimitiveVisible(Primitive) = PrimitiveVisible	Установить свойство (*)
PrimitiveVisible = Object.GetPrimitiveVisible(Primitive)	Получить свойство (**)
Object.SetPrimitiveVisible(Primitive, PrimitiveVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_PrimitiveVisible(Primitive, &PrimitiveVisible)	Получить свойство
Object.put_PrimitiveVisible(Primitive, PrimitiveVisible)	Установить свойство

Входные параметры:

ksManipulatorPrimitiveEnum	Primitive	- тип примитива манипулятора.

ReadOnly – Запрет изменений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)

ReadOnly = Object.GetReadOnly()	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

RotateStep – Шаг поворота манипулятора вокруг оси

Интерфейс...

Тип данных: double

Синтаксис Automation:

RotateStep = Object.RotateStep	Получить свойство (*)
Object.RotateStep = RotateStep	Установить свойство (*)
RotateStep = Object.GetRotateStep()	Получить свойство (**)
Object.SetRotateStep(RotateStep)	Установить свойство (**)

Синтаксис COM:

Object.get_RotateStep(&RotateStep)	Получить свойство
Object.put_RotateStep(RotateStep)	Установить свойство

ShiftStep – Шаг сдвига манипулятора

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShiftStep = Object.ShiftStep	Получить свойство (*)
Object.ShiftStep = ShiftStep	Установить свойство (*)
ShiftStep = Object.GetShiftStep()	Получить свойство (**)
Object.SetShiftStep(ShiftStep)	Установить свойство (**)

Синтаксис COM:

Object.get_ShiftStep(&ShiftStep)	Получить свойство
Object.put_ShiftStep(ShiftStep)	Установить свойство

IPlacement3DManipulator– методы

SetShiftRange – Установить диапазон для сдвига манипулятора

Интерфейс...

Синтаксис Automation:

BOOL SetShiftRange(double Min, double Max);

Синтаксис COM :

HRESULT SetShiftRange(double Min, double Max, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Min	- минимальное значение.

Выходные параметры:

Max	- максимальное значение.

SetRotateRange – Установить диапазон для поворота манипулятора

Интерфейс...

Синтаксис Automation:

BOOL SetRotateRange(double Min, double Max);

Синтаксис COM:

HRESULT SetRotateRange(double Min, double Max, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Min	- минимальное значение.

Выходные параметры:

Max	- максимальное значение.

Интерфейс IMouseEnterLeaveParameters7

Параметры отображения точки, позволяющей определить место применения контрола.

Иерархия:

IDispatch

IKompasAPIObject

IMouseEnterLeaveParameters7

Примечание:

Интерфейс параметров используется в событии ksProcess2DNotify::GetMouseEnterLeavePoint.

Версия: KOMPAS v19

IMouseEnterLeaveParameters7 – свойства

Offset – Смещение символа относительно точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Offset = Object.Offset	Получить свойство (*)
Object.Offset = Offset	Установить свойство (*)
Offset = Object.GetOffset()	Получить свойство (**)
Object.SetOffset(Offset)	Установить свойство (**)

Синтаксис COM:

Object.get_Offset(&Offset)	Получить свойство
Object.put_Offset(Offset)	Установить свойство

OffsetAngle – Направление смещения символа относительно точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

OffsetAngle = Object.OffsetAngle	Получить свойство (*)
Object.OffsetAngle = OffsetAngle	Установить свойство (*)
OffsetAngle = Object.GetOffsetAngle()	Получить свойство (**)
Object.SetOffsetAngle(OffsetAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetAngle(&OffsetAngle)	Получить свойство
Object.put_OffsetAngle(OffsetAngle)	Установить свойство

Symbol – Символ, отображаемый в поле документа, при наведении на контрол

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Symbol = Object.Symbol                  Получить свойство (* )
Object.Symbol = Symbol                  Установить свойство (* )
Symbol = Object.GetSymbol()             Получить свойство (**)
Object.SetSymbol(Symbol)                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Symbol( &Symbol )            Получить свойство
Object.put_Symbol(Symbol)               Установить свойство
```

SymbolColor – Цвет символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
SymbolColor   = Object.SymbolColor      Получить свойство (* )
Object.SymbolColor   = SymbolColor      Установить свойство (* )
SymbolColor   = Object.GetSymbolColor ()  Получить свойство (**)
Object.SetSymbolColor  (SymbolColor  )  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SymbolColor  ( &SymbolColor   )  Получить свойство
Object.put_SymbolColor  (SymbolColor )      Установить свойство
```

SymbolFont – Шрифт символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
SymbolFont  = Object.SymbolFont         Получить свойство (* )
Object.SymbolFont  = SymbolFont         Установить свойство (* )
SymbolFont  = Object.GetSymbolFont ()   Получить свойство (**)
Object.SetSymbolFont (SymbolFont )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SymbolFont ( &SymbolFont  )  Получить свойство
Object.put_SymbolFont (SymbolFont )     Установить свойство
```

SymbolScale – Увеличение высоты символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: double

SymbolScale = Object.SymbolScale	Получить свойство (*)
Object.SymbolScale = SymbolScale	Установить свойство (*)
SymbolScale = Object.GetSymbolScale ()	Получить свойство (**)
Object.SetSymbolScale (SymbolScale)	Установить свойство (**)

Синтаксис COM:

Object.get_SymbolScale (&SymbolScale)	Получить свойство
Object.put_SymbolScale (SymbolScale)	Установить свойство

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство (*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство (*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Интерфейс ISaveToPreviusParam7

Параметры конвертации при сохранении в предыдущую версию.

Иерархия:

IDispatch

ISaveToPreviusParam7

Версия: KOMPAS v19

ISaveToPreviusParam7- методы

AddOption – Добавить настройку конвертации с возможностью выбора варианта конвертации

Интерфейс...

Синтаксис Automation:

BOOL AddOption(BSTR UniqueId, BSTR OptionName, VARIANT Options, BSTR DefaultValue);

Синтаксис COM :

HRESULT AddOption(BSTR UniqueId, BSTR OptionName, VARIANT Options, BSTR DefaultValue, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

UniqueId	- уникальный идентификатор,
OptionName	- отображаемое имя свойства,
Options	- список возможных значений (массив строк SafeArray VT_ARRAY | VT_BSTR),
DefaultValue	- базовое значение.

AddWarning – Добавить предупреждение

Интерфейс...

Синтаксис Automation:

BOOL AddWarning(BSTR UniqueId, BSTR OptionName, BSTR Text);

Синтаксис COM

HRESULT AddWarning(BSTR UniqueId, BSTR OptionName, BSTR Text, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

UniqueId - уникальный идентификатор,
OptionName - отображаемое имя свойства,
Text - текст предупреждения.

GetCurrentOptionValue – Получить текущее значение, выбранное в диалоге параметров конвертации

Интерфейс...

Синтаксис Automation:

BSTR GetCurrentOptionValue(BSTR UniqueId);

Синтаксис COM:

HRESULT GetCurrentOptionValue(BSTR UniqueId, BSTR * Result);

Возвращаемое значение:

- Текущее значение свойства.

Входные параметры:

UniqueId - уникальный идентификатор.

Работа с панелью свойств

Интерфейс IPropertyControl

Справка системы КОМПАС...

`kompas.chm::/33_1_1_Panelq_svojstv.htm`

Базовый интерфейс элемента управления Панели свойств.

Иерархия:

IKompasAPIObject
IPropertyControl
IPropertyControl1

Описание:

При изменении библиотечных элементов управления система передает управление библиотеке посредством вызовов соответствующих событий. Данный интерфейс является базовым для всех элементов управления Панели свойств. Все элементы управления из ControlTypeEnum являются наследниками от IPropertyControl.

Примечание:

1. Имя выводится рядом с элементом управления (в зависимости от настроек Панели свойств). Если имя не задано или задана пустая строка, имя выводится не будет.
2. Подсказки для элемента управления устанавливаются функциями Hint и Tips.
3. Идентификатор элемента управления, установленный методом Id, будет приходить в событиях от данного элемента управления как параметр.
4. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add.
5. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить дополнительный интерфейс IPropertyControl1.

IPropertyControl – свойства

ControlType – Тип элемента управления

Интерфейс...

Тип данных: тип элемента управления из перечисления ControlTypeEnum.

Синтаксис Automation:

```
ControlType = iObject.ControlType          Получить свойство (* )
ControlType = iObject.GetControlType()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_ControlType (&ControlType)    Получить свойство
```

Примечание:

Свойство только для чтения.

Enable – Состояние (доступность) элемента управления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Enable = iObject.Enable                    Получить свойство (* )
iObject.Enable = Enable                    Установить свойство (* )
Enable = iObject.GetEnable()               Получить свойство(**)
iObject.SetEnable (Enable)                 Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Enable (&Enable)              Получить свойство
iObject->put_Enable (Enable)               Установить свойство
```

Примечание:

Позволяет получить и установить доступность элемента управления.

Hint – Подсказки элемента управления

Функция не поддерживается

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Hint = iObject.Hint                 Получить свойство (* )
iObject.Hint = Hint                 Установить свойство (* )
Hint = iObject.GetHint()            Получить свойство(**)
iObject.SetHint (Hint)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Hint (&Hint)           Получить свойство
iObject->put_Hint (Hint)            Установить свойство
```

Примечание:

Позволяет получить и установить подсказку элемента управления.

Id – Идентификатор элемента управления

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Id = iObject.Id                     Получить свойство (* )
iObject.Id = Id                     Установить свойство (* )
Id = iObject.GetId()                Получить свойство(**)
iObject.SetId (Id)                  Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Id (&Id)               Получить свойство
iObject->put_Id (Id)                Установить свойство
```

Примечание:

Позволяет получить и установить идентификатор элемента управления.

Name – Имя элемента управления

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Name = iObject.Name                    Получить свойство (* )
iObject.Name = Name                    Установить свойство (* )
Name = iObject.GetName()               Получить свойство(**)
iObject.SetName (Name)                 Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name (&Name)              Получить свойство
iObject->put_Name (Name)               Установить свойство
```

Примечание:

Позволяет получить и установить имя элемента управления.

NameVisibility – Видимость имени на Панели свойств

Интерфейс...

Тип данных: PropertyControlNameVisibility

Синтаксис Automation:

```
NameVisibility = iObject.NameVisibility          Получить свойство (* )
iObject.NameVisibility = NameVisibility          Установить свойство (* )
NameVisibility = iObject.GetNameVisibility()     Получить свойство(**)
iObject.SetNameVisibility (NameVisibility)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_NameVisibility (&NameVisibility)    Получить свойство
iObject->put_NameVisibility (NameVisibility)     Установить свойство
```

Примечание:

Позволяет получить и установить видимость имени элемента управления на Панели свойств.

Tips – Подсказки элемента управления

Функция не поддерживается

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Tips = iObject.Tips                    Получить свойство (* )
iObject.Tips = Tips                    Установить свойство (* )
Tips = iObject.GetTips()               Получить свойство(**)
iObject.SetTips (Tips)                 Установить свойство (**)
```

Синтаксис COM:

iObject->get_Tips (&Tips)	Получить свойство
iObject->put_Tips (Tips)	Установить свойство

Примечание:

Позволяет получить и установить подсказку элемента управления.

Value – Значение элемента управления

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Value = iObject.Value	Получить свойство (*)
iObject.Value = Value	Установить свойство (*)
Value = iObject.GetValue()	Получить свойство(**)
iObject.SetValue (Value)	Установить свойство (**)

Синтаксис COM:

iObject->get_Value (&Value)	Получить свойство
iObject->put_Value (Value)	Установить свойство

Примечание:

1. Позволяет получить и установить значение элемента управления.
2. Тип данных определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlEditStr ограничения на значение не накладываются,
- для типа ksControlEditAngle используется тип double,
- для типа ksControlEditLength используется тип double,
- для типа ksControlEditPoint используется тип VT_ARRAY | VT_R8 - на 2 значения координат.

Visible – Видимость элемента управления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство(**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

Позволяет получить и установить свойство видимости элемента управления.

Интерфейс IPropertyBasePoint

Интерфейс Элемент панели свойств – Базовая точка

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyBasePoint

1. Интерфейс позволяет добавить элемент панели свойств - Базовая точка.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyBasePoint – свойства

CenterPointVisible – Отображать центральную точку

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CenterPointVisible = Object.CenterPointVisible	Получить свойство (*)
Object.CenterPointVisible = CenterPointVisible	Установить свойство (*)
CenterPointVisible = Object.GetCenterPointVisible()	Получить свойство(**)
Object.SetCenterPointVisible(CenterPointVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_CenterPointVisible(&CenterPointVisible)	Получить свойство
Object.put_CenterPointVisible(CenterPointVisible)	Установить свойство

Интерфейс IPropertyMarking

Интерфейс элемента панели свойств – Обозначение.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyMarking

1. Интерфейс позволяет добавить элемент панели свойств - Обозначение.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyMarking – свойства

MarkingVisible – Признак отображение части обозначения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MarkingVisible = Object.MarkingVisible(MarkingType)	Получить свойство (*)
Object.MarkingVisible(MarkingType) = MarkingVisible	Установить свойство (*)
MarkingVisible = Object.GetMarkingVisible(MarkingType)	Получить свойство(**)
Object.SetMarkingVisible(MarkingType, MarkingVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_MarkingVisible(MarkingType, &MarkingVisible)	Получить свойство
Object.put_MarkingVisible(MarkingType, MarkingVisible)	Установить свойство

Входные параметры:

MarkingType -- индекс части обозначения из перечисления ksVariantMarkingTypeEnum.

ReadOnly – Запрет редактирования значения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = Object.GetReadOnly()	Получить свойство(**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly) Получить свойство
Object.put_ReadOnly(ReadOnly) Установить свойство

IPropertyMarking – методы

GetMarking – Обозначение исполнения

Интерфейс...

Синтаксис Automation:

BSTR GetMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer);

Синтаксис COM:

HRESULT GetMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer, BSTR * Result);

Входные параметры:

MarkingType - индекс части обозначения из перечисления ksVariantMarkingTypeEnum,
AddSystemDelimer - добавлять разделители.

SetMarking – Обозначение исполнения

Интерфейс...

Синтаксис Automation:

BOOL SetMarking(ksVariantMarkingTypeEnum MarkingType, BSTR Marking);

Синтаксис COM:

HRESULT SetMarking(ksVariantMarkingTypeEnum MarkingType, BSTR Marking, BOOL * Result);

Входные параметры:

MarkingType -- индекс части обозначения из перечисления ksVariantMarkingTypeEnum,
Marking - текст обозначения.

Примечание:

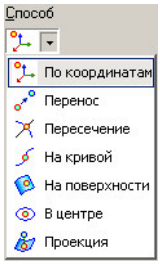
При передаче полного обозначения части обозначения нужно разделить разделителями |$

Интерфейс IPropertyBmpList

Элемент панели свойств Раскрывающийся список графических объектов.

Иерархия:

IDispatch
IKompasAPIObject
IPropertyControl
IPropertyBmpList



Элемент панели свойств

Примечание:

Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyBmpList – свойства

Count – Количество элементов списка

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Count = Object.GetCount()	Получить свойство(**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство

Примечание:

1. Свойство позволяет получить количество элементов списка.
2. Свойство доступно только для чтения.

CurrentIndex – Индекс текущего значения

Интерфейс...

Тип данных: long.

Синтаксис Automation:

CurrentIndex = Object.CurrentIndex	Получить свойство (*)
Object.CurrentIndex = CurrentIndex	Установить свойство (*)
CurrentIndex = Object.GetCurrentIndex( )	Получить свойство(**)2
Object.SetCurrentIndex(CurrentIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentIndex(&CurrentIndex)	Получить свойство
Object.put_CurrentIndex(CurrentIndex)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать индекс текущего элемента.

IconFont – Шрифт иконок списка

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

IconFont = Object.IconFont(Index)	Получить свойство (*)
Object.IconFont(Index) = IconFont	Установить свойство (*)
IconFont = Object.GetIconFont(Index)	Получить свойство(**)2
Object.SetIconFont(Index, IconFont)	Установить свойство (**)

Синтаксис COM:

Object.get_IconFont(Index, &IconFont)	Получить свойство
Object.put_IconFont(Index, IconFont)	Установить свойство

Входные параметры:

Index	- индекс значения в списке.

ResModule – Модуль с ресурсом растрового изображения

Интерфейс...

Тип данных: VARIANT.

Синтаксис Automation:

ResModule = Object.ResModule	Получить свойство (*)
Object.ResModule = ResModule	Установить свойство (*)
ResModule = Object.GetResModule( )	Получить свойство(**)
Object.SetResModule(ResModule)	Установить свойство (**)

Синтаксис COM:

Object.get_ResModule(&ResModule)	Получить свойство
Object.put_ResModule(ResModule)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать модуль с ресурсом битмапа.

IPropertyBmpList – методы

Add– Добавить значение в список

Интерфейс...

Синтаксис Automation:

void Add(BSTR NewStr, VARIANT NewBmp);

Синтаксис COM:

HRESULT Add(BSTR NewStr, VARIANT NewBmp);

Входные параметры:

NewStr - добавляемое значение,
NewBmp - идентификатор значка в файле ресурсов (тип данных long) или полный путь к bmp-файлу значка (тип данных BSTR).

ClearList – Очистить список значений

Интерфейс...

Синтаксис Automation:

void ClearList();

Синтаксис COM:

HRESULT ClearList();

Find – Найти индекс в списке по значению

Интерфейс...

Синтаксис Automation:

long Find(BSTR val);

Синтаксис COM:

HRESULT Find(BSTR Val, long * PVal);

Входные параметры:

Val - значение в раскрывающемся списке.

Возвращаемое значение:

Индекс строки со значением Val в списке - в случае успеха,
-1 - в случае неудачи.

Интерфейс IPropertyCheckBox

Интерфейс элемента управления Панели свойств Опция.

Иерархия:

IKompasAPIObject

Элемент панели свойств

IPropertyControl

IPropertyCheckBox

Примечание:

1. Вместо стандартного вида элемента управления можно задать значки для его различных состояний.
2. Данный интерфейс можно получить через интерфейс коллекции элементов управления панели свойств IPropertyControls.
3. При активизации элемента управления вызывается событие ksIPropertyManagerNotify::ChangeControlValue.

IPropertyCheckBox – свойства

VisualStyle – Визуальный стиль

Интерфейс...

Тип данных: из перечисления ksCheckBoxVisualStyleEnum

Синтаксис Automation:

VisualStyle = Object.VisualStyle	Получить свойство (*)
Object.VisualStyle = VisualStyle	Установить свойство (*)
VisualStyle = Object.GetVisualStyle()	Получить свойство (**)
Object.SetVisualStyle(VisualStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_VisualStyle(&VisualStyle)	Получить свойство
Object.put_VisualStyle(VisualStyle)	Установить свойство

Примечание

Свойство задает визуальный стиль отображения чекбокса:

- ksCheckBoxDefault - обычный чекбокс,
- ksCheckBoxSwitcher - переключатель.

IPropertyCheckBox – методы

SetCustomBitmaps – Задать значки для различных состояний элемента управления

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

void SetCustomBitmaps (const VARIANT& idUnchecked,
const VARIANT& idChecked,
const VARIANT& idIndeterminate,
const VARIANT& hInstance);

Синтаксис COM:

HRESULT SetCustomBitmaps ([in]VARIANT idUnchecked,
[in]VARIANT idChecked,
[in]VARIANT idIndeterminate,
[in]VARIANT hInstance);

Входные параметры:

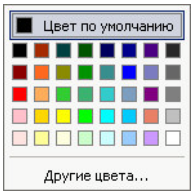
idUnchecked	- значок ненажатого состояния,
idChecked	- битмап нажатого состояния,
idIndeterminate	- битмап неопределенного состояния,
hInstance	- dll-модуль с ресурсами значков.

Примечание:

1. Значки могут быть заданы в виде идентификаторов в ресурсе dll-модуля (тип данных long) или в виде полного пути к bmp-файлу значка (тип данных BSTR). Обязательным является задание значка для ненажатого состояния idUnchecked, остальные могут быть не заданы.
2. Все значения переменных должны быть одного типа (или long или BSTR).
3. Если значки задаются в виде идентификатора, то обязательно должен быть задан dll-модуль, в ресурсах которого лежат эти значки. Модуль с ресурсами значков hInstance может быть задан в виде HINSTANCE (тип значения long) или полного пути к файлу (тип значения BSTR) dll-модуля.

Интерфейс IPropertyColor

Интерфейс элемента управления Панели свойств – Выбор цвета.



Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyColor

Примечание:

1. Данный интерфейс можно получить, используя коллекцию элементов управления Панели свойств IPropertyControls.
2. При изменении значения параметра вызывается событие ksPropertyManagerNotify::ChangeControlValue.

IPropertyColor – свойства

EnableDefaultButton – Отображение кнопки «Цвет по умолчанию»

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

```
EnableDefaultButton = iObject.EnableDefaultButton          Получить свойство (* )
iObject.EnableDefaultButton = EnableDefaultButton          Установить свойство (* )
EnableDefaultButton = iObject.GetEnableDefaultButton()     Получить свойство(**)
iObject.SetEnableDefaultButton (EnableDefaultButton)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_EnableDefaultButton (&EnableDefaultButton)    Получить свойство
iObject->put_EnableDefaultButton (EnableDefaultButton)     Установить свойство
```

Значения свойства:

TRUE - вверху раскрывающегося списка будет отображаться дополнительная кнопка **Цвет по умолчанию**,
FALSE - кнопка **Цвет по умолчанию** не будет отображаться.

DefaultButtonName – Имя кнопки

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
DefaultButtonName = iObject.DefaultButtonName              Получить свойство (* )
iObject.DefaultButtonName = DefaultButtonName              Установить свойство (* )
iObject.GetDefaultButtonName()                             Получить свойство(**)
iObject.SetDefaultButtonName (DefaultButtonName)           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DefaultButtonName (&DefaultButtonName)        Получить свойство
iObject->put_DefaultButtonName (DefaultButtonName)         Установить свойство
```

Примечание:

1. Свойство позволяет получить имя кнопки **Цвет по умолчанию** или изменить имя на произвольное.
2. Свойство доступно только если свойство EnableDefaultButton имеет значение TRUE.

DefaultColor – Цвет по умолчанию

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
DefaultColor = iObject.DefaultColor          Получить свойство (* )
iObject.DefaultColor = DefaultColor          Установить свойство (* )
DefaultColor = iObject.GetDefaultColor()     Получить свойство(**)
iObject.SetDefaultColor (DefaultColor)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DefaultColor (&DefaultColor)    Получить свойство
iObject->put_DefaultColor (DefaultColor)     Установить свойство
```

Примечание:

1. Свойство позволяет получить или изменить значение цвета для кнопки **Цвет по умолчанию**.
2. Цвет по умолчанию доступен только если свойство EnableDefaultButton имеет значение TRUE.

Width – Шаблон для расчета ширины элемента управления в горизонтальном положении (в символах)

Интерфейс...

Тип данных:long.

Синтаксис Automation:

```
Width = iObject.Width                        Получить свойство (* )
iObject.Width = Width                        Установить свойство (* )
Width = iObject.GetWidth()                   Получить свойство(**)
iObject.SetWidth (Width)                     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Width (&Width)                  Получить свойство
iObject->put_Width (Width)                   Установить свойство
```

Значения свойства:

Количество видимых символов в текстовом поле при горизонтальном положении Панели свойств.

Примечание:

Для расчета ширины элемента управления в горизонтальном положении используется символ «0».

Интерфейс IPropertyGroupBegin

Элемент панели свойств – Начало группы контролов.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyGroupBegin

Примечание:

1. Данный интерфейс позволяет объединить несколько контролов панели свойств в группу. Контролы, добавленные после IPropertyGroupBegin, будут объединены в группу.
2. Для завершения группы нужно добавить контрол IPropertyGroupEnd.
3. Группа иметь возможность сворачивать и прятать контролы, входящие в нее, если у группы признак IPropertyGroupBegin::Expanding выставлен равным true.
4. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyGroupBegin – свойства

Expanding – Доступность сворачивания

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Expanding = Object.Expanding	Получить свойство (*)
Object.Expanding = Expanding	Установить свойство (*)
Expanding = Object.GetExpanding()	Получить свойство(**)
Object.SetExpanding(Expanding)	Установить свойство (**)

Синтаксис COM:

Object.get_Expanding(&Expanding	Получить свойство
Object.put_Expanding(Expanding)	Установить свойство

Expanded – Текущее состояние. TRUE – развернута. FALSE – свернута

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Expanded = Object.Expanded	Получить свойство (*)
Object.Expanded = Expanded	Установить свойство (*)
Expanded = Object.GetExpanded()	Получить свойство(**)
Object.SetExpanded(Expanded)	Установить свойство (**)

Синтаксис COM:

Object.get_Expanded(&Expanded)	Получить свойство
Object.put_Expanded(Expanded)	Установить свойство

Интерфейс IPropertyGroupEnd

Интерфейс элемента панели свойств – Конец группы контролов.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyGroupEnd

Примечание:

1. Данный интерфейс позволяет завершить объединение несколько контролов панели свойств в группу. Контролы, добавленные после IPropertyGroupBegin, будут объединены в группу.
2. Для завершения группы нужно добавить контрол IPropertyGroupEnd.
3. Группа иметь возможность сворачивать и прятать контролы, входящие в нее, если у группы признак IPropertyGroupBegin::Expanding выставлен равным true.
4. Данный интерфейс может быть получен от интерфейса коллекции элементов управления *IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

Интерфейс IPropertyLinkButton

Интерфейс элемента Панели свойств – Набор кнопок в виде ссылок.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyLinkButton

1. Интерфейс позволяет добавить элемент панели свойств - Набор кнопок в виде ссылок.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления *IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyLinkButton – свойства

ButtonChecked – Нажатие кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ButtonChecked = Object.ButtonChecked( BtnID )          Получить свойство (* )
Object.ButtonChecked( BtnID ) = ButtonChecked          Установить свойство (* )
ButtonChecked = Object.GetButtonChecked( BtnID )       Получить свойство(**)
Object.SetButtonChecked( BtnID, ButtonChecked )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ButtonChecked( BtnID, &ButtonChecked )      Получить свойство
Object.put_ButtonChecked(  BtnID, ButtonChecked )      Установить свойство
```

Входные параметры:

```
long   BtnID                  - идентификатор кнопки
```

ButtonEnable – Доступность кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ButtonEnable = Object.ButtonEnable( BtnID )            Получить свойство (* )
Object.ButtonEnable( BtnID ) = ButtonEnable            Установить свойство (* )
ButtonEnable = Object.GetButtonEnable( BtnID )         Получить свойство(**)
Object.SetButtonEnable( BtnID, ButtonEnable )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ButtonEnable( BtnID, &ButtonEnable )        Получить свойство
Object.put_ButtonEnable(  BtnID, ButtonEnable )        Установить свойство
```

Входные параметры:

long BtnID - идентификатор кнопки.

ButtonVisible – Отображение кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ButtonVisible = Object.ButtonVisible(BtnID)	Получить свойство (*)
Object.ButtonVisible(BtnID) = ButtonVisible	Установить свойство (*)
ButtonVisible = Object.GetButtonVisible(BtnID)	Получить свойство(**)
Object.SetButtonVisible(BtnID, ButtonVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_ButtonVisible(BtnID, &ButtonVisible)	Получить свойство
Object.put_ButtonVisible(BtnID, ButtonVisible)	Установить свойство

Входные параметры:

long BtnID - идентификатор кнопки.

IPropertyLinkButton – методы

AddButton – Добавить кнопку

Интерфейс...

Синтаксис Automation:

BOOL AddButton(long BtnID, BSTR Lebel);

Синтаксис COM:

HRESULT AddButton(long BtnID, BSTR Lebel, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BtnID	- идентификатор кнопки,
Lebel	- заголовок кнопки.

Интерфейс IPropertyTwinSwitcher

Интерфейс элемента панели свойств – переключателя.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyTwinSwitcher

Примечание:

1. Интерфейс позволяет добавить элемент панели свойств - переключатель.

   Для обозначения состояния контрола имеются две текстовых метки.

   Первая метка задается свойством IPropertyControl::Name.

   Вторая метка задается свойством IPropertyTwinSwitcher::Label2.

2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyTwinSwitcher – свойства

Label2 – Текст для второго состояния

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Label2 = Object.Label2	Получить свойство (*)
Object.Label2 = Label2	Установить свойство (*)
Label2 = Object.GetLabel2()	Получить свойство(**)
Object.SetLabel2(Label2)	Установить свойство (**)

Синтаксис COM:

Object.get_Label2(&Label2)	Получить свойство
Object.put_Label2(Label2)	Установить свойство

Интерфейс IPropertyEdit

Интерфейс элемента управления Панели свойств типа Поле ввода.

Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyEdit

Примечание:

1. Данный интерфейс позволяет получить и установить параметры поля ввода.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyEdit – свойства

CheckState – Состояние опции

Интерфейс...

Тип данных: состояние опции из перечисления CheckStateEnum.

Синтаксис Automation:

```
CheckState = iObject.CheckState          Получить свойство (* )
iObject.CheckState = CheckState          Установить свойство (* )
CheckState = iObject.GetCheckState()     Получить свойство(**)
iObject.SetCheckState (CheckState)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CheckState (&CheckState)    Получить свойство
iObject->put_CheckState (CheckState)     Установить свойство
```

Примечание:

Это свойство влияет на состояние переключателя, если свойство WithCheck имеет значение TRUE, и на отображение текста в поле элемента управления, подробнее см. CheckStateEnum.

Expanded – Шаг

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step = Object.Step                       Получить свойство (* )
Object.Step = Step                       Установить свойство (* )
Step = Object.GetStep()                  Получить свойство(**)
Object.SetStep( Step )                   Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step( &Step )                 Получить свойство
Object.put_Step(  Step )                 Установить свойство
```

Шаг изменения значения колесом мыши.

MaxValue – Максимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
MaxValue = iObject.MaxValue          Получить свойство (* )
MaxValue = iObject.GetMaxValue()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_MaxValue (&MaxValue)    Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlEditStr ограничения на значение не накладываются,
- для типа ksControlEditAngle используется тип double,
- для типа ksControlEditLength используется тип double,
- для типа ksControlEditPoint используется тип VT_ARRAY | VT_R8 - на 2 значения координат.

MinValue – Минимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
MinValue = iObject.MinValue          Получить свойство (* )
MinValue = iObject.GetMinValue()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_MinValue (&MinValue)    Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlEditStr ограничения на значение не накладываются,
- для типа ksControlEditAngle используется тип double,
- для типа ksControlEditLength используется тип double,
- для типа ksControlEditPoint используется тип VT_ARRAY | VT_R8 - на 2 значения координат.

ReadOnly – Запрет редактирования значения

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

ReadOnly = iObject.ReadOnly	Получить свойство (*)
iObject.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = iObject.GetReadOnly()	Получить свойство(**)
iObject.SetReadOnly (ReadOnly)	Установить свойство (**)

Синтаксис COM:

iObject->get_ReadOnly (&ReadOnly)	Получить свойство
iObject->put_ReadOnly (ReadOnly)	Установить свойство

Значение свойства:

TRUE	- ввод нового значения с клавиатуры в поле запрещен,
FALSE	- ввод нового значения с клавиатуры в раскрывающийся список разрешен.

Width – Шаблон для расчета ширины элемента управления в горизонтальном положении (в символах)

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство

iObject->put_Width (Width) Установить свойство

Значение свойства:

Количество видимых символов в поле при горизонтальном положении Панели свойств.

Примечание:

Для расчета ширины элемента управления используется символ «0».

WithCheck – Наличие опции рядом с полем ввода

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithCheck = iObject.WithCheck Получить свойство (*)
iObject.WithCheck = WithCheck Установить свойство (*)
WithCheck = iObject.GetWithCheck() Получить свойство(**)
iObject.SetWithCheck (WithCheck) Установить свойство (**)

Синтаксис COM:

iObject->get_WithCheck (&WithCheck) Получить свойство
iObject->put_WithCheck (WithCheck) Установить свойство

Значение свойства:

TRUE - рядом с элементом управления выводится опция,
FALSE - рядом с элементом управления не выводится опция.

IPropertyEdit – методы

SetValueRange – Установить новые ограничения на значение

Интерфейс...

Синтаксис Automation:

BOOL SetValueRange (const VARIANT& minVal, const VARIANT& maxVal);

Синтаксис COM:

HRESULT SetValueRange ([in] VARIANT minVal,
[in] VARIANT maxVal,
[out, retval] VARIANT_BOOL * pVal);

Входные параметры:

minVal - минимальное значение,
maxVal - максимальное значение.

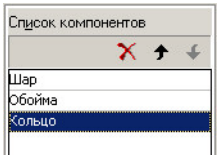
Примечание:

Тип данных максимального и минимального допустимого значения определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlEditStr ограничения на значение не накладываются,
- для типа ksControlEditAngle используется тип double,
- для типа ksControlEditLength используется тип double,
- для типа ksControlEditPoint используется тип VT_ARRAY | VT_R8 - на 2 значения координат.

Интерфейс IPropertyEditList

Интерфейс элемента управления Панели свойств Список.



Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyEditList

IPropertyToolBar

Описание:

Позволяет управлять списком из Панели свойств.

Примечание:

1. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс вида IPropertyToolBar.

IPropertyEditList – свойства

AllowDelete – Разрешить удалять элементы из списка

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AllowDelete = Object.AllowDelete                    Получить свойство (* )
Object.AllowDelete = AllowDelete                    Установить свойство (* )
AllowDelete = Object.GetAllowDelete()               Получить свойство (**)
Object.SetAllowDelete( AllowDelete )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AllowDelete( &AllowDelete )              Получить свойство
Object.put_AllowDelete(  AllowDelete )              Установить свойство
```

CheckState – Состояние контрола. Текущий, Не текущий

Интерфейс...

Тип данных: из перечисления CheckStateEnum

Синтаксис Automation:

```
CheckState = Object.CheckState                      Получить свойство (* )
Object.CheckState = CheckState                      Установить свойство (* )
CheckState = Object.GetCheckState()                 Получить свойство (**)
Object.SetCheckState( CheckState )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CheckState( &CheckState )                Получить свойство
Object.put_CheckState(  CheckState )                Установить свойство
```

ComputeUniqueNames – Генерация уникальных имен элементов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ComputeUniqueNames = Object.ComputeUniqueNames      Получить свойство (* )
Object.ComputeUniqueNames = ComputeUniqueNames      Установить свойство (* )
ComputeUniqueNames = Object.GetComputeUniqueNames() Получить свойство (**)
Object.SetComputeUniqueNames( ComputeUniqueNames )  Установить свойство (**)
```

Синтаксис COM:

Object.get_ComputeUniqueNames(&ComputeUniqueNames) Получить свойство
Object.put_ComputeUniqueNames(ComputeUniqueNames) Установить свойство

TRUE - автоматически нумеровать одноименные элементы в списке.

DefaultValue – Значение по умолчанию

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DefaultValue = Object.DefaultValue Получить свойство (*)
Object.DefaultValue = DefaultValue Установить свойство (*)
DefaultValue = Object.GetDefaultValue() Получить свойство (**)
Object.SetDefaultValue(DefaultValue) Установить свойство (**)

Синтаксис COM:

Object.get_DefaultValue(&DefaultValue) Получить свойство
Object.put_DefaultValue(DefaultValue) Установить свойство

Примечание

Свойство задает строку- приглашение, отображаемую при незаполненном списке значений.

ItemMissing – Потерянный объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ItemMissing = Object.ItemMissing(Index) Получить свойство (*)
Object.ItemMissing(Index) = ItemMissing Установить свойство (*)
ItemMissing = Object.GetItemMissing(Index) Получить свойство (**)
Object.SetItemMissing(Index, ItemMissing) Установить свойство (**)

Синтаксис COM:

Object.get_ItemMissing(Index, &ItemMissing) Получить свойство
Object.put_ItemMissing(Index, ItemMissing) Установить свойство

Входные параметры:

Index - индекс значения.

ItemChecked – Состояние элемента списка

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

```
BOOL ItemChecked = iObject.ItemChecked(index)          Получить свойство (* )
iObject.ItemChecked(index) = ItemChecked               Установить свойство (* )
BOOL ItemChecked = iObject.GetItemChecked(index)       Получить свойство(**)
iObject.SetItemChecked( index, ItemChecked )           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ItemChecked( index, &ItemChecked )    Получить свойство
iObject->put_ItemChecked( index,  ItemChecked )    Установить свойство
```

Входные параметры:

index - индекс элемента списка.

ItemsCount – Количество элементов в списке

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
long ItemsCount = iObject.ItemsCount          Получить свойство (* )
long ItemsCount = iObject.GetItemsCount()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_ItemsCount( &ItemsCount )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ItemSelected – Состояние выделенности элемента списка

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
long ItemSelected = iObject.ItemSelected      Получить свойство (* )
iObject.ItemSelected = ItemSelected           Установить свойство (* )
long ItemSelected =                           Получить свойство(**)
iObject.GetItemSelected()
```

iObject.SetItemSelected (ItemSelected)	Установить свойство (**)

Синтаксис COM:

iObject->get_ItemSelected (&ItemSelected)	Получить свойство
iObject->put_ItemSelected (ItemSelected)	Установить свойство

ItemValue – Значение элемента списка по индексу

Интерфейс...

Синтаксис Automation:

Тип данных: LPSTR.

ItemValue = iObject.ItemValue(Index)	Получить свойство (*)
iObject.ItemValue(Index) = ItemValue	Установить свойство (*)
ItemValue = iObject.GetItemValue(Index)	Получить свойство(**)
iObject.SetItemValue(Index)	Установить свойство (**)

Синтаксис COM:

Тип данных:BSTR.

iObject->get_Name (Index)	Получить свойство
iObject->put_Name (Index)	Установить свойство

Входные параметры:

Index	- индекс элемента.

ListType – Тип списка

Интерфейс...

Тип данных: из перечисления ksEditListTypeEnum.

Синтаксис Automation:

ksEditListTypeEnum ListType = iObject.ListType	Получить свойство (*)
iObject.ListType = ListType	Установить свойство (*)
ksEditListTypeEnum ListType = iObject.GetListType()	Получить свойство(**)
iObject.SetListType (ListType)	Установить свойство (**)

Синтаксис COM:

iObject->get_ListType (&ListType)	Получить свойство
iObject->put_ListType (ListType)	Установить свойство

MultySelect – Возможность множественного выделения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
BOOL MultySelect = iObject.MultySelect                  Получить свойство (* )
iObject.MultySelect = MultySelect                       Установить свойство (* )
BOOL MultySelect = iObject.GetMultySelect()             Получить свойство(**)
iObject.SetMultySelect (MultySelect)                    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_MultySelect (&MultySelect)     Получить свойство
iObject->put_MultySelect (MultySelect)      Установить свойство
```

Sort – Состояние сортированности списка

Интерфейс...

Тип данных:BOOL.

Синтаксис Automation:

```
BOOL Sort = iObject.Sort                    Получить свойство (* )
iObject.Sort = Sort                         Установить свойство (* )
BOOL Sort = iObject.GetSort()               Получить свойство(**)
iObject.SetSort (Sort)                      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Sort (&Sort)                   Получить свойство
iObject->put_Sort (Sort)                    Установить свойство
```

IPropertyEditList – методы

Add – Добавить значение в список

Интерфейс...

Синтаксис Automation:

```
void Add( LPCTSTR  NewVal );
```

Синтаксис COM:

```
HRESULT Add( BSTR  NewVal );
```

Входные параметры:

NewVal - добавляемое значение.

BeginEditItem – Запустить редактирование значения в списке

Интерфейс...

Синтаксис Automation:

BOOL BeginEditItem(long Index);

Синтаксис COM:

HRESULT BeginEditItem(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

index	- индекс значения.

ClearList – Очистить список значений

Интерфейс...

Синтаксис Automation:

void ClearList();

Синтаксис COM:

HRESULT ClearList();

Delete – Удалить элемент из списка по значению или индексу

Интерфейс...

Синтаксис Automation:

void Delete (VARIANT Index);

Синтаксис COM:

HRESULT Delete (VARIANT Index);

Входные параметры:

index	- элемент из списка; значение элемента - строка или индекс элемента типа long.

Find – Найти индекс в списке по значению

Интерфейс...

Синтаксис Automation:

long Find(VARIANT Val);

Синтаксис COM:

HRESULT Find(VARIANT Val, long * Result);

Возвращаемое значение:

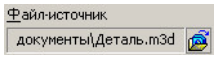
- индекс искомого значения.

Входные параметры:

Val - искомое значение.

Интерфейс IPropertyFileName

Интерфейс элемента управления Панели свойств – Выбор файла.



Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyFileName

Описание:

Элемент управления состоит из текстового поля и кнопки вызова диалога открытия файла.

Примечание:

1. Данный интерфейс можно получить, используя коллекцию элементов управления Панели свойств IPropertyControls.
1. При изменении значения параметра вызывается событие ksPropertyManagerNotify::ChangeControlValue.

IPropertyFileName – свойства

Bitmap – Рисунок кнопки

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Синтаксис Automation:

iObject.Bitmap = Bitmap Установить свойство (*)
iObject.SetBitmap (Bitmap) Установить свойство (**)

Синтаксис COM:

iObject->put_Bitmap (CreateOpenButton) Установить свойство

Примечание:

1. Свойство доступно только для записи.
2. Позволяет задать изображение для кнопки вызова диалога открытия файла.
3. Изображение можно задать в виде идентификаторов в ресурсе dll-модуля (тип данных long) или в виде полного пути к bmp-файлу рисунка (тип данных BSTR).
4. Если изображение задается в виде идентификатора, то обязательно должен быть задан dll-модуль, в ресурсах которого лежат эти рисунки (см. ResModule).

CreateOpenButton – Отображение кнопки выбора файла

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

CreateOpenButton = iObject.CreateOpenButton Получить свойство (*)
iObject.CreateOpenButton = CreateOpenButton Установить свойство (*)
CreateOpenButton = iObject.GetCreateOpenButton() Получить свойство(**)
iObject.SetCreateOpenButton (CreateOpenButton) Установить свойство (**)

Синтаксис COM:

iObject->get_CreateOpenButton (&CreateOpenButton) Получить свойство
iObject->put_CreateOpenButton (CreateOpenButton) Установить свойство

Значения свойства:

TRUE - справа от текстового поля будет находится кнопка вызова диалога открытия файла,
FALSE - элемент управления будет состоять только из текстового поля (без кнопки).

DefaultValue – Значение по умолчанию

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DefaultValue = Object.DefaultValue Получить свойство (*)
Object.DefaultValue = DefaultValue Установить свойство (*)
DefaultValue = Object.GetDefaultValue() Получить свойство(**)
Object.SetDefaultValue(DefaultValue) Установить свойство (**)

Синтаксис COM:

```
Object.get_DefaultValue( &DefaultValue )    Получить свойство
Object.put_DefaultValue(  DefaultValue )    Установить свойство
```

Extension – Расширение имени файла

Интерфейс...

Тип данных: BSTR (строка).

Синтаксис Automation:

```
Extension = iObject.Extension               Получить свойство (* )
iObject.Extension = Extension               Установить свойство (* )
Extension = iObject.GetExtension()          Получить свойство(**)
iObject.SetExtension (Extension)            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Extension (&Extension)         Получить свойство
iObject->put_Extension (Extension)          Установить свойство
```

Примечание:

1. Свойство позволяет получить/задать расширение имени файла для диалога открытия файла, используемое по умолчанию.
2. Если пользователь при задании имени файла не укажет расширение, автоматически для открытия будут предлагаться файлы с заданным расширением имени.
3. Если присвоить свойству значение NULL, то расширение файлов должно задаваться пользователем явно.

Filter – Активный фильтр типов файлов

Интерфейс...

Тип данных: BSTR (строка).

Синтаксис Automation:

```
Filter = iObject.Filter                     Получить свойство (* )
iObject.Filter = Filter                     Установить свойство (* )
Filter = iObject.GetFilter()                Получить свойство(**)
iObject.SetFilter (Filter)                  Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Filter (&Filter)               Получить свойство
iObject->put_Filter (Filter)                Установить свойство
```

Примечание:

Диалог открытия файла имеет список так называемых фильтров, включающих названия типов файлов и расширения имен файлов данного типа. Выбрав фильтр, пользователь указывает, что он желает работать только с файлами определенного типа, имеющими соответствующее расширение. Файлы с другими расширениями в диалоге не отображаются. Одновременно можно указать несколько фильтров. Каждый фильтр задается двумя строками - строкой, содержащей имя фильтра, и строкой, в которой перечислены соответствующие ему расширения имен файлов. Если одному типу соответствует несколько расширений, они разделяются символом «;». Строка, содержащая имя фильтра, отделяется от строки с расширениями файлов символом «|». Если используется несколько фильтров, то они также отделяются друг от друга символом «|». Например, в качестве строки, задающей фильтры, можно использовать строку вида: КОМПАС-Детали (*.m3d)|*.m3d|Все файлы (*.*)|*.*||.

Preview – Предварительный просмотр документа в диалоговом окне

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Preview = Object.Preview	Получить свойство (*)
Object.Preview = Preview	Установить свойство (*)
Preview = Object.GetPreview()	Получить свойство(**)
Object.SetPreview(Preview)	Установить свойство (**)

Синтаксис COM:

Object.get_Preview(&Preview)	Получить свойство
Object.put_Preview(Preview)	Установить свойство

ReadOnly – Запрет редактирования значения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = iObject.ReadOnly	Получить свойство (*)
iObject.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = iObject.GetReadOnly()	Получить свойство(**)
iObject.SetReadOnly (ReadOnly)	Установить свойство (**)

Синтаксис COM:

iObject->get_ReadOnly (&ReadOnly)	Получить свойство
iObject->put_ReadOnly (ReadOnly)	Установить свойство

Значения свойства:

TRUE - запрещает ввод нового значения с клавиатуры в текстовом поле,
FALSE - разрешает ввод нового значения с клавиатуры в текстовом поле.

ResModule – Модуль с ресурсом рисунка кнопки

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Синтаксис Automation:

```
ResModule = iObject.ResModule          Получить свойство (* )
iObject.ResModule = ResModule          Установить свойство (* )
ResModule = iObject.GetResModule()     Получить свойство(**)
iObject.SetResModule (ResModule)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ResModule (ResModule)     Получить свойство
iObject->put_ResModule (ResModule)     Установить свойство
```

Значения свойства:

HINSTANCE (тип значения long),
полный путь к файлу (тип значения BSTR) dll модуля, в котором находятся все необходимые ресурсы.

Примечание:

Данное свойство позволяет установить или получить dll-модуль, в котором находится рисунок для кнопки (см. Bitmap).

Width – Шаблон для расчета ширины элемента управления в горизонтальном положении (в символах)

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Width = iObject.Width                  Получить свойство (* )
iObject.Width = Width                  Установить свойство (* )
Width = iObject.GetWidth()             Получить свойство(**)
iObject.SetWidth (Width)               Установить свойство (**)
```

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значения свойства:

Количество видимых символов в текстовом поле при горизонтальном положении Панели свойств.

Примечание:

Для расчета ширины элемента управления в горизонтальном положении используется символ «0».

Интерфейс IPropertyGrid

Интерфейс элемента управления Панели свойств типа Сетка.

Параметры	Значение
Шаг резьбы	2.5
Размер под ключ	30
Высота головки	16
Масса 1000 шт	71.44

Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyGrid

IPropertyToolBar

Примечание:

1. Данный интерфейс можно получить через коллекцию элементов управления Панели свойств IPropertyControls.
2. После создания элемента управления и вывода его на экран все изменения вносимые с помощью свойств и методов данного интерфейса вступят в силу только после вызова метода UpdateParam.
3. Для IPropertyGrid можно создавать панели команд. Для создания команд в IPropertyGrid добавлена поддержка интерфейса IPropertyToolBar. Этот дополнительный интерфейс вида можно получить у интерфейса IPropertyGrid посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).
4. Для IPropertyGrid поддерживаются события менеджера панели свойств ChangeControlValue, ControlCommand, SelectItem.

IPropertyGrid – свойства

AutoSizeColumns – Авторазмер колонок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AutoSizeColumns = iObject.AutoSizeColumns                Получить свойство (* )
iObject.AutoSizeColumns = AutoSizeColumns                Установить свойство (* )
AutoSizeColumns = iObject.GetAutoSizeColumns()           Получить свойство(**)
iObject.SetAutoSizeColumns (AutoSizeColumns)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_AutoSizeColumns (&AutoSizeColumns)    Получить свойство
iObject->put_AutoSizeColumns (AutoSizeColumns)     Установить свойство
```

Значение свойства:

```
TRUE      - расчет размера колонок от ширины элемента управления,
FALSE     - фиксированный размер колонок.
```

Примечание:

Ширина колонки определяется путем деления всей ширины элемента управления на количество колонок.

CellFormat – Формат вывода текста в ячейке

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
CellFormat = iObject.CellFormat (nRow, nCol)          Получить свойство (* )
iObject.CellFormat (nRow, nCol) = CellFormat          Установить свойство (* )
CellFormat = iObject.GetCellFormat (nRow, nCol)       Получить свойство(**)
iObject.SetCellFormat (nRow, nCol, CellFormat)        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CellFormat (nRow, nCol, &CellFormat)    Получить свойство
iObject->put_CellFormat (nRow, nCol, CellFormat)     Установить свойство
```

Входные параметры:

```
nRow (long)      - индекс строки,
nCol (long)      - индекс колонки.
```

Значение свойства:

формат вывода текста в ячейке.

Примечание:

Формат вывода текста в ячейке задается стандартными значениями, используемыми в методе DrawText ОС Windows.

CellText – Текст в ячейке

Интерфейс...

Тип данных: строка

Синтаксис Automation:

cllText = iObject.CellText (nRow, nCol)	Получить свойство (*)
iObject.CellText (nRow, nCol) = CellText	Установить свойство (*)
CellText = iObject.GetCellText (nRow, nCol)	Получить свойство(**)
iObject.SetCellText (nRow, nCol, CellText)	Установить свойство (**)

Синтаксис COM:

iObject->get_CellText (nRow, nCol, &CellText)	Получить свойство
iObject->put_CellText (nRow, nCol, CellText)	Установить свойство

Входные параметры:

nRow (long)	- индекс строки,
nCol (long)	- индекс колонки.

Значение свойства:

Текст в ячейке.

Примечание:

Позволяет получить/установить текст в ячейке.

ColumnCount – Количество колонок

Интерфейс...

Тип данных: long.

Синтаксис Automation:

ColumnCount = iObject.ColumnCount	Получить свойство (*)
iObject.ColumnCount = ColumnCount	Установить свойство (*)
ColumnCount = iObject.GetColumnCount()	Получить свойство(**)
iObject.SetColumnCount (ColumnCount)	Установить свойство (**)

Синтаксис COM:

```
iObject->get_ColumnCount (&ColumnCount)      Получить свойство
iObject->put_ColumnCount (ColumnCount)       Установить свойство
```

Значение свойства:

Общее количество колонок в сетке.

ColumnWidth – Ширина колонки

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
ColumnWidth = iObject.ColumnWidth (index)          Получить свойство (* )
iObject.ColumnWidth (index) = ColumnWidth          Установить свойство (* )
ColumnWidth = iObject.GetColumnWidth (index)       Получить свойство(**)
iObject.SetColumnWidth (index, ColumnWidth)        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ColumnWidth (index, &ColumnWidth)     Получить свойство
iObject->put_ColumnWidth (index, ColumnWidth)      Установить свойство
```

Входные параметры:

index - индекс колонки.

Значение свойства:

Ширина колонки с индексом index в сетке (в пикселах).

ColumnVisible – Видимость колонки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ColumnVisible = Object.ColumnVisible( index )      Получить свойство (* )
Object.ColumnVisible( index ) = ColumnVisible      Установить свойство (* )
ColumnVisible = Object.GetColumnVisible( index )   Получить свойство(**)
Object.SetColumnVisible( index, ColumnVisible )    Установить свойство (**)
```

Синтаксис COM:

Object.get_ColumnVisible(index, &ColumnVisible)	Получить свойство
Object.put_ColumnVisible(index, ColumnVisible)	Установить свойство

Входные параметры:

index	- индекс колонки.

CurrentColumn – Текущая колонка

Интерфейс...

Тип данных: long.

Синтаксис Automation:

CurrentColumn = iObject.CurrentColumn	Получить свойство (*)
iObject.CurrentColumn = CurrentColumn	Установить свойство (*)
CurrentColumn = iObject.GetCurrentColumn()	Получить свойство(**)
iObject.SetCurrentColumn(CurrentColumn)	Установить свойство (**)

Синтаксис COM:

iObject->get_CurrentColumn(&CurrentColumn)	Получить свойство
iObject->put_CurrentColumn(CurrentColumn)	Установить свойство

Примечание:

Позволяет получить и установить текущую колонку для таблицы.

CurrentRow – Текущая строка

Интерфейс...

Тип данных: long.

Синтаксис Automation:

CurrentRow = iObject.CurrentRow	Получить свойство (*)
iObject.CurrentRow = CurrentRow	Установить свойство (*)
CurrentRow = iObject.GetCurrentRow()	Получить свойство(**)
iObject.SetCurrentRow(CurrentRow)	Установить свойство (**)

Синтаксис COM:

iObject->get_CurrentRow(&CurrentRow)	Получить свойство
iObject->put_CurrentRow(CurrentRow)	Установить свойство

Примечание:

Позволяет получить и установить текущую строку для таблицы.

EnableDeleteRows – Разрешить удаление строк

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableDeleteRows = Object.EnableDeleteRows	Получить свойство (*)
Object.EnableDeleteRows = EnableDeleteRows	Установить свойство (*)
EnableDeleteRows = Object.GetEnableDeleteRows()	Получить свойство(**)
Object.SetEnableDeleteRows(EnableDeleteRows)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableDeleteRows(&EnableDeleteRows)	Получить свойство
Object.put_EnableDeleteRows(EnableDeleteRows)	Установить свойство

EnableEdit – Запрет редактирования ячеек

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

EnableEdit = Object.EnableEdit(NRow, NCol)	Получить свойство (*)
Object.EnableEdit(NRow, NCol) = EnableEdit	Установить свойство (*)
EnableEdit = Object.GetEnableEdit(NRow, NCol)	Получить свойство(**)
Object.SetEnableEdit(NRow, NCol, EnableEdit))	Установить свойство (**)

Синтаксис COM:

Object.get_EnableEdit(NRow, NCol, &EnableEdit)	Получить свойство
Object.put_EnableEdit(NRow, NCol, EnableEdit)	Установить свойство

Входные параметры:

long NRow	- номер строки
long NCol	- номер столбца

FixedColumnCount – Количество фиксированных колонок

Интерфейс...

Тип данных: long.

Синтаксис Automation:

FixedColumnCount = iObject.FixedColumnCount	Получить свойство (*)
iObject.FixedColumnCount = FixedColumnCount	Установить свойство (*)
FixedColumnCount = iObject.GetFixedColumnCount()	Получить свойство(**)
iObject.SetFixedColumnCount (FixedColumnCount)	Установить свойство (**)

Синтаксис COM:

iObject->get_FixedColumnCount (&FixedColumnCount)	Получить свойство
iObject->put_FixedColumnCount (FixedColumnCount)	Установить свойство

Значение свойства:

Количество фиксированных колонок в сетке.

FixedRowCount – Количество фиксированных строк

Интерфейс...

Тип данных: long.

Синтаксис Automation:

FixedRowCount = iObject.FixedRowCount	Получить свойство (*)
iObject.FixedColumnCount = FixedColumnCount	Установить свойство (*)
FixedColumnCount = iObject.GetFixedColumnCount()	Получить свойство(**)

iObject.SetFixedColumnCount (FixedColumnCount)	Установить свойство (**)

Синтаксис COM:

iObject->get_FixedColumnCount (&FixedColumnCount)	Получить свойство
iObject->put_FixedColumnCount (FixedColumnCount)	Установить свойство

Значение свойства:

Количество фиксированных строк в сетке.

Height – Высота элемента управления

Интерфейс...

Тип данных: long

Синтаксис Automation:

Height = iObject.Height	Получить свойство (*)
iObject.Height = Height	Установить свойство (*)
Height = iObject.GetHeight()	Получить свойство(**)
iObject.SetHeight (Height)	Установить свойство (**)

Синтаксис COM:

iObject->get_Height (&Height)	Получить свойство
iObject->put_Height (Height)	Установить свойство

Значение свойства:

Высота видимой части элемента управления (в пикселах).

RowCount – Количество строк

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowCount = iObject.RowCount	Получить свойство (*)
iObject.RowCount = RowCount	Установить свойство (*)
RowCount = iObject.GetRowCount()	Получить свойство(**)
iObject.SetRowCount (RowCount)	Установить свойство (**)

Синтаксис COM:

iObject->get_RowCount (&RowCount)	Получить свойство
iObject->put_RowCount (RowCount)	Установить свойство

Значение свойства:

Общее количество строк в сетке.

RowHeight – Высота строки

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowHeight = iObject.RowHeight (index)	Получить свойство (*)
iObject.RowHeight (index) = RowHeight	Установить свойство (*)
RowHeight = iObject.GetRowHeight (index)	Получить свойство(**)
iObject.SetRowHeight (index, RowHeightr)	Установить свойство (**)

Синтаксис COM:

iObject->get_RowHeight (index, &RowHeight)	Получить свойство
iObject->put_RowHeight (index, RowHeight)	Установить свойство

Входные параметры:

index	- индекс строки.

Значение свойства:

Высота строки с индексом index в сетке (в пикселах).

Width – Ширина элемента управления

Интерфейс...

Тип данных: long

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значение свойства:

Ширина видимой части элемента управления (в пикселах).

ReadOnly – Запрет редактирования таблицы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = iObject.ReadOnly	Получить свойство (*)
iObject.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = iObject.GetReadOnly()	Получить свойство(**)
iObject.SetReadOnly (ReadOnly)	Установить свойство (**)

Синтаксис COM:

iObject->get_ReadOnly (&ReadOnly)	Получить свойство
iObject->put_ReadOnly (ReadOnly)	Установить свойство

Значение свойства:

TRUE	- редактирование ячеек запрещено,
FALSE	- не реализовано.

Примечание:

Временно сетка работает только в режиме **Только для чтения**. Редактирование ее ячеек запрещено.

IPropertyGrid – методы

UpdateParam – Обновить параметры сетки

Интерфейс...

Синтаксис Automation:

void UpdateParam();

Синтаксис COM:

HRESULT UpdateParam();

Примечание:

Все изменения в сетке вступят в силу только после вызова этого метода.

Интерфейс IPropertyLibExplorer

Элемент панели свойств Отображение библиотеки документов.

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyLibExplorer

Примечание:

1. Интерфейс позволяет просматривать библиотеки фрагментов(*.lfr) и библиотеки моделей (*.l3d). Путь к выбранному элементу можно получить через IPropertyControl::Value. Для установки активного элемента также используется IPropertyControl::Value.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyLibExplorer – свойства

FileName – Имя файла библиотеки документов

Интерфейс...

Тип данных: BSTR.

Синтаксис Automation:

FileName = iObject.FileName	Получить свойство (*)
iObject.FileName = FileName	Установить свойство (*)
FileName = iObject.GetFileName( )	Получить свойство(**)
iObject.SetFileName (FileName)	Установить свойство (**)

Синтаксис COM:

iObject->get_FileName (&FileName)	Получить свойство
iObject->put_FileName (FileName)	Установить свойство

Примечание:

Полное имя к файлу библиотеки документов.

FileValue – Признак – выбранный узел является файлом

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE	- выбранный узел является файлом,
FALSE	- выбранный узел не является файлом.

Синтаксис Automation:

FileValue = iObject.FileValue	Получить свойство (*)
FileValue = iObject.GetFileValue( )	Получить свойство(**)

Синтаксис COM:

iObject->get_FileValue (&FileValue)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. В дереве библиотеки документов могут быть папки и файлы. Путь к выбранному элементу можно получить, используя свойство IPropertyControl::Value.

RootName – Имя верхнего узла дерева

Интерфейс...

Тип данных: BSTR.

Синтаксис Automation:

RootName = iObject.RootName	Получить свойство (*)
iObject.RootName = RootName	Установить свойство (*)
FileName = iObject.GetRootName( )	Получить свойство(**)
iObject.SetRootName (RootName)	Установить свойство (**)

Синтаксис COM:

iObject->get_RootName (&RootName)	Получить свойство
iObject->put_RootName (RootName)	Установить свойство

Примечание:

1. Имя верхнего узла дерева внутренней структуры библиотеки документов. После вызова установки полного пути к библиотеке документов IPropertyLibExplorer::FileName считывается как ее имя.
2. Задать значение свойства можно только до запуска процесса.

Интерфейс IPropertyList

Интерфейс элемента управления Панели свойств Поле со списком.

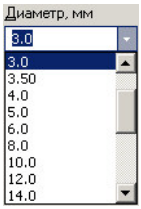
Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyList

Примечание:



Элемент панели свойств

1. Данный интерфейс можно получить через интерфейс коллекции элементов управления панели свойств IPropertyControls.
2. При нажатии на кнопку элемента управления вызывается событие ksPropertyManagerNotify::ControlCommand.
3. При изменении значения элемента управления вызывается событие ksIPropertyManagerNotify::ChangeControlValue.

IPropertyList – свойства

CheckState – Состояние опции

Интерфейс...

Тип данных: состояние опции из перечисления CheckStateEnum.

Синтаксис Automation:

```
CheckState = iObject.CheckState                Получить свойство (* )
iObject.CheckState = CheckState                Установить свойство (* )
CheckState = iObject.GetCheckState()           Получить свойство(**)
iObject.SetCheckState (CheckState)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CheckState (&CheckState)          Получить свойство
iObject->put_CheckState (CheckState)           Установить свойство
```

Примечание:

Это свойство влияет на состояние переключателя, если свойство WithCheck имеет значение TRUE, и на отображение текста в поле элемента управления, подробнее см. CheckStateEnum.

Count – Количество элементов списка

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Count = Object.Count                           Получить свойство (* )
```

Count = Object.GetCount() Получить свойство(**)

Синтаксис COM:

Object.get_Count(&Count) Получить свойство

Примечание:

Свойство доступно только для чтения.

Expanded – Шаг

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step Получить свойство (*)
Object.Step = Step Установить свойство (*)
Step = Object.GetStep() Получить свойство(**)
Object.SetStep(Step) Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step) Получить свойство
Object.put_Step(Step) Установить свойство

Шаг изменения значения колесом мыши.

MaxValue – Максимальное значение

Интерфейс...

Тип данных: VARIANT (long или double).

Синтаксис Automation:

MaxValue = iObject.MinValue Получить свойство (*)
MaxValue = iObject.GetMinValue() Получить свойство(**)

Синтаксис COM:

iObject->get_MaxValue (&MaxValue) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,

- для типа ksControlListStr ограничения на значение не накладываются,
- для типа ksControlListLength используется тип double,
- для типа ksControlListAngle используется тип double.

MinValue – Минимальное значение

Интерфейс...

Тип данных: VARIANT (long или double).

Синтаксис Automation:

```
MinValue = iObject.MinValue          Получить свойство (* )
MinValue = iObject.GetMinValue()     Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_MinValue (&MinValue)    Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlListStr ограничения на значение не накладываются,
- для типа ksControlListLength используется тип double,
- для типа ksControlListAngle используется тип double.

Precision – Точность сравнения

Интерфейс...

Тип данных: double.

Синтаксис Automation:

```
Precision = iObject.Precision        Получить свойство (* )
iObject.Precision = Precision        Установить свойство (* )
Precision = iObject.GetPrecision()   Получить свойство(**)
iObject.SetPrecision(Precision)      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Precision(&Precision)   Получить свойство
iObject->put_Precision(Precision)    Установить свойство
```

Примечание:

1. Свойство позволяет установить точность сравнения значений в списке. Сравнение происходит при добавлении значения в список IPropertyList::Add.
2. Если добавляемое значение в списке есть, то оно не добавляется при установке значения IPropertyControl::Value.
3. Если устанавливается значение, которое есть в списке, оно селектируется при поиске IPropertyList::Find.
4. Свойство работает для вещественных типов списка: ksControlListReal, ksControlListLength, ksControlListAngle.
5. По умолчанию устанавливается точность 0.000001.

ReadOnly – Запрет редактирования значения

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

ReadOnly = iObject.ReadOnly	Получить свойство (*)
iObject.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = iObject.GetReadOnly()	Получить свойство(**)
iObject.SetReadOnly (ReadOnly)	Установить свойство (**)

Синтаксис COM:

iObject->get_ReadOnly (&ReadOnly)	Получить свойство
iObject->put_ReadOnly (ReadOnly)	Установить свойство

Значение свойства:

TRUE	- ввод нового значения с клавиатуры в раскрывающийся список запрещен; элемент управления работает в режиме DropList,
FALSE	- ввод нового значения с клавиатуры в раскрывающийся список разрешен; элемент управления работает в режиме DropDown,

Sort – Сортировать список

Интерфейс...

Тип данных BOOL.

Синтаксис Automation:

Sort = iObject.Sort	Получить свойство (*)
iObject.Sort = Sort	Установить свойство (*)
Sort = iObject.GetSort()	Получить свойство(**)
iObject.SetSort (Sort)	Установить свойство (**)

Синтаксис COM:

iObject->get_Sort (&Sort)	Получить свойство
iObject->put_Sort (Sort)	Установить свойство

Значение свойства:

TRUE	- значения в списке будут сортироваться при добавлении новых,
FALSE	- значения в списке будут в той последовательности, в какой они были добавлены.

Width– Шаблон для расчета ширины элемента управления в горизонтальном положении (в символах)

Интерфейс...

Тип данных long.

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значение свойства:

Количество видимых символов в поле при горизонтальном положении Панели свойств.

Примечание:

Для расчета ширины элемента управления используется символ «0».

WithCheck – Наличие опции рядом с полем ввода

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

WithCheck = iObject.WithCheck	Получить свойство (*)
iObject.WithCheck = WithCheck	Установить свойство (*)
WithCheck = iObject.GetWithCheck()	Получить свойство(**)
iObject.SetWithCheck (WithCheck)	Установить свойство (**)

Синтаксис COM:

iObject->get_WithCheck (&WithCheck)	Получить свойство
iObject->put_WithCheck (WithCheck)	Установить свойство

Значение свойства:

TRUE	- рядом с элементом управления выводится опция,
FALSE	- рядом с элементом управления не выводится опция.

Примечание:

На текущий момент свойство не реализовано, его значение можно установить только равным FALSE.

IPropertyList – методы

Add – Добавить значение в список

Интерфейс...

Синтаксис Automation:

void Add (const VARIANT& newVal);

Синтаксис COM:

HRESULT Add ([in]VARIANT newVal);

Входные параметры:

newVal	- новое значение в раскрывающемся списке.

Примечание:

1. Тип данных нового значения определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlListStr ограничения на значение не накладываются,
- для типа ksControlListLength используется тип double,
- для типа ksControlListAngle используется тип double.

2. В список можно добавить только уникальные значения. Если в списке уже есть значение, которое пытаются добавить повторно, то оно не будет добавлено.
3. В список можно сразу передать массив значений. Пример...
4. См. также: IPropertyControl::Value - Значение элемента управления.

ClearList – Очистить список значений

Интерфейс...

Синтаксис Automation:

void ClearList();

Синтаксис COM:

HRESULT ClearList();

Примечание:

Очищает раскрывающийся список и содержимое его поля ввода.

Find – Найти индекс строки в списке по значению

Интерфейс...

Синтаксис Automation:

long Find (const VARIANT FAR& val);

Синтаксис COM:

HRESULT Find([in] VARIANT Val, [out, retval] long * PVal);

Входные параметры:

val - значение в раскрывающемся списке.

Возвращаемое значение:

Индекс строки со значением val в списке - в случае успеха,
-1 - в случае неудачи.

Примечание:

Метод работает только на несортированном списке, см. IPropertyList::Sort.

SetCurrentByIndex – Установить текущую строку по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetCurrentByIndex(long index);

Синтаксис COM:

HRESULT SetCurrentByIndex([in] long Index, [out, retval] VARIANT_BOOL * PVal);

Входные параметры:

index - номер строки в раскрывающемся списке.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод работает только на несортированном списке, см. IPropertyList::Sort.

SetValueRange – Установить новые ограничения на значение

Интерфейс...

Синтаксис Automation:

BOOL SetValueRange (const VARIANT& minVal, const VARIANT& maxVal);

Синтаксис COM:

HRESULT SetValueRange ([in] VARIANT minVal, [in] VARIANT maxVal, [out, retval] VARIANT_BOOL * pVal);

Входные параметры:

minVal	- минимальное допустимое значение (тип данных long или double),
maxVal	- максимальное допустимое значение (тип данных long или double).

Примечание:

Тип данных максимального и минимального допустимого значения определяется типом элемента управления из перечисления ControlTypeEnum:

- для типа ksControlListInt используется тип long,
- для типа ksControlListReal используется тип double,
- для типа ksControlListStr ограничения на значение не накладываются,
- для типа ksControlListLength используется тип double,
- для типа ksControlListAngle используется тип double.

Интерфейс IPropertyMultiButton

Интерфейс элемента управления Панели свойств Набор кнопок.

Элемент панели свойств

IKompasAPIObject

IPropertyControl

IPropertyMultiButton

Примечание:

1. Элемент управления состоит из набора кнопок определенного типа из перечисления ButtonTypeEnum.
2. Данный интерфейс можно получить через интерфейс коллекции элементов управления панели свойств IPropertyControls.

3. При нажатии на одну из кнопок набора вызывается событие ksPropertyManagerNotify::ControlCommand.
4. Для корректного отображения строки подсказки конкретной кнопки, задаваемой при помощи свойств ButtonHint или ButtonTips, необходимо задать строку подсказки для набора кнопок. Если текст подсказки набора не нужен или не имеет значения, эта строка может состоять, например, из одного пробела.

IPropertyMultiButton – свойства

ButtonChecked – Нажатие кнопки

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

ButtonChecked = iObject.ButtonChecked (btnID)	Получить свойство (*)
iObject.ButtonChecked (btnID) = ButtonChecked	Установить свойство (*)
ButtonChecked = iObject.GetButtonChecked (btnID)	Получить свойство(**)
iObject.SetButtonChecked (btnID, ButtonChecked)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonChecked (btnID, &ButtonChecked)	Получить свойство
iObject->put_ButtonChecked (btnID, ButtonChecked)	Установить свойство

Входные параметры:

btnID	- идентификатор кнопки.

Значение свойства:

TRUE	- кнопка нажата,
FALSE	- кнопка не нажата.

Примечание:

Доступно только для кнопок типа ksCheckButton и ksRadioButton из перечисления ButtonTypeEnum.

ButtonEnable – Доступность кнопки

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

ButtonEnable = iObject.ButtonEnable (btnID)	Получить свойство (*)
iObject.ButtonEnable (btnID) = ButtonEnable	Установить свойство (*)
ButtonEnable = iObject.GetButtonEnable (btnID)	Получить свойство(**)
iObject.SetButtonEnable (btnID, ButtonEnable)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonEnable (btnID, &ButtonEnable)	Получить свойство
iObject->put_ButtonEnable (btnID, ButtonEnable)	Установить свойство

Входные параметры:

btnID	- идентификатор кнопки.

Значение свойства:

TRUE	- кнопка доступна,
FALSE	- кнопка недоступна.

Примечание:

Данное свойство позволяет установить или получить доступ к кнопке.

ButtonHint – Строка подсказки для кнопки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ButtonHint = iObject.ButtonHint (btnID)	Получить свойство (*)
iObject.ButtonHint (btnID) = ButtonHint	Установить свойство (*)
ButtonHint = iObject.GetButtonHint (btnID)	Получить свойство(**)
iObject.SetButtonHint (btnID, ButtonHint)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonHint (btnID, &ButtonHint)	Получить свойство
iObject->put_ButtonHint (btnID, ButtonHint)	Установить свойство

Входные параметры:

btnID	- идентификатор кнопки.

Примечание:

1. Данное свойство позволяет установить или получить строку подсказки для кнопки с идентификатором btnID.
2. Для корректного отображения строки подсказки конкретной кнопки необходимо задать строку подсказки для набора кнопок. Если текст подсказки набора не нужен или не имеет значения, эта строка может состоять, например, из одного пробела.

ButtonIconFont – Шрифт для кнопок

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ButtonIconFont = Object.ButtonIconFont(BtnID)	Получить свойство (*)
Object.ButtonIconFont(BtnID) = ButtonIconFont	Установить свойство (*)
ButtonIconFont = Object.GetButtonIconFont(BtnID)	Получить свойство(**)
Object.SetButtonIconFont(BtnID, ButtonIconFont)	Установить свойство (**)

Синтаксис COM:

Object.get_ButtonIconFont(BtnID, &ButtonIconFont)	Получить свойство
Object.put_ButtonIconFont(BtnID, ButtonIconFont)	Установить свойство

Входные параметры:

BtnID	- идентификатор кнопки.

ButtonTips – Строка подсказки для кнопки

Интерфейс...

Тип данных BSTR

Синтаксис Automation:

ButtonTips = iObject.ButtonTips (btnID)	Получить свойство (*)
iObject.ButtonTips (btnID) = ButtonTips	Установить свойство (*)
ButtonTips = iObject.GetButtonTips (btnID)	Получить свойство(**)
iObject.SetButtonTips (btnID, ButtonTips)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonTips (btnID, &ButtonTips)	Получить свойство
iObject->put_ButtonTips (btnID, ButtonTips)	Установить свойство

Входные параметры:

btnID - идентификатор кнопки.

Примечание:

1. Данное свойство позволяет установить или получить строку подсказки для кнопки с идентификатором btnID.
2. Для корректного отображения строки подсказки конкретной кнопки необходимо задать строку подсказки для набора кнопок. Если текст подсказки набора не нужен или не имеет значения, эта строка может состоять, например, из одного пробела.

ButtonsType – Тип кнопок

Интерфейс...

Тип данных: тип кнопок из перечисления ButtonTypeEnum.

Синтаксис Automation:

ButtonsType = iObject.ButtonsType	Получить свойство (*)
iObject.ButtonsType = ButtonsType	Установить свойство (*)
ButtonsType = iObject.GetButtonsType()	Получить свойство(**)
iObject.SetButtonsType (ButtonsType)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonsType (&ButtonsType)	Получить свойство
iObject->put_ButtonsType (ButtonsType)	Установить свойство

Примечание:

Изменить тип кнопок можно только до создания элемента управления, то есть до запуска процесса у процессных параметров или до вызова метода IPropertyManager::ShowTabs.

ButtonVisible – Отображение кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ButtonVisible = Object.ButtonVisible(BtnID)	Получить свойство (*)
Object.ButtonVisible(BtnID) = ButtonVisible	Установить свойство (*)

ButtonVisible = Object.GetButtonVisible(BtnID)	Получить свойство(**)
Object.SetButtonVisible(BtnID, ButtonVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_ButtonVisible(BtnID, &ButtonVisible)	Получить свойство
Object.put_ButtonVisible(BtnID, ButtonVisible)	Установить свойство

Входные параметры:

BtnID	- идентификатор кнопки.

ResModule – Модуль с ресурсами значков

Интерфейс...

Тип данных: VARIANT (long или BSTR)

Синтаксис Automation:

ResModule = iObject.ResModule	Получить свойство (*)
iObject.ResModule = ResModule	Установить свойство (*)
ResModule = iObject.GetResModule()	Получить свойство(**)
iObject.SetResModule (ResModule)	Установить свойство (**)

Синтаксис COM:

iObject->get_ResModule (&ResModule)	Получить свойство
iObject->put_ResModule (ResModule)	Установить свойство

Значение свойства:

HINSTANCE (тип значения long) или полный путь к файлу (тип значения BSTR) dll модуля, в котором находятся все необходимые ресурсы.

Примечание:

Данное свойство позволяет установить или получить dll-модуль, из которого будут извлекаться значки для кнопок.

IPropertyMultiButton – методы

AddButton – Добавить кнопку

Интерфейс...

Синтаксис Automation:

void AddButton (long btnID, const VARIANT& bmp, long insertAt);

Синтаксис COM:

HRESULT AddButton ([in] long btnID, [in] VARIANT bmp, [in, defaultvalue (-1)] long insertAt);

Входные параметры:

btnID	- идентификатор добавляемой кнопки,
bmp	- идентификатор значка в файле ресурсов (тип данных long) или полный путь к bmp-файлу значка (тип данных BSTR),
insertAt	- индекс кнопки, перед которой нужно вставить добавляемую (-1 - в конец).

Примечание:

Если значок задается в виде идентификатора, то до этого должен быть задан dll-модуль, в ресурсах которого лежит этот значок. Если кнопка с заданным идентификатором уже есть в списке, то ничего добавляться не будет.

NextCommand – Переключать кнопки по VK_TAB

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

nextCommand = iKompasObject.NextCommand	Получить свойство (*)
iKompasObject.NextCommand = nextCommand	Установить свойство (*)
nextCommand = iKompasObject.GetNextCommand()	Получить свойство(**)
iKompasObject.SetNextCommand(nextCommand)	Установить свойство (**)

Примечание:

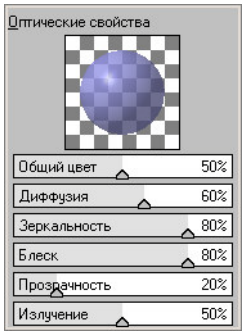
1. Доступно только для кнопок типа и ksRadioButton из перечисления ButtonTypeEnum.
2. Позволяет получить и установить свойство переключения кнопок комбинацией CTLRTAB.

Интерфейс IPropertyOpticalProps

Интерфейс элемента управления Панели свойств – Контрол оптических свойств.

Иерархия:

IDispatch



Элемент панели свойств

IKompasAPIObject

IPropertyControl

IPropertyOpticalProps

Данный интерфейс можно получить, используя коллекцию элементов управления Панели свойств IPropertyControls.

IPropertyOptical – свойства

ColorParam – Получить указатель на параметры

Интерфейс...

Тип данных: указатель на интерфейс IColorParam7

Синтаксис Automation:

```
ColorParam = Object.ColorParam          Получить свойство (* )
ColorParam = Object.GetColorParam()     Получить свойство(**)
```

Синтаксис COM:

```
Object.get_ColorParam( &ColorParam )    Получить свойство
```

Свойство позволяет получить указатель на интерфейс оптических свойств.

Примечание:

Свойство доступно только для чтения.

EnableAmbient – Доступность выбора общего цвета

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableAmbient = Object.EnableAmbient          Получить свойство (* )
```

```
Object.EnableAmbient = EnableAmbient                Установить свойство (* )
EnableAmbient = Object.GetEnableAmbient()           Получить свойство (**)
Object.SetEnableAmbient( EnableAmbient )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableAmbient( &EnableAmbient )          Получить свойство
Object.put_EnableAmbient(  EnableAmbient )          Установить свойство
```

EnableDiffuse – Доступность выбора диффузии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableDiffuse = Object.EnableDiffuse                Получить свойство (* )
Object.EnableDiffuse = EnableDiffuse                Установить свойство (* )
EnableDiffuse = Object.GetEnableDiffuse()           Получить свойство (**)
Object.SetEnableDiffuse( EnableDiffuse )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableDiffuse( &EnableDiffuse )          Получить свойство
Object.put_EnableDiffuse(  EnableDiffuse )          Установить свойство
```

EnableEmission – Доступность выбора излучения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableEmission = Object.EnableEmission              Получить свойство (* )
Object.EnableEmission = EnableEmission              Установить свойство (* )
EnableEmission = Object.GetEnableEmission()         Получить свойство (**)
Object.SetEnableEmission( EnableEmission )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableEmission( &EnableEmission )        Получить свойство
Object.put_EnableEmission(  EnableEmission )        Установить свойство
```

EnableShininess – Доступность выбора блеска

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableShininess = Object.EnableShininess	Получить свойство (*)
Object.EnableShininess = EnableShininess	Установить свойство (*)
EnableShininess = Object.GetEnableShininess()	Получить свойство (**)
Object.SetEnableShininess(EnableShininess))	Установить свойство (**)

Синтаксис COM:

Object.get_EnableShininess(&EnableShininess)	Получить свойство
Object.put_EnableShininess(EnableShininess)	Установить свойство

EnableSpecularity – Доступность выбора зеркальности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableSpecularity = Object.EnableSpecularity	Получить свойство (*)
Object.EnableSpecularity = EnableSpecularity	Установить свойство (*)
EnableSpecularity = Object.GetEnableSpecularity()	Получить свойство (**)
Object.SetEnableSpecularity(EnableSpecularity)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableSpecularity(&EnableSpecularity)	Получить свойство
Object.put_EnableSpecularity(EnableSpecularity)	Установить свойство

EnableTransparency – Доступность выбора прозрачности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableTransparency = Object.EnableTransparency	Получить свойство (*)
Object.EnableTransparency = EnableTransparency	Установить свойство (*)
EnableTransparency = Object.GetEnableTransparency()	Получить свойство (**)
Object.SetEnableTransparency(EnableTransparency)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableTransparency(&EnableTransparency)	Получить свойство
Object.put_EnableTransparency(EnableTransparency)	Установить свойство

IPropertyOptical – методы

Init – Инициализировать контрол по параметрам

Интерфейс...

Синтаксис Automation:

BOOL Init(IColorParam7 * PVal);

Синтаксис COM:

HRESULT Init(IColorParam7 * PVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

PVal - указатель на интерфейс оптических свойств.

Интерфейс IPropertySeparator

Интерфейс элемента управления Панели свойств типа разделитель (сепаратор).

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertySeparator

Примечание:

1. Данный интерфейс позволяет получить и установить параметры разделителя.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertySeparator – свойства

SeparatorType – Тип разделителя

Функция не поддерживается

Интерфейс...

Тип данных: тип разделителя из перечисления SeparatorTypeEnum.

Синтаксис Automation:

SeparatorType = iObject.SeparatorType	Получить свойство (*)
iObject.SeparatorType = SeparatorType	Установить свойство (*)

SeparatorType = iObject.GetSeparatorType() Получить свойство(**)
iObject.SetSeparatorType (SeparatorType) Установить свойство (**)

Синтаксис COM:

iObject->get_SeparatorType (&SeparatorType) Получить свойство
iObject->put_SeparatorType (SeparatorType) Установить свойство

Примечание:

Позволяет получить и установить тип разделителя.

IPropertySeparator – методы

SetImage – Установить изображение

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL SetImage (VARIANT idBmp, VARIANT hInstance);

Синтаксис COM:

HRESULT SetImage ([in]VARIANT idBmp,
[in]VARIANT hInstance,
[out, retval] VARIANT_BOOL* pVal);

Входные параметры:

idBmp - идентификатор значка в файле ресурсов (тип данных long) или полный путь к bmp файлу значка (тип данных BSTR),
hInstance - HINSTANCE модуля, в котором находится изображение.

Возвращаемое значение:

Результат выполнения метода.

Примечание:

Если значок задается в виде идентификатора, то должен быть задан dll-модуль hInstance, в ресурсах которого лежит этот значок.

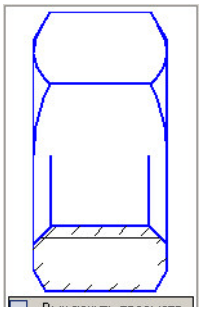
Интерфейс IPropertySlideBox

Интерфейс элемента управления Панели свойств типа Окно отображения слайда, растрового изображения, файла документа КОМПАС или группы файлов.

Иерархия:

IKompasAPIObject

IPropertyControl



Элемент панели свойств

IPropertySlideBox

Примечание:

1. Данный интерфейс можно получить через коллекцию элементов управления Панели свойств IPropertyControls.
2. После создания элемента управления и вывода его на экран все изменения, вносимые с помощью свойств и методов данного интерфейса, вступят в силу только после вызова метода UpdateParam.
3. Для обновления изображения в окне, например после изменения геометрии в отображаемом документе, можно использовать метод UpdateParam.

IPropertySlideBox – свойства

CheckBoxVisibility – Видимость опции, управляющей отрисовкой изображения

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

CheckBoxVisibility = iObject.CheckBoxVisibility	Получить свойство (*)
iObject.CheckBoxVisibility = CheckBoxVisibility	Установить свойство (*)
CheckBoxVisibility = iObject.GetCheckBoxVisibility()	Получить свойство(**)
iObject.SetCheckBoxVisibility (CheckBoxVisibility)	Установить свойство (**)

Синтаксис COM:

iObject->get_CheckBoxVisibility (&CheckBoxVisibility)	Получить свойство

iObject->put_CheckBoxVisibility (CheckBoxVisibility) — Установить свойство

Возвращаемое значение:

TRUE - опция отображается под слайдом,
FALSE - опция не отображается.

Примечание:

Свойство позволяет определить тип отображаемого значения.

DrawingSlide – Отображаемый слайд

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Синтаксис Automation:

DrawingSlide = iObject.DrawingSlide	Получить свойство (*)
iObject.DrawingSlide = DrawingSlide	Установить свойство (*)
DrawingSlide = iObject.GetDrawingSlide()	Получить свойство(**)
iObject.SetDrawingSlide (DrawingSlide)	Установить свойство (**)

Синтаксис COM:

iObject->get_DrawingSlide (&DrawingSlide)	Получить свойство
iObject->put_DrawingSlide (DrawingSlide)	Установить свойство

Значение свойства:

Идентификатор или указатель на группу, документ (тип значения long), или полный путь к файлу (тип значения BSTR) документа или текстового файла с описанием слайда.

Примечание:

Тип отображаемого объекта определяется свойством SlideType.

Height – Высота элемента управления

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Height = iObject.Height	Получить свойство (*)
iObject.Height = Height	Установить свойство (*)
Height = iObject.GetHeight()	Получить свойство(**)
iObject.SetHeight (Height)	Установить свойство (**)

Синтаксис COM:

```
iObject->get_Height (&Height)        Получить свойство
iObject->put_Height (Height)         Установить свойство
```

Значение свойства:

Высота элемента управления (в пикселах).

ResModule – Модуль с ресурсами

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Синтаксис Automation:

```
ResModule = iObject.ResModule                Получить свойство (* )
iObject.ResModule = ResModule                Установить свойство (* )
ResModule = iObject.GetResModule()           Получить свойство(**)
iObject.SetResModule (ResModule)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ResModule (&ResModule)          Получить свойство
iObject->put_ResModule (ResModule)           Установить свойство
```

Значение свойства:

HINSTANCE (тип значения long) или полный путь к файлу (тип значения BSTR) dll модуля, в котором находятся все необходимые ресурсы.

Примечание:

Данное свойство позволяет установить или получить dll-модуль, с которого будут доставаться значки или слайды для отображения в окне.

SlideType – Тип отображения

Интерфейс...

Тип данных: тип отображения из перечисления SlideTypeEnum.

Синтаксис Automation:

```
SlideType = iObject.SlideType                Получить свойство (* )
iObject.SlideType = SlideType                Установить свойство (* )
SlideType = iObject.GetSlideType()           Получить свойство(**)
iObject.SetSlideType (SlideType)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_SlideType (&SlideType)          Получить свойство
```

iObject->put_SlideType (SlideType)	Установить свойство

Примечание:

Свойство позволяет определить тип отображаемого значения.

Width – Ширина элемента управления

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значение свойства:

Ширина элемента управления (в пикселах).

IPropertySlideBox – методы

UpdateParam – Обновить параметры окна отображения слайда

Интерфейс...

Синтаксис Automation:

void UpdateParam();

Синтаксис COM:

HRESULT UpdateParam();

Примечание:

1. Все изменения в окне отображения слайда вступят в силу только после вызова этого метода.
2. Этот метод можно использовать для обновления изображения в окне.

Интерфейс IPropertySpinEdit

Интерфейс элемента управления Панели свойств типа Поле ввода со счетчиком.

Иерархия:

IKompasAPIObject

Элемент панели свойств

IPropertyControl

IPropertySpinEdit

Примечание:

1. Данный интерфейс можно получить через коллекцию элементов управления Панели свойств IPropertyControls.
2. При нажатии на кнопку счетчика вызывается событие ksPropertyManagerNotify::ControlCommand.
3. При изменении значения параметра вызывается событие ksPropertyManagerNotify::ChangeControlValue.

IPropertySpinEdit – свойства

CheckState – Состояние опции

Интерфейс...

Тип данных: состояние опции из перечисления CheckStateEnum.

Синтаксис Automation:

```
CheckState = iObject.CheckState              Получить свойство (* )
iObject.CheckState = CheckState              Установить свойство (* )
CheckState = iObject.GetCheckState()         Получить свойство(**)
iObject.SetCheckState (CheckState)           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_CheckState (&CheckState)        Получить свойство
iObject->put_CheckState (CheckState)         Установить свойство
```

Примечание:

Это свойство влияет на состояние переключателя, если свойство WithCheck имеет значение TRUE, и на отображение текста в поле элемента управления, подробнее см. CheckStateEnum.

MaxValue – Максимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
MaxValue = iObject.MaxValue                  Получить свойство (* )
MaxValue = iObject.GetMaxValue()             Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_MaxValue (&MaxValue)          Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления:

- для типа ksControlSpinInt из ControlTypeEnum используется тип long,
- для типа ksControlSpinReal из ControlTypeEnum используется тип double.

MinValue – Минимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
MinValue = iObject.MinValue                Получить свойство (* )
MinValue = iObject.GetMinValue()           Получить свойство(**)
```

Синтаксис COM:

```
iObject->get_MinValue (&MinValue)          Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Тип данных определяется типом элемента управления:

- для типа ksControlSpinInt из ControlTypeEnum используется тип long,
- для типа ksControlSpinReal из ControlTypeEnum используется тип double.

ReadOnly – Запрет редактирования значения

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

```
ReadOnly = iObject.ReadOnly                Получить свойство (* )
iObject.ReadOnly = ReadOnly                Установить свойство (* )
ReadOnly = iObject.GetReadOnly()           Получить свойство(**)
iObject.SetReadOnly (ReadOnly)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ReadOnly (&ReadOnly)          Получить свойство
iObject->put_ReadOnly (ReadOnly)           Установить свойство
```

Значение свойства:

TRUE	- ввод нового значения с клавиатуры в поле запрещен,
FALSE	- ввод нового значения с клавиатуры в раскрывающийся список разрешен.

Step – Шаг

Интерфейс...

Тип данных: double.

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство(**)
Object.SetStep (Step)	Установить свойство (**)

Синтаксис COM:

Object->get_Step (&Step)	Получить свойство
Object->put_Step (Step)	Установить свойство

Width – Шаблон для расчета ширины элемента управления в горизонтальном положении (в символах)

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значение свойства:

Количество видимых символов в поле при горизонтальном положении Панели свойств.

Примечание:

Для расчета ширины элемента управления используется символ «0».

WithCheck – Наличие опции рядом с полем ввода

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
WithCheck = iObject.WithCheck              Получить свойство (* )
iObject.WithCheck = WithCheck              Установить свойство (* )
WithCheck = iObject.GetWithCheck()         Получить свойство(**)
iObject.SetWithCheck (WithCheck)           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_WithCheck (&WithCheck)        Получить свойство
iObject->put_WithCheck (WithCheck)         Установить свойство
```

Значение свойства:

TRUE - рядом с элементом управления выводится опция,
FALSE - рядом с элементом управления не выводится опция.

IPropertySpinEdit – методы

SetValueRange – Установить новые ограничения на значение

Интерфейс...

Синтаксис Automation:

BOOL SetValueRange (const VARIANT& minVal, const VARIANT& maxVal);

Синтаксис COM:

HRESULT SetValueRange ([in] VARIANT minVal,
[in] VARIANT maxVal,
[out, retval] VARIANT_BOOL * pVal);

Входные параметры:

minVal - минимальное значение,
maxVal - максимальное значение.

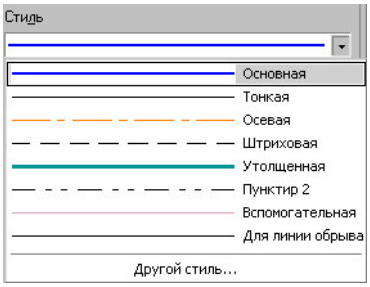
Примечание:

Тип данных максимального и минимального допустимого значения определяется типом элемента управления:

- для типа ksControlSpinInt из ControlTypeEnum используется тип long,
- для типа ksControlSpinReal из ControlTypeEnum используется тип double.

Интерфейс IPropertyStyleList

Элемент Панели свойств – Комбобокс со стилем.



Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyStyleList

Описание:

Интерфейс списка, из которого можно выбрать текущий стиль линии.

Примечания:

Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

Перечень линий, доступных для выбора, а также порядок следования линий в списке определяется настройкой фильтра линий, сделанной для текущего документа.

Кнопка **Другой стиль** позволяет выбрать стиль из внешней библиотеки, или из сформированного пользователем набора, или один из стилей, хранящихся в документе. Ее наличием можно управлять с помощью свойства AnotherStyleBtnEnable.

Можно наполнить комбобокс только нужными стилями.

IPropertyStyleList – свойства

AnotherStyleBtnEnable – Доступность кнопки Другой стиль

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AnotherStyleBtnEnable = Object.AnotherStyleBtnEnable	Получить свойство (*)

Object.AnotherStyleBtnEnable = AnotherStyleBtnEnable	Установить свойство (*)
AnotherStyleBtnEnable = Object.GetAnotherStyleBtnEnable()	Получить свойство(**)
Object.SetAnotherStyleBtnEnable(AnotherStyleBtnEnable)	Установить свойство (**)

Синтаксис COM:

Object.get_AnotherStyleBtnEnable(&AnotherStyleBtnEnable)	Получить свойство
Object.put_AnotherStyleBtnEnable(AnotherStyleBtnEnable)	Установить свойство

Свойство позволяет включать и отключать доступность кнопки **Другой стиль**.

Count – Количество элементов списка

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Object.Count = Count	Установить свойство (*)
Count = Object.GetCount()	Получить свойство(**)
Object.SetCount(Count)	Установить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство
Object.put_Count(Count)	Установить свойство

Свойство позволяет устанавливать и получать количество элементов списка.

CurrentIndex – Индекс текущего стиля

Интерфейс...

Тип данных: long

Синтаксис Automation:

CurrentIndex = Object.CurrentIndex	Получить свойство (*)
Object.CurrentIndex = CurrentIndex	Установить свойство (*)
CurrentIndex = Object.GetCurrentIndex()	Получить свойство(**)
Object.SetCurrentIndex(CurrentIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentIndex(&CurrentIndex)	Получить свойство
Object.put_CurrentIndex(CurrentIndex)	Установить свойство

Свойство позволяет устанавливать и получать индекс текущего стиля.

IPropertyStyleList – методы

Add – Добавить массив стилей в список

Интерфейс...

Синтаксис Automation:

BOOL Add(VARIANT Styles);

Синтаксис COM:

HRESULT Add(VARIANT Styles, BOOL * Result);

Входные параметры:

Styles	- список стилей.

Возвращаемое значение:

TRUE	- в случае удачи.

Примечание:

Если добавляется один стиль, его можно передать как VT_I4.

Если требуется добавить несколько стилей, то нужно создать массив VT_ARRAY | VT_I4.

ClearList – Очистить список

Интерфейс...

Синтаксис Automation:

BOOL ClearList();

Синтаксис COM:

HRESULT ClearList(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачи.

Find – Найти индекс стиля по значению

Интерфейс...

Синтаксис Automation:

long Find(long Val);

Синтаксис COM:

HRESULT Find(long Val, long * Result);

Возвращаемое значение:

индекс значения в списке - в случае удачи,
-1 - если значение не найдено.

Интерфейс IPropertyTextButton

Интерфейс элемента управления Панели свойств типа Кнопка с текстом.

Элемент панели свойств

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyTextButton

Примечание:

1. Данный интерфейс можно получить через коллекцию элементов управления Панели свойств IPropertyControls.
2. При нажатии на кнопку вызывается событие ksPropertyManagerNotify::ControlCommand.
3. Элемент управления работает как обычная кнопка со стилем BS_PUSHBUTTON.

Интерфейс IPropertyUserControl

События...

Интерфейс элемента управления Панели свойств Пользовательский элемент управления.

Иерархия:

IKompasAPIObject

IPropertyControl

IPropertyUserControl

Данный элемент управления позволяет отображать на Панели свойств нестандартные элементы управления и их наборы. В качестве вложенного нестандартного элемента управления используется элемент управления OCX. Используемый элемент управления OCX должен быть зарегистрирован в реестре Windows.

После создания элемента управления OCX библиотеке посылается событие ksPropertyUserControlNotify::CreateOCX. В этом событии передается Dispatch-интерфейс элемента управления OCX и библиотека может подписаться на события от этого элемента управления.

Перед удалением элемента управления OCX библиотеке посылается событие ksPropertyUserControlNotify::DestroyOCX. При обработке этого события библиотека должна отписаться от событий этого элемента управления.

Примечание:

Данный интерфейс можно получить через коллекцию элементов управления Панели свойств IPropertyControls.

IPropertyUserControl – свойства

AutoSize – Автоматический расчет высоты контрола. TRUE – Растягивать на всю доступную область

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSize = Object.AutoSize	Получить свойство (*)
Object.AutoSize = AutoSize	Установить свойство (*)
AutoSize = Object.GetAutoSize()	Получить свойство(**)
Object.SetAutoSize(AutoSize)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSize(&AutoSize)	Получить свойство
Object.put_AutoSize(AutoSize)	Установить свойство

FixHeight – Зафиксировать высоту элемента управления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixHeight = iObject.FixHeight	Получить свойство (*)
iObject.FixHeight = FixHeight	Установить свойство (*)
FixHeight = iObject.GetFixHeight()	Получить свойство(**)
iObject.SetFixHeight(FixHeight)	Установить свойство (**)

Синтаксис COM:

iObject->get_FixHeight(&FixHeight)	Получить свойство
iObject->put_FixHeight(FixHeight)	Установить свойство

Значение свойства:

TRUE - размер элемента управления по высоте зафиксирован и равен значению, установленному в свойстве Height - Высота элемента управления.

Примечание:

По умолчанию свойство имеет значение FALSE - размер элемента управления по высоте не зафиксирован.

FixWidth – Зафиксировать ширину элемента управления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
FixWidth = iObject.FixWidth                 Получить свойство (* )
iObject.FixWidth = FixWidth                 Установить свойство (* )
FixWidth = iObject.GetFixWidth()            Получить свойство(**)
iObject.SetFixWidth(FixWidth)               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_FixWidth(&FixWidth)            Получить свойство
iObject->put_FixWidth(FixWidth)             Установить свойство
```

Значение свойства:

TRUE - размер элемента управления по ширине зафиксирован и равен значению, установленному в свойстве Width - Ширина элемента управления.

Примечание:

По умолчанию свойство имеет значение FALSE - размер элемента управления по ширине не зафиксирован. Свойство работает при горизонтальном размещении Панели свойств.

Height – Высота элемента управления

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Height = iObject.Height                     Получить свойство (* )
iObject.Height = Height                     Установить свойство (* )
Height = iObject.GetHeight()                Получить свойство(**)
iObject.SetHeight (Height)                  Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Height (&Height)               Получить свойство
iObject->put_Height (Height)                Установить свойство
```

Значение свойства:

Высота элемента управления в пикселах.

Width – Ширина элемента управления

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Width = iObject.Width	Получить свойство (*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство(**)
iObject.SetWidth (Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width (&Width)	Получить свойство
iObject->put_Width (Width)	Установить свойство

Значение свойства:

Ширина элемента управления в пикселах.

Примечание:

Используется только для Панели свойств в горизонтальном положении. В вертикальном определяется шириной самой Панели свойств.

IPropertyUserControl – методы

SetOCXControl – Установить progID элемента управления OCX

Интерфейс...

Синтаксис Automation:

void SetOCXControl (LPCTSTR progID);

Синтаксис COM:

HRESULT SetOCXControl ([in] BSTR progID);

Входные параметры:

progID	- строка progID элемента управления OCX.

Примечание:

OCX контрол должен быть зарегистрирован в реестре Windows.

Интерфейс IPropertyManager

Интерфейс событий...

Справка системы КОМПАС...

kompas.chm::/CM_2DPROCESSBAR_VISIBLE.htm

Интерфейс Панели свойств.

Иерархия:

IKompasAPIObject

IPropertyManager

КОМПАС-3D использует стандартную Панель свойств, которая позволяет задавать параметры процессов. Библиотеки КОМПАС могут добавлять вкладки в стандартную Панель свойств и создавать собственные Панели свойств. Их состав формируется при подключении библиотеки или выполнении библиотечной команды. После отключения библиотеки или завершения работы команды созданные ими элементы управления исчезают. Сигналом для этого служат изменения активности документов или процессов. Чтобы отслеживать эти изменения активности, библиотека должна быть подписана на соответствующие события. При изменении состояния элементов управления Панели свойств, созданных библиотекой, система передает управление библиотеке посредством вызовов событий.

Для стандартной Панели свойств все свойства данного интерфейса доступны только по чтению. Библиотека может управлять состоянием только собственной Панели свойств либо вкладок на стандартной. Остальные вкладки и их элементы управления будут недоступны для изменения.

Панель свойств или вкладки на стандартной Панели свойств, созданные библиотекой, будут удалены после освобождения библиотекой их интерфейса. Чтобы эти элементы управления были доступны, библиотека должна держать ссылку на интерфейс Панели свойств все время от начала до завершения ее использования.

Панель свойств библиотеки может быть расположена только слева, справа или находиться в плавающем состоянии.

Примечание:

Создать панель свойств и получить данный интерфейс можно от интерфейса приложения IApplication с помощью свойства IApplication::CreatePropertyManager.

IPropertyManager – свойства

AutoHideMode – Свернутость Панели свойств

Функция не поддерживается

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

```
AutoHideMode = iObject.AutoHideMode;          Получить свойство (* )
iObject.AutoHideMode = AutoHideMode;          Установить свойство (* )
AutoHideMode =                                Получить свойство(**)
iObject.GetAutoHideMode();
iObject.SetAutoHideMode( AutoHideMode );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_AutoHideMode( &AutoHideMode );     Получить свойство
iObject->put_AutoHideMode( AutoHideMode );      Установить свойство
```

Значения свойства:

```
TRUE    - панель свернута,
FALSE   - панель развернута.
```

Примечание:

Позволяет получать и устанавливать свойство, определяющее свернута панель или нет.

Caption – Заголовок Панели свойств

Интерфейс...

Тип данных: BSTR (строка).

Синтаксис Automation:

```
Caption = iObject.Caption           Получить свойство (* )
iObject.Caption = Caption           Установить свойство (* )
Caption = iObject.GetCaption()      Получить свойство(**)
iObject.SetCaption (Caption)        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Caption (&Caption)     Получить свойство
iObject->put_Caption (Caption)      Установить свойство
```

Примечание:

С помощью данного свойства можно получить и изменить заголовок Панели свойств.

EnterButtonIconType – Тип иконки для кнопки Создать

Интерфейс...

Тип данных: из перечисления ksEnterButtonIconTypeEnum

Синтаксис Automation:

```
EnterButtonIconType = Object.EnterButtonIconType        Получить свойство (* )
Object.EnterButtonIconType = EnterButtonIconType        Установить свойство (* )
EnterButtonIconType = Object.GetEnterButtonIconType()   Получить свойство(**)
```

Object.SetEnterButtonIconType(EnterButtonIconType)	Установить свойство (**)

Синтаксис COM:

Object.get_EnterButtonIconType(&EnterButtonIconType)	Получить свойство
Object.put_EnterButtonIconType(EnterButtonIconType)	Установить свойство

Label – Заголовок 2 панели свойств

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Label = Object.Label	Получить свойство (*)
Object.Label = Label	Установить свойство (*)
Label = Object.GetLabel()	Получить свойство(**)
Object.SetLabel(Label)	Установить свойство (**)

Синтаксис COM:

Object.get_Label(&Label)	Получить свойство
Object.put_Label(Label)	Установить свойство

Layout – Положение Панели свойств (вверху, внизу, слева, справа, плавает)

Интерфейс...

Тип данных: положение Панели свойств из перечисления PropertyManagerLayout.

Синтаксис Automation:

Layout = iObject.Layout	Получить свойство (*)
iObject.Layout = Layout	Установить свойство (*)
Layout = iObject.GetLayout()	Получить свойство(**)
iObject.SetLayout (Layout)	Установить свойство (**)

Синтаксис COM:

iObject->get_Layout (&Layout)	Получить свойство
iObject->put_Layout (Layout)	Установить свойство

Примечание:

С помощью данного свойства можно получить и изменить положение Панели свойств.

PropertyTabs – Коллекция закладок Панели свойств

Интерфейс...

Тип данных: указатель на интерфейс IPropertyTabs.

Синтаксис Automation:

PropertyTabs = iObject.PropertyTabs	Получить свойство (*)
PropertyTabs = iObject.GetPropertyTabs()	Получить свойство(**)

Синтаксис COM:

iObject->get_PropertyTabs (&PropertyTabs)	Получить свойство

Примечание:

1. Свойство только для чтения.
2. Позволяет получить доступ к вкладкам Панели свойств.

ResModule – Модуль с описанием пользовательской спецпанели

Интерфейс...

Тип данных: VARIANT (LONG_PTR или BSTR).

Синтаксис Automation:

ResModule = Object.ResModule	Получить свойство (*)
Object.ResModule = ResModule	Установить свойство (*)
ResModule = Object.GetResModule()	Получить свойство(**)
Object.SetResModule(ResModule)	Установить свойство (**)

Синтаксис COM:

Object.get_ResModule(&ResModule)	Получить свойство
Object.put_ResModule(ResModule)	Установить свойство

Значение свойства:

HINSTANCE (тип значения LONG_PTR)	- полный путь к файлу (тип значения BSTR) dll-модуля, в котором находятся все необходимые ресурсы.

Примечание:

1. HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

   Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

2. Данное свойство позволяет установить или получить dll-модуль, в котором находится описание пользовательской спецпанели, (пример для пользовательских кнопок см. IPropertyManager::SpecToolbarEx).
3. Свойство не используется при работе со шрифтовыми иконками

SpecToolbar – Специальная панель

Интерфейс...

Тип данных: тип предопределенной спецпанели для Панели свойств из перечисления SpecPropertyToolBarEnum.

Синтаксис Automation:

```
SpecToolbar = iObject.SpecToolbar          Получить свойство (* )
iObject.SpecToolbar = SpecToolbar          Установить свойство (* )
SpecToolbar = iObject.GetSpecToolbar()     Получить свойство(**)
iObject.SetSpecToolbar (SpecToolbar)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_SpecToolbar (&SpecToolbar)    Получить свойство
iObject->put_SpecToolbar (SpecToolbar)     Установить свойство
```

Примечание:

С помощью данного свойства можно получить и изменить тип предопределенной спецпанели для Панели свойств.

SpecToolbarEx – Пользовательская спецпанель

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
SpecToolbarEx = Object.SpecToolbarEx       Получить свойство (* )
Object.SpecToolbarEx = SpecToolbarEx       Установить свойство (* )
SpecToolbarEx = Object.GetSpecToolbarEx()  Получить свойство(**)
Object.SetSpecToolbarEx( SpecToolbarEx )   Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SpecToolbarEx( &SpecToolbarEx )     Получить свойство
Object.put_SpecToolbarEx(  SpecToolbarEx )     Установить свойство
```

Примечание:

1. Данное свойство является расширением свойства IPropertyManager::SpecToolbar.
2. Свойство позволяет установить или получить идентификатор спецпанели.

Может быть установлен идентификатор из перечисления SpecPropertyToolBarEnum (предопределенных спецпанелей для панели свойств) либо задан идентификатор ресурса пользовательской спецпанели, уникальный в рамках библиотеки.

3. Свойство SpecToolbarEx используется совместно со свойством IPropertyManager::ResModule.
4. Чтобы описать пользовательскую спецпанель, нужно:

   4.1. Объявить уникальный идентификатор спецпанели в h файле

   #define ITB_SPEC_TOOLBAR 3000

   #define END_OF_RESOURCE_TABLE 0xffff

   4.2. В ресурсном файле описать RCDATA панели и пользовательские кнопки

   rc2-файл:

```
ITB_SPEC_TOOLBAR RCDATA
{
  1 //pbEnter
  2 //pbEsc
  3 //pbHelpсправка
  20 //пользовательская кнопка

  END_OF_RESOURCE_TABLE
}
```

   В примере для пользовательской спецпанели определено 4 кнопки.

   1, 2, 3 - предопределенные кнопки из перечисления SpecPropertyToolBarEnum

   20 - пользовательская кнопка

   //битмап для пользовательской кнопки спецпанели

   20 ICON DISCARDABLE "res\\st_reduc.ico" // CREATE процесса настройки фильтра для выбора объектов отчета

   //tips\hint для пользовательской кнопки спецпанели

```
STRINGTABLE LOADONCALL MOVEABLE DISCARDABLE
BEGIN
  20 "Для тестирования пользовательской кнопки\nПользовательская кнопка"
END
```

StatusMessage – Строка подсказки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
StatusMessage = Object.StatusMessage          Получить свойство (* )
```

Object.StatusMessage = StatusMessage	Установить свойство (*)
StatusMessage = Object.GetStatusMessage()	Получить свойство(**)
Object.SetStatusMessage(StatusMessage)	Установить свойство (**)

Синтаксис COM:

Object.get_StatusMessage(&StatusMessage)	Получить свойство
Object.put_StatusMessage(StatusMessage)	Установить свойство

Visible – Видимость Панели свойств

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство(**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

С помощью данного свойства можно управлять видимостью Панели свойств.

IPropertyManager – методы

AddSetupMenuCommand – Добавить пункт в меню настроек процесса

Интерфейс...

Синтаксис Automation:

BOOL AddSetupMenuCommand(BSTR Title, long Command, BOOL Checable);

Синтаксис COM:

HRESULT AddSetupMenuCommand(BSTR Title, long Command, BOOL Checable, BOOL * Result);

Возвращаемое значение:

TRUE	- успешное завершение,
FALSE	- в случае неудачи.

Входные параметры:

Title - заголовок,
Command - идентификатор команды,
Checable - FALSE -выполняемая команда, TRUE - флаг состояния.

AddSpecToolbarButton – Добавить кнопку в спецпанель

Интерфейс...

Синтаксис Automation:

BOOL AddSpecToolbarButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont);

Синтаксис COM:

HRESULT AddSpecToolbarButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont, BOOL * Result);

Возвращаемое значение:

TRUE - успешное завершение,
FALSE - в случае неудачи.

Входные параметры:

BtnID - идентификатор кнопки,
Bmp - идентификатор картинки или символа шрифта или путь к файлу,
Tips - имя кнопки,
IconFont - имя шрифта.

Примечание:

1. Для использования иконок из шрифта Компас нужно передать параметр IconFont == пустой строке.
2. Для использования шрифта библиотеки, описанного в манифесте библиотеки, нужно передать строчку, полученную методом IProceduresLibrary::IconsFont.
3. Для использования картинок из ресурсов требуется установить свойство IPropertyManager::ResModule.
4. Для использования шрифтовых иконок свойство IPropertyManager::ResModule устанавливаться не должно.

ClearSpecToolbar – Очистить спецпанель

Интерфейс...

Синтаксис Automation:

BOOL ClearSpecToolbar();

Синтаксис COM:

HRESULT ClearSpecToolbar(BOOL * Result);

Возвращаемое значение:

TRUE - успешное завершение.

GetGabaritRect – Получить габаритный прямоугольник Панели свойств

Интерфейс...

Синтаксис Automation:

void GetGabaritRect (long* left,
long* top,
long* right,
long* bottom);

Синтаксис COM:

HRESULT GetGabaritRect ([out] long* left,
[out] long* top,
[out] long* right,
[out] long* bottom);

Выходные параметры:

left, top - координата верхней левой точки,
right, bottom - координата нижней правой точки.

Примечание:

Метод позволяет получить габаритный прямоугольник Панели свойств. Точки задаются в относительных координатах окна КОМПАС.

HideTabs – Скрыть вкладки и их содержимое на Панели свойств

Интерфейс...

Синтаксис Automation:

BOOL HideTabs();

Синтаксис COM:

HRESULT HideTabs ([out, retval] VARIANT_BOOL * pVal);

Возвращаемое значение:

TRUE - успешное завершение,
FALSE - в случае неудачи.

Примечание:

Метод позволяет скрыть вкладки и их содержимое на Панели свойств.

RepeatCommand – Повторить команду (посылает повторно событие панели свойств)

Интерфейс...

Синтаксис Automation:

BOOL RepeatCommand();

Синтаксис COM:

HRESULT RepeatCommand(BOOL * Result);

Возвращаемое значение:

TRUE	- успешное завершение,
FALSE	- в случае неудачи.

SetGabaritRect – Установить габаритный прямоугольник Панели свойств

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

void SetGabaritRect (long left,
long top,
long right,
long bottom);

Синтаксис COM:

HRESULT SetGabaritRect ([in] long left,
[in] long top,
[in] long right,
[in] long bottom);

Входные параметры:

left, top	- координата верхней левой точки,
right, bottom	- координата нижней правой точки.

Примечание:

Метод позволяет установить габаритный прямоугольник Панели свойств. Точки задаются в относительных координатах окна КОМПАС.

SetSetupMenuCommandState – Установить состояние команды меню настроек процесса

Интерфейс...

Синтаксис Automation:

BOOL SetSetupMenuCommandState(long Command, BOOL Visible, BOOL Enable, BOOL Checked);

Синтаксис COM:

HRESULT SetSetupMenuCommandState(long Command, BOOL Visible, BOOL Enable, BOOL Checked, BOOL * Result);

Возвращаемое значение:

TRUE	- успешное завершение,
FALSE	- в случае неудачи.

Входные параметры:

Command	- идентификатор команды,
Visible	- видимость,
Enable	- доступность,
Checked	- признак выбора.

ShowTabs – Отобразить вкладки и их содержимое на Панели свойств

Интерфейс...

Синтаксис Automation:

BOOL ShowTabs();

Синтаксис COM:

HRESULT ShowTabs ([out, retval] VARIANT_BOOL * pVal);

Возвращаемое значение:

TRUE	- успешное завершение,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет отобразить вкладки и их содержимое на Панели свойств.

UpdateTabs – Обновить вкладки и их содержимое на Панели свойств

Интерфейс...

Синтаксис Automation:

BOOL UpdateTabs();

Синтаксис COM:

HRESULT UpdateTabs ([out, retval] VARIANT_BOOL * pVal);

Возвращаемое значение:

TRUE	- успешное завершение,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет обновить вкладки и их содержимое на Панели свойств.

Интерфейс IPropertyTab

Интерфейс вкладки Панели свойств.

Иерархия:

IKompasAPIObject

IPropertyTab

Примечание:

Данный интерфейс может быть получен от интерфейса коллекции закладок панели свойств IPropertyTabs с помощью свойств IPropertyTabs::Item, IPropertyTabs::Active и метода IPropertyTabs::Add.

IPropertyTab – свойства

Active – Активность закладки Панели свойств

Функция не поддерживается

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Active = iObject.Active                 Получить свойство (* )
iObject.Active = Active                 Установить свойство (* )
Active = iObject.GetActive()            Получить свойство(**)
iObject.SetActive (Active)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Active (&Active)           Получить свойство
iObject->put_Active (Active)            Установить свойство
```

Примечание:

Свойство позволяет получить и изменить активность вкладки Панели свойств.

Caption – Заголовок вкладки Панели свойств

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

```
Caption = iObject.Caption               Получить свойство (* )
iObject.Caption = Caption               Установить свойство (* )
Caption = iObject.GetCaption()          Получить свойство(**)
iObject.SetCaption (Caption)            Установить свойство (**)
```

Синтаксис COM:

iObject->get_Caption (&Caption)	Получить свойство
iObject->put_Caption (Caption)	Установить свойство

Примечание:

Свойство позволяет получить и изменить заголовок вкладки Панели свойств.

Expanded – Признак развернутой закладки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Expanded = Object.Expanded	Получить свойство (*)
Object.Expanded = Expanded	Установить свойство (*)
Expanded = Object.GetExpanded()	Получить свойство(**)
Object.SetExpanded(Expanded)	Установить свойство (**)

Синтаксис COM:

Object.get_Expanded(&Expanded)	Получить свойство
Object.put_Expanded(Expanded)	Установить свойство

Image – Растровый рисунок на вкладке

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Значения свойства:

HINSTANCE (тип значения long)	- полный путь к файлу (тип значения BSTR) dll модуля в котором находятся все необходимые ресурсы.

Синтаксис Automation:

Image = iObject.Image	Получить свойство (*)
iObject.Image = Image	Установить свойство (*)
Image = iObject.GetImage()	Получить свойство(**)
iObject.SetImage(Image)	Установить свойство (**)

Синтаксис COM:

iObject->get_Image(&Image)	Получить свойство
iObject->put_Image(Image)	Установить свойство

Примечание:

Свойство позволяет получить или установить битмап на закладке.

Битмап можно задать в виде идентификатора в файле ресурсов (тип данных long) или полного пути к bmp-файлу значка (тип данных BSTR).

PropertyControls – Коллекция элементов управления на вкладке

Интерфейс...

Тип данных: указатель на интерфейс IPropertyControls.

Синтаксис Automation:

PropertyControls = iObject.PropertyControls	Получить свойство (*)
PropertyControls = iObject.GetPropertyControls()	Получить свойство(**)

Синтаксис COM:

iObject->get_PropertyControls (&PropertyControls)	Получить свойство

Примечание:

Свойство доступно только для чтения. Позволяет получить коллекцию элементов управления вкладки Панели свойств.

ResModule – Растровый рисунок на вкладке

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT (long или BSTR).

Значения свойства:

HINSTANCE (тип значения long)	- полный путь к файлу (тип значения BSTR) dll модуля, в котором находятся все необходимые ресурсы.

Синтаксис Automation:

ResModule = iObject.ResModule	Получить свойство (*)
iObject.ResModule = ResModule	Установить свойство (*)
ResModule = iObject.GetResModule()	Получить свойство(**)
iObject.SetResModule(ResModule)	Установить свойство (**)

Синтаксис COM:

iObject->get_ResModule(&ResModule)	Получить свойство
iObject->put_ResModule(ResModule)	Установить свойство

Примечание:

Свойство позволяет получить или установить dll-модуль, в котором находится рисунок для кнопки.

Visible – Видимость вкладки Панели свойств

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible =iObject.Visible	Получить свойство(**)
iObject.SetVisible	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

Свойство позволяет получить и изменить видимость вкладки Панели свойств.

Интерфейс IPropertyControl1

Справка системы КОМПАС...

`kompas.chm::/CM_2DPROCESSBAR_VISIBLE.htm`

Дополнительный интерфейс элемента управления Панели свойств.

Иерархия:

IKompasAPIObject

IPropertyControl1

Примечание:

Данный интерфейс можно получить у элемента управления Панели свойств IPropertyControl посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IPropertyControl1 – свойства

AdditionButtonNeedMouseEnterLeaveMessages – Необходимость получать сообщения о наведении/

уходе курсора мыши на дополнительных кнопках от данного контрола

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AdditionButtonNeedMouseEnterLeaveMessages = Object.AdditionButtonNeedMouseEnterLeaveMessages	Получить свойство(*)
Object.AdditionButtonNeedMouseEnterLeaveMessages = AdditionButtonNeedMouseEnterLeaveMessages	Установить свойство (*)
AdditionButtonNeedMouseEnterLeaveMessages = Object.GetAdditionButtonNeedMouseEnterLeaveMessages()	Получить свойство (**)
Object.SetAdditionButtonNeedMouseEnterLeaveMessages(AdditionButtonNeedMouseEnterLeaveMessages)	Установить свойство (**)

Синтаксис COM:

Object.get_AdditionButtonNeedMouseEnterLeaveMessages(&AdditionButtonNeedMouseEnterLeaveMessages)	Получить свойство(*)
Object.put_AdditionButtonNeedMouseEnterLeaveMessages(AdditionButtonNeedMouseEnterLeaveMessages)	Установить свойство (*)

Входные параметры:

long BtnID	- пользовательский идентификатор кнопки.

Примечание:

Включение свойства позволит обрабатывать событие ksProcess2DNotify::GetMouseEnterLeavePoint.

Это позволит подсвечивать точку в поле 2D документа, положение которой зависит от контрола под курсором.

AdditionButtonChecked – Состояние дополнительной кнопки (нажата/отжата)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AdditionButtonChecked = Object.AdditionButtonChecked(BtnID)	Получить свойство (*)
Object.AdditionButtonChecked(BtnID) = AdditionButtonChecked	Установить свойство (*)
AdditionButtonChecked = Object.GetAdditionButtonChecked(BtnID)	Получить свойство(**)

Object.SetAdditionButtonChecked(BtnID, AdditionButtonChecked) — Установить свойство (**)

Синтаксис COM:

Object.get_AdditionButtonChecked(BtnID, &AdditionButtonChecked) — Получить свойство

Object.put_AdditionButtonChecked(BtnID, AdditionButtonChecked) — Установить свойство

Входные параметры:

BtnID - идентификатор кнопки.

AdditionButtonEnable – Состояние (доступность) дополнительной кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AdditionButtonEnable = Object.AdditionButtonEnable(BtnID) — Получить свойство (*)

Object.AdditionButtonEnable(BtnID) = AdditionButtonEnable — Установить свойство (*)

AdditionButtonEnable = Object.GetAdditionButtonEnable(BtnID) — Получить свойство(**)

Object.SetAdditionButtonEnable(BtnID, AdditionButtonEnable) — Установить свойство (**)

Синтаксис COM:

Object.get_AdditionButtonEnable(BtnID, &AdditionButtonEnable) — Получить свойство

Object.put_AdditionButtonEnable(BtnID, AdditionButtonEnable) — Установить свойство

Входные параметры:

BtnID - идентификатор кнопки.

AdditionButtonVisible – Видимость дополнительной кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AdditionButtonVisible = Object.AdditionButtonVisible(BtnID)	Получить свойство (*)
Object.AdditionButtonVisible(BtnID) = AdditionButtonVisible	Установить свойство (*)
AdditionButtonVisible = Object.GetAdditionButtonVisible(BtnID)	Получить свойство(**)
Object.SetAdditionButtonVisible(BtnID, AdditionButtonVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_AdditionButtonVisible(BtnID, &AdditionButtonVisible)	Получить свойство
Object.put_AdditionButtonVisible(BtnID, AdditionButtonVisible)	Установить свойство

Входные параметры:

BtnID - идентификатор кнопки.

HyperLinkNameStyle – Отображать имя контрола в виде ссылки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HyperLinkNameStyle = Object.HyperLinkNameStyle	Получить свойство (*)
Object.HyperLinkNameStyle = HyperLinkNameStyle	Установить свойство (*)
HyperLinkNameStyle = Object.GetHyperLinkNameStyle()	Получить свойство(**)
Object.SetHyperLinkNameStyle(HyperLinkNameStyle))	Установить свойство (**)

Синтаксис COM:

Object.get_HyperLinkNameStyle(&HyperLinkNameStyle)	Получить свойство
Object.put_HyperLinkNameStyle(HyperLinkNameStyle)	Установить свойство

Image – Дополнительный битмап контрол

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Image = iObject.Image	Получить свойство (*)
iObject.Image = Image	Установить свойство (*)
Image = iObject.GetImage()	Получить свойство(**)
iObject.SetImage (Image)	Установить свойство (**)

Синтаксис COM:

iObject->get_Image (&Image)	Получить свойство
iObject->put_Image (Image)	Установить свойство

Значения свойства:

- HINSTANCE (тип значения LONG_PTR).

NeedMouseEnterLeaveMessages – Необходимость получать сообщения о наведении/уходе курсора мыши на поле редактирования контрола

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NeedMouseEnterLeaveMessages = Object.NeedMouseEnterLeaveMessages	Получить свойство(*)
Object.NeedMouseEnterLeaveMessages = NeedMouseEnterLeaveMessages	Установить свойство (*)
NeedMouseEnterLeaveMessages = Object.GetNeedMouseEnterLeaveMessages()	Получить свойство (**)
Object.SetNeedMouseEnterLeaveMessages(NeedMouseEnterLeaveMessages)	Установить свойство (**)

Синтаксис COM:

Object.get_NeedMouseEnterLeaveMessages(&NeedMouseEnterLeaveMessages)	Получить свойство(*)
Object.put_NeedMouseEnterLeaveMessages(NeedMouseEnterLeaveMessages)	Установить свойство (*)

Примечание:

Включение свойства позволит обрабатывать событие ksProcess2DNotify::GetMouseEnterLeavePoint.

Это позволит подсвечивать точку в поле 2D документа, положение которой зависит от контрола под курсором.

PredefineNumber – Идентификатор предопределённого ввода

Интерфейс...

Тип данных: long

Синтаксис Automation:

predefineNumber = iKompasObject.PredefineNumber	Получить свойство (*)
iKompasObject.PredefineNumber = predefineNumber	Установить свойство (*)
predefineNumber = iKompasObject.GetPredefineNumber()	Получить свойство(**)
iKompasObject.SetPredefineNumber(predefineNumber)	Установить свойство (**)

Примечание:

1. Позволяет получить и установить идентификатор предопределённого ввода для элементов управления, содержащих окно ввода, например, поле ввода IPropertyEdit.
2. Значение по умолчанию = -1.
3. Если идентификатор задан (положительное число), то он должен быть уникальным в пределах коллекции элементов управления конкретной вкладки.
4. Работает только на запущенном процессе. При нажатии клавиш клавиатуры из коллекции выбирается элемент управления с наименьшим идентификатором предопределённого ввода (среди отображаемых) элементов управления и ему передается фокус. При подтверждении ввода ищется следующий индекс предопределённого ввода и запоминается для последующих нажатий клавиш. Если пользователь отказывается от ввода, например нажимает клавишу *Esc*, то индекс элемента управления предопределённого ввода, которому будут передаваться значения, не меняется.
5. Задать значение свойства можно только до запуска процесса.

ResModule – Модуль с ресурсом битмапа

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ResModule = iObject.ResModule	Получить свойство (*)
iObject.ResModule = ResModule	Установить свойство (*)

ResModule = iObject.GetResModule() Получить свойство(**)
iObject.SetResModule (ResModule) Установить свойство (**)

Синтаксис COM:

iObject->get_ResModule (&ResModule) Получить свойство
iObject->put_ResModule (ResModule) Установить свойство

Значения свойства:

- HINSTANCE (тип значения LONG_PTR).

Примечание:

Данное свойство позволяет установить или получить dll-модуль, в котором находится описание пользовательской спецпанели, рисунки для пользовательских кнопок.

UserMenu – Дескриптор пользовательского меню

Интерфейс...

Тип данных: OLE_HANDLE

Синтаксис Automation:

UserMenu = iObject.UserMenu Получить свойство (*)
iObject.UserMenu = UserMenu Установить свойство (*)
UserMenu = iObject.GetUserMenu() Получить свойство(**)
iObject.SetUserMenu (UserMenu) Установить свойство (**)

Синтаксис COM:

iObject->get_UserMenu (&UserMenu) Получить свойство
iObject->put_UserMenu (UserMenu) Установить свойство

Значения свойства:

Дескриптор пользовательского контекстного меню на элементе управления.

Примечание:

Если на элементе управления уже было меню, то в случае задания оно заменяется на пользовательское. Дескриптор должен создаваться в библиотеке и быть валидным до конца процесса. Деструктурироваться дескриптор должен на стороне библиотеки по окончанию процесса. После того как меню вывелось и пользователь выбрал какой-то пункт генерируется новое событие - prUserMenuCommand.

IPropertyControl1 – методы

AddAdditionButton – Добавить дополнительную кнопку

Интерфейс...

Синтаксис Automation:

BOOL AddAdditionButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont);

Синтаксис COM:

HRESULT AddAdditionButton(long BtnID, VARIANT Bmp, BSTR Tips, BSTR IconFont, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BtnID	- идентификатор кнопки,
Bmp	- идентификатор иконки из шрифта или идентификатор картинки из ресурсов,
Tips	- текст подсказки для кнопки,
IconFont	- имя библиотечного шрифта.

AddAdditionCheckButton – Добавить дополнительную кнопку (checkbox)

Интерфейс...

Синтаксис Automation:

BOOL AddAdditionCheckButton(long BtnID, VARIANT BmpChecked, VARIANT BmpUnChecked, VARIANT BmpUndefine, BSTR Tips, BSTR IconFont);

Синтаксис COM:

HRESULT AddAdditionCheckButton(long BtnID, VARIANT BmpChecked, VARIANT BmpUnChecked, VARIANT BmpUndefine, BSTR Tips, BSTR IconFont, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BtnID	- идентификатор кнопки,
BmpChecked	- идентификатор иконки из шрифта или идентификатор картинки из ресурсов для нажатого состояния,
BmpUnChecked	- идентификатор иконки из шрифта или идентификатор картинки из ресурсов для отжатого состояния

BmpUndefine	- идентификатор иконки из шрифта или идентификатор картинки из ресурсов для неопределенного состояния,
Tips	- подсказка для кнопки,
IconFont	- имя шрифта.

GetHWND – Получить Handle окна

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

OLE_HANDLE GetHWND();

Синтаксис COM:

HRESULT GetHWND(OLE_HANDLE * Result);

Возвращаемое значение:

- дескриптор окна.

Интерфейс IPropertyPoint3D

Интерфейс элемента Панели свойств Точка 3D.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyPoint3D

Примечания:

1. Интерфейс позволяет добавить элемент Панели свойств - точка 3D.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyPoint3D – свойства

Coordinate – Параметры координаты по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IPropertyEdit

Синтаксис Automation:

Coordinate = Object.Coordinate(Index)	Получить свойство (*)
Coordinate = Object.GetCoordinate(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Coordinate(Index, &Coordinate);	Получить свойство
Object.put_Coordinate(Index, Coordinate));	Установить свойство

Входные параметры:

Index	- индекс координаты.

CoordinateState – Состояние координаты по индексу

Интерфейс...

Тип данных: из перечисления CheckStateEnum

Синтаксис Automation:

CoordinateState = Object.CoordinateState(Index)	Получить свойство (*)
Object.CoordinateState(Index) = CoordinateState	Установить свойство (*)
CoordinateState = Object.GetCoordinateState(Index)	Получить свойство(**)
Object.SetCoordinateState(Index, CoordinateState)	Установить свойство (**)

Синтаксис COM:

Object.get_CoordinateState(Index, &CoordinateState)	Получить свойство
Object.put_CoordinateState(Index, CoordinateState)	Установить свойство

Входные параметры:

Index	- индекс координаты.

CoordinateValue – Значение координаты по индексу

Интерфейс...

Тип данных: double

Синтаксис Automation:

CoordinateValue = Object.CoordinateValue(Index)	Получить свойство (*)
Object.CoordinateValue(Index) = CoordinateValue	Установить свойство (*)
CoordinateValue = Object.GetCoordinateValue(Index)	Получить свойство(**)
Object.SetCoordinateValue(Index, CoordinateValue)	Установить свойство (**)

Синтаксис COM:

Object.get_CoordinateValue(Index, &CoordinateValue)	Получить свойство
Object.put_CoordinateValue(Index, CoordinateValue)	Установить свойство

Входные параметры:

Index - индекс координаты.

Point3DType – Тип системы координат точки

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

Point3DType = Object.Point3DType	Получить свойство (*)
Object.Point3DType = Point3DType	Установить свойство (*)
Point3DType = Object.GetPoint3DType()	Получить свойство(**)
Object.SetPoint3DType(Point3DType)	Установить свойство (**)

Синтаксис COM:

Object.get_Point3DType(&Point3DType)	Получить свойство
Object.put_Point3DType(Point3DType)	Установить свойство

Интерфейс IPropertyEditCheckBox

Интерфейс элемента управления Панели свойств типа «Поле с чекбоксом в виде картинки»

Элемент панели свойств

Иерархия:

IDispatch

IKompasAPIObject

IPropertyEditCheckBox

Данный интерфейс можно получить, используя коллекцию элементов управления Панели свойств IPropertyControls.

IPropertyEditCheckBox – свойства

CheckButton – Состояние кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CheckButton = Object.CheckButton	Получить свойство (*)
Object.CheckButton = CheckButton	Установить свойство (*)
CheckButton = Object.GetCheckButton()	Получить свойство(**)
Object.SetCheckButton(CheckButton)	Установить свойство (**)

Синтаксис COM:

Object.get_CheckButton(&CheckButton)	Получить свойство
Object.put_CheckButton(CheckButton)	Установить свойство

Значения свойства:

TRUE	- включено,
FALSE	- отключено.

DefaultValue – Значение по умолчанию

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DefaultValue = Object.DefaultValue	Получить свойство (*)
Object.DefaultValue = DefaultValue	Установить свойство (*)
DefaultValue = Object.GetDefaultValue()	Получить свойство(**)
Object.SetDefaultValue(DefaultValue)	Установить свойство (**)

Синтаксис COM:

Object.get_DefaultValue(&DefaultValue	Получить свойство
Object.put_DefaultValue(DefaultValue)	Установить свойство

Примечание:

Свойство задает строку приглашения отображаемую в пустом регистраторе.

EnableCheckButton – Состояние доступности кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableCheckButton = Object.EnableCheckButton                Получить свойство (* )
Object.EnableCheckButton = EnableCheckButton                Установить свойство (* )
EnableCheckButton = Object.GetEnableCheckButton()           Получить свойство(**)
Object.SetEnableCheckButton( EnableCheckButton )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableCheckButton( &EnableCheckButton )    Получить свойство
Object.put_EnableCheckButton(  EnableCheckButton )    Установить свойство
```

Значения свойства:

```
TRUE        - доступна,
FALSE       - недоступна.
```

EnableDeleteValue – Признак возможности удаления значения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
EnableDeleteValue = Object.EnableDeleteValue        Получить свойство (* )
Object.EnableDeleteValue = EnableDeleteValue        Установить свойство (* )
EnableDeleteValue =                                 Получить свойство(**)
Object.GetEnableDeleteValue()
Object.SetEnableDeleteValue( EnableDeleteValue )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EnableDeleteValue( &EnableDeleteValue )    Получить свойство
Object.put_EnableDeleteValue(  EnableDeleteValue )    Установить свойство
```

IsLinkValue – Признак ссылочного содержимого

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsLinkValue = Object.IsLinkValue                    Получить свойство (* )
Object.IsLinkValue = IsLinkValue                    Установить свойство (* )
IsLinkValue = Object.GetIsLinkValue()               Получить свойство(**)
Object.SetIsLinkValue( IsLinkValue )                Установить свойство (**)
```

Синтаксис COM:

Object.get_IsLinkValue(&IsLinkValue	Получить свойство
Object.put_IsLinkValue(IsLinkValue)	Установить свойство

Missing – Потерянный объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Missing = Object.Missing	Получить свойство (*)
Object.Missing = Missing	Установить свойство (*)
Missing = Object.GetMissing()	Получить свойство(**)
Object.SetMissing(Missing)	Установить свойство (**)

Синтаксис COM:

Object.get_Missing(&Missing)	Получить свойство
Object.put_Missing(Missing)	Установить свойство

IPropertyEditCheckBox – методы

SetCustomBitmaps – Установить собственное изображение кнопки в разных состояниях

Интерфейс...

Синтаксис Automation:

void SetCustomBitmaps(VARIANT IdUnchecked,
VARIANT IdChecked,
VARIANT IdIndeterminate,
VARIANT HInstance);

Синтаксис COM:

HRESULT SetCustomBitmaps(VARIANT IdUnchecked,
VARIANT IdChecked,
VARIANT IdIndeterminate,
VARIANT HInstance,
void * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

IdUnchecked	- значок ненажатого состояния,

IdChecked	- значок нажатого состояния,
IdIndeterminate	- значок неопределенного состояния,
HInstance	- dll-модуль с ресурсами значков.

Примечания:

1. Значки могут быть заданы в виде идентификаторов в ресурсе dll-модуля (тип данных long) или в виде полного пути к bmp-файлу значка (тип данных BSTR). Обязательным является задание значка для ненажатого состояния idUnchecked, остальные могут быть не заданы.
2. Все значения переменных должны быть одного типа (или long, или BSTR).
3. Если значки задаются в виде идентификатора, то обязательно должен быть задан dll-модуль, в ресурсах которого лежат эти значки. Модуль с ресурсами значков hInstance может быть задан в виде HINSTANCE (тип значения long) или полного пути к файлу (тип значения BSTR) dll-модуля.

Интерфейс IPropertyPreviewText

Интерфейс элемента Панели свойств Предпросмотр текста.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyPreviewText

Примечания:

1. Интерфейс позволяет добавить элемент Панели свойств - Предпросмотр текста.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyPreviewText – свойства

DoubleSize – Двойной размер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DoubleSize = Object.DoubleSize	Получить свойство (*)
Object.DoubleSize = DoubleSize	Установить свойство (*)
DoubleSize = Object.GetDoubleSize()	Получить свойство(**)
Object.SetDoubleSize(DoubleSize)	Установить свойство (**)

Синтаксис COM:

Object.get_DoubleSize(&DoubleSize) Получить свойство
Object.put_DoubleSize(DoubleSize) Установить свойство

PreviewText – Текст

Интерфейс...

Тип данных: Указатель на интерфейс IText

Синтаксис Automation:

PreviewText = Object.PreviewText Получить свойство (*)
PreviewText = Object.GetPreviewText() Получить свойство (**)

Синтаксис COM:

Object.get_PreviewText(&PreviewText) Получить свойство,

Примечание:

Свойство доступно только для чтения.

IPropertyPreviewText – методы

UpdateParam – Обновить параметры

Интерфейс...

Синтаксис Automation:

BOOL UpdateParam();

Синтаксис COM:

HRESULT UpdateParam(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IPropertyToolBar

Интерфейс вспомогательного элемента Панели свойств. Набор команд элемента управления.

Иерархия:

IDispatch

IPropertyToolBar

Описание:

Позволяет управлять командами элемента управления на Панели свойств.

Примечание:

Данный интерфейс можно получить у интерфейсов IPropertyGrid или IPropertyEditList посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IPropertyToolBar – свойства

ButtonChecked – Нажатие кнопки

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

BOOL ButtonChecked = iObject.ButtonChecked (BtnID)	Получить свойство (*)
BOOL ButtonChecked = iObject.GetButtonChecked (BtnID)	Установить свойство (*)
iObject.ButtonChecked (BtnID) = ButtonChecked	Получить свойство(**)
iObject.SetButtonChecked (BtnID, ButtonChecked)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonChecked (BtnID, &ButtonChecked)	Получить свойство
iObject->put_ButtonChecked (BtnID, ButtonChecked)	Установить свойство

Входные параметры:

Long BtnID - идентификатор кнопки.

ButtonEnable – Доступность кнопки

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

BOOL ButtonEnable = iObject.ButtonEnable(BtnID)	Получить свойство (*)
BOOL ButtonEnable = iObject.GetButtonEnable(BtnID)	Установить свойство (*)
iObject.ButtonEnable(BtnID) = ButtonEnable	Получить свойство(**)
iObject.SetButtonEnable(BtnID, ButtonEnable)	Установить свойство (**)

Синтаксис COM:

iObject->get_ButtonEnable(BtnID, &ButtonEnable) — Получить свойство

iObject->put_ButtonEnable(BtnID, ButtonEnable) — Установить свойство

Входные параметры:

Long BtnID - идентификатор кнопки.

ButtonIconFont – Шрифт для кнопок

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ButtonIconFont = Object.ButtonIconFont(BtnID) — Получить свойство (*)

Object.ButtonIconFont(BtnID) = ButtonIconFont — Установить свойство (*)

ButtonIconFont = Object.GetButtonIconFont(BtnID) — Получить свойство(**)

Object.SetButtonIconFont(BtnID, ButtonIconFont) — Установить свойство (**)

Синтаксис COM:

Object.get_ButtonIconFont(BtnID, &ButtonIconFont) — Получить свойство

Object.put_ButtonIconFont(BtnID, ButtonIconFont) — Установить свойство

Входные параметры:

BtnID - идентификатор кнопки.

ButtonHint – Строка подсказки для кнопки, отображаемая в Строке сообщений

Интерфейс...

Тип данных: BSTR (строка).

Синтаксис Automation:

ButtonHint = iObject.ButtonHint (BtnID) — Получить свойство (*)

ButtonHint = iObject.GetButtonHint (BtnID) — Установить свойство (*)

```
iObject.ButtonHint( BtnID ) = ButtonHint      Получить свойство(**)
iObject.SetButtonHint (BtnID, ButtonHint)     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ButtonHint (BtnID,               Получить свойство
&ButtonHint)
iObject->put_ButtonHint (BtnID, ButtonHint)   Установить свойство
```

Входные параметры:

Long BtnID - идентификатор кнопки.

ButtonTips – Строка контекстной всплывающей подсказки для кнопки

Интерфейс...

Тип данных: BSTR (строка).

Синтаксис Automation:

```
ButtonTips = iObject.ButtonTips (BtnID)       Получить свойство (* )
ButtonTips = iObject.GetButtonTips (BtnID)    Установить свойство (* )
iObject.ButtonTips (BtnID) = ButtonTips       Получить свойство(**)
iObject.SetButtonTips (BtnID, ButtonTips)     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ButtonTips (BtnID,               Получить свойство
&ButtonTips)
iObject->put_ButtonTips (BtnID,               Установить свойство
ButtonTips)
```

Входные параметры:

Long BtnID - идентификатор кнопки.

ButtonType – Тип кнопки

Интерфейс...

Тип данных: из перечисления ButtonTypeEnum

Синтаксис Automation:

```
ButtonType = Object.ButtonType                Получить свойство (* )
Object.ButtonType = ButtonType                Установить свойство (* )
ButtonType = Object.GetButtonType()           Получить свойство(**)
Object.SetButtonType( ButtonType )            Установить свойство (**)
```

Синтаксис COM:

Object.get_ButtonType(&ButtonType)	Получить свойство
Object.put_ButtonType(ButtonType)	Установить свойство

ButtonVisible – Отображение кнопки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ButtonVisible = Object.ButtonVisible(BtnID)	Получить свойство (*)
Object.ButtonVisible(BtnID) = ButtonVisible	Установить свойство (*)
ButtonVisible = Object.GetButtonVisible(BtnID)	Получить свойство(**)
Object.SetButtonVisible(BtnID, ButtonVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_ButtonVisible(BtnID, &ButtonVisible)	Получить свойство
Object.put_ButtonVisible(BtnID, ButtonVisible)	Установить свойство

Входные параметры:

BtnID	- идентификатор кнопки.

ResModule – Модуль с ресурсами битмапов

Интерфейс...

Тип данных: VARIANT.

Синтаксис Automation:

VARIANT ResModule = iObject.ResModule	Получить свойство (*)
iObject.ResModule = ResModule	Установить свойство (*)
VARIANT ResModule = iObject.GetResModule()	Получить свойство(**)
iObject.SetResModule (ResModule)	Установить свойство (**)

Синтаксис COM:

iObject->get_ResModule (&ResModule)	Получить свойство
iObject->put_ResModule (ResModule)	Установить свойство

Примечание:

1. Значение свойства ResModule необходимо устанавливать до добавления кнопок.
2. Модуль с ресурсами может быть задан строкой (имя файла ресурса) или значением типа long (идентификатор ресурса).

IPropertyToolBar – методы

AddButton – Добавить кнопку

Интерфейс...

Синтаксис Automation:

void AddButton(long BtnID, VARIANT Bmp, long InsertAt);

Синтаксис COM:

HRESULT AddButton(long BtnID, VARIANT Bmp, long InsertAt);

Входные параметры:

BtnID	- идентификатор кнопки,
Bmp	- строка (имя bmp-файла) или идентификатор битмапа (long),
InsertAt	- индекс кнопки.

Интерфейс IPropertyAggregateControl

Интерфейс составного контрола.

Иерархия:

IDispatch

IKompasAPIObject

IPropertyControl

IPropertyAggregateControl

Примечания:

1. Интерфейс позволяет добавить элемент Панели свойств - составной контрол.
2. Данный интерфейс может быть получен от интерфейса коллекции элементов управления IPropertyControls с помощью свойства IPropertyControls::Item или метода IPropertyControls::Add с последующим приведением интерфейса IPropertyControl к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IPropertyAggregateControl – свойства

PropertyControl – Получить контрол по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IPropertyControl

Синтаксис Automation:

PropertyControl = Object.PropertyControl(Index)	Получить свойство (*)
PropertyControl = Object.GetPropertyControl(Index)	Получить свойство(**)

Синтаксис COM:

Object.get_PropertyControl(Index, &PropertyControl)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPropertyAggregateControl – методы

Add – Добавить контрол в группу

Интерфейс...

Синтаксис Automation:

BOOL Add(IPropertyControl * Control);

Синтаксис COM:

HRESULT Add(IPropertyControl * Control, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Control	- указатель на интерфейс добавляемого контрола.

Интерфейс IPropertyControls

Интерфейс коллекции элементов управления.

Иерархия:

IKompasAPIObject

IKompasCollection

IPropertyControls

Примечание:

1. Интерфейс позволяет управлять коллекцией элементов управления.
2. Данный интерфейс может быть получен от интерфейса закладки панели свойств IPropertyTab с помощью свойства IPropertyTab::PropertyControls.

IPropertyControls – свойства

Item – Элемент управления, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IPropertyControl.

Синтаксис Automation:

Item = iObject.Item (Index)	Получить свойство (*)
Item = iObject.GetItem (Index)	Получить свойство(**)

Синтаксис COM:

iObject->get_Item (Index, &IPropertyControl)	Получить свойство

Входные параметры:

index	- индекс элемента управления.

Примечание:

Свойство доступно только для чтения.

IPropertyControls – методы

Add – Создать элемент управления

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (ControlTypeEnum type);

Синтаксис COM:

HRESULT Add ([in] ControlTypeEnum Type, [out, retval] IPropertyControl** Result);

Входные параметры:

type	- тип элемента управления из ControlTypeEnum.

Возвращаемое значение:

Указатель на интерфейс IPropertyControl

Примечание:

Добавляет элемент управления в коллекцию.

Delete – Удалить элемент управления

Интерфейс...

Синтаксис Automation:

BOOL Delete (VARIANT Index);

Синтаксис COM:

HRESULT Delete ([in] VARIANT Index, [out, retval] VARIANT_BOOL* pVal);

Входные параметры:

index - индекс элемента управления или имя элемента управления.

Возвращаемое значение:

- результат выполнения метода.

Примечание:

Метод позволяет удалить элемент управления, заданный по индексу.

Интерфейс IPropertyTabs

Интерфейс коллекции вкладок Панели свойств.

Иерархия:

IKompasAPIObject

IKompasCollection

IPropertyTabs

Примечание:

Данный интерфейс может быть получен от интерфейса параметров процесса IProcessParam с помощью свойства IProcessParam::PropertyTabs или от интерфейса панели свойств IPropertyManager с помощью свойства IPropertyManager::PropertyTabs.

IPropertyTabs – свойства

Active – Интерфейс текущей вкладки Панели свойств

Функция не поддерживается

Интерфейс...

Тип данных: указатель на интерфейс IPropertyTab.

Синтаксис Automation:

Active = iObject.Active	Получить свойство (*)
Active = iObject.GetActive()	Получить свойство(**)

Синтаксис COM:

iObject->get_Active (&Active) Получить свойство

Item – Вкладка, заданная по индексу или по имени

Интерфейс...

Тип данных: указатель на интерфейс IPropertyTab.

Синтаксис Automation:

Item = iObject.Item Получить свойство (*)
Item = iObject.GetItem() Получить свойство(**)

Синтаксис COM:

iObject->get_Item (&PropertyTab) Получить свойство

Входные параметры:

index - индекс или имя закладки (тип VARIANT).

Примечание:

Свойство доступно только для чтения.

SystemTab – Интерфейс системной вкладки

Интерфейс...

Интерфейс системной вкладки.

Тип данных: указатель на интерфейс IPropertyTab.

Синтаксис Automation:

SystemTab = iObject.SystemTab Получить свойство (*)
SystemTab = iObject.GetSystemTab() Получить свойство(**)

Синтаксис COM:

iObject->get_SystemTab(&SystemTab) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Системную вкладку можно получить для процессных параметров. Для системной вкладки можно установить свойства:

- Visible - видимость закладки Панели свойств, т.е. сделать закладку невидимой. Caption- изменить заголовок вкладки Панели свойств.
- Caption- заголовок вкладки Панели свойств.

- Image - установить битмап на вкладке.
- ResModule - модуль с ресурсом битмапа, если битмап задан в виде идентификатора ресурса библиотеки.

IPropertyTabs – методы

Add – Создать вкладку Панели свойств (добавляет вкладку в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (BSTR caption);

Синтаксис COM:

HRESULT Add ([in] BSTR caption, [out, retval] IPropertyTab** Result);

Входные параметры:

caption - имя закладки.

Возвращаемое значение:

- указатель на интерфейс IPropertyTab.

Delete – Удалить вкладку Панели свойств, заданную по индексу или по имени

Интерфейс...

Синтаксис Automation:

BOOL Delete (VARIANT index);

Синтаксис COM:

HRESULT Delete ([in] VARIANT index, [out, retval] VARIANT_BOOL* pVal);

Входные параметры:

Index - индекс или имя закладки.

Возвращаемое значение:

- результат выполнения метода.

Работа с контекстной панелью и меню

Интерфейс IProcessContextPanel

Интерфейс контекстной панели процесса.

Иерархия:

IDispatch

IProcessContextPanel

Примечание:

Интерфейс используется в событии ksPropertyManagerNotify::FillContextPanel

Интерфейс позволяет заменить обычное меню процесса на панель которая может содержать контролы Панели свойств.

Для использования контекстной панели в процессе нужно в событии ksPropertyManagerNotify::GetContextMenuType вернуть константу ksProcessContextPanel из перечисления ksProcessContextMenuType .

IProcessContextPanel – методы

AddAdditionalButtonsFromControl – Добавить группу дополнительных кнопок

Интерфейс...

Синтаксис Automation:

BOOL AddAdditionalButtonsFromControl(IPropertyControl * PVal, BSTR Title);

Синтаксис COM:

HRESULT AddAdditionalButtonsFromControl(IPropertyControl * PVal, BSTR Title, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

PVal	- указатель на интерфейс добавляемого контрола,
Title	- заголовок.

AddControl – Добавить контрол

Интерфейс...

Синтаксис Automation:

BOOL AddControl(IPropertyControl * PVal);

Синтаксис COM:

HRESULT AddControl(IPropertyControl * PVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

PVal - указатель на интерфейс добавляемого контрола.

AddGeomCalculatorCommands – Добавить команды меню геометрического калькулятора

Интерфейс...

Синтаксис Automation:

BOOL AddGeomCalculatorCommands(IPropertyControl * PVal);

Синтаксис COM:

HRESULT AddGeomCalculatorCommands(IPropertyControl * PVal, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

PVal - указатель на интерфейс контрола.

AddGroupBegin – Добавить начало подгруппы

Интерфейс...

Синтаксис Automation:

BOOL AddGroupBegin(BSTR Title);

Синтаксис COM:

HRESULT AddGroupBegin(BSTR Title, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Title - заголовок.

AddGroupEnd – Добавить конец подгруппы

Интерфейс...

Синтаксис Automation:

BOOL AddGroupEnd();

Синтаксис COM:

HRESULT AddGroupEnd(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

AddLinkContentButton – Добавить кнопку ссылочного содержимого регистратора

Интерфейс...

Синтаксис Automation:

BOOL AddLinkContentButton(IPropertyControl * PVal, BSTR Title);

Синтаксис COM:

HRESULT AddLinkContentButton(IPropertyControl * PVal, BSTR Title, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

PVal - указатель на интерфейс добавляемого контрола,
Title - заголовок.

AddNewSearchButton – Добавить кнопку перебор объектов

Интерфейс...

Синтаксис Automation:

BOOL AddNewSearchButton();

Синтаксис COM:

HRESULT AddNewSearchButton(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

AddSeparator – Добавить разделитель

Интерфейс...

Синтаксис Automation:

BOOL AddSeparator();

Синтаксис COM:

HRESULT AddSeparator(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

AddSnapCommands – Добавить команды привязок

Интерфейс...

Синтаксис Automation:

BOOL AddSnapCommands();

Синтаксис COM:

HRESULT AddSnapCommands(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IProcessContextIconMenu

Интерфейс контекстного меню процесса с иконками.

Иерархия:

IDispatch

IProcessContextIconMenu

IProcessContextIconMenu – методы

AddMenuCommand – Добавить пункт меню

Интерфейс...

Синтаксис Automation:

BOOL AddMenuCommand(long Id, BSTR Title, long Icon, BSTR IconFont);

Синтаксис COM:

HRESULT AddMenuCommand(long Id, BSTR Title, long Icon, BSTR IconFont, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Id	- идентификатор команды,
Title	- заголовок,
Icon	- идентификатор иконки из шрифта,
IconFont	- имя библиотечного шрифта или пустая строка для шрифта Компас.

Документ

Атрибуты

Интерфейс IAttrTypeMng

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Менеджер типов атрибутов.

Иерархия:

IKompasAPIObject

IAttrTypeMng

Примечание:

1. Интерфейс позволяет создавать, получать и просматривать типы атрибутов.
2. Интерфейс является дополнительным к интерфейсу приложения IApplication.

IAttrTypeMng – методы

ChoiceAttrTypes – Открыть диалог для просмотра в библиотеке атрибутов списка типов атрибутов и выбора нужного типа

Интерфейс...

Синтаксис Automation:

LPDISPATCH ChoiceAttrTypes(OLE_HANDLE HWnd,
VARIANT Libname);

Синтаксис COM:

HRESULT ChoiceAttrTypes(OLE_HANDLE HWnd,
VARIANT Libname,
IAttributeType ** Result);

Входные параметры:

HWnd	- дескриптор главного окна системы КОМПАС, который можно получить с помощью функции GetHWindow,
Libname	- имя библиотеки типов атрибутов.

Возвращаемое значение:

Указатель на интерфейс выбранного в диалоге типа атрибута IAttributeType.

Примечание:

Метод позволяет открыть диалог просмотра типов атрибутов в библиотеке.

CreateAttrType – Создать тип атрибута в заданной библиотеке

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateAttrType(VARIANT Libname);

Синтаксис COM:

HRESULT CreateAttrType(VARIANT Libname,
IAttributeType ** Result);

Входные параметры:

Libname - имя библиотеки типов атрибутов.

Возвращаемое значение:

- указатель на интерфейс типа атрибута IAttributeType.

Примечание:

Метод позволяет создать тип атрибута в библиотеке Libname.

GetAttrType – Получить тип атрибута из заданной библиотеки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetAttrType(double AttrID,
VARIANT Libname);

Синтаксис COM:

HRESULT GetAttrType(double AttrID,
VARIANT Libname,
IAttributeType ** Result);

Входные параметры:

AttrID - уникальный номер типа атрибута,
Libname - имя библиотеки типов атрибутов.

Возвращаемое значение:

- указатель на интерфейс типа атрибута IAttributeType.

Примечание:

Метод позволяет получить тип атрибута из библиотеки Libname.

GetAttrTypes – Возвращает массив типов атрибутов, находящихся в заданной библиотеке типов

Интерфейс...

Синтаксис Automation:

VARIANT GetAttrTypes(VARIANT Libname);

Синтаксис COM:

HRESULT GetAttrTypes(VARIANT Libname,
VARIANT * Result);

Входные параметры:

Libname - имя библиотеки типов атрибутов.

Возвращаемое значение:

- массив SAFEARRAY типа VT_ARRAY | VT_DISPATCH указателей на интерфейсы типов атрибутов, находящихся в заданной библиотеке.

Примечание:

Метод позволяет получить массив типов атрибутов из заданной библиотеки. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Интерфейс IAttribute

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Интерфейс атрибута.

Иерархия:

IKompasAPIObject

IAttribute

Примечание:

1. Интерфейс позволяет осуществлять работу с атрибутом.
2. Интерфейс можно получить с помощью метода IKompasDocument1::CreateAttr.

IAttribute – свойства

AttributeType – Получить тип атрибута

Интерфейс...

Тип данных: указатель на интерфейс типа атрибута IAttributeType.

Синтаксис Automation:

```
AttributeType = iObject.AttributeType  ( Index );        Получить свойство (* )
AttributeType = iObject.GetAttributeType( Index );       Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_AttributeType( Index, &AttributeType )          Получить свойство,
```

Примечание:

1. Свойство позволяет получить указатель на интерфейс типа атрибута.
2. Свойство доступно только для чтения.

ColumnsCount – Получить количество столбцов в таблице атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
ColumnsCount = Object.ColumnsCount                  Получить свойство (* )
ColumnsCount = Object.GetColumnsCount()             Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ColumnsCount( &ColumnsCount )          Получить свойство,
```

Примечание:

1. Свойство позволяет получить количество столбцов в таблице атрибута.
2. Свойство доступно только для чтения.

ColumnKey – Получить ключи колонок

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_I4

Синтаксис Automation:

```
ColumnKey = Object.ColumnKey                        Получить свойство (* )
ColumnKey = Object.GetColumnKey()                   Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ColumnKey( &ColumnKey )                Получить свойство,
```

Примечание:

1. Свойство позволяет получать значения ключей колонок атрибута.
2. Для задания значений ключей колонок следует использовать метод IAttribute::SetColumnKey.
3. Свойство доступно только для чтения.

FlagVisible – Получить значения видимости колонок в виде массива SAFEARRAY VT_ARRAY | VT_BOOL

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_BOOL

Синтаксис Automation:

```
FlagVisible = Object.FlagVisible                Получить свойство (* )
FlagVisible = Object.GetFlagVisible()           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_FlagVisible( &FlagVisible )          Получить свойство,
```

Примечание:

1. Свойство позволяет получать значения видимости колонок атрибута.
2. Для задания значений видимости колонок следует использовать метод IAttribute::SetFlagVisible.
3. Свойство доступно только для чтения.

Objects – Объекты, прикрепленные к атрибуту в виде массива SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных:VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

```
Objects = Object.Objects                        Получить свойство (* )
Object.Objects = Objects                        Установить свойство (* )
Objects = Object.GetObjects()                   Получить свойство (**)
Object.SetObjects( Objects )                    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Objects( &Objects )                  Получить свойство,
Object.put_Objects(  Objects )                  Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать массив объектов, прикрепленных к атрибуту.

RecordFlagVisible – Получить значения видимости колонок внутри записи в виде массива SAFEARRAY VT_ARRAY | VT_BOOL

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_BOOL

Синтаксис Automation:

RecordFlagVisible = Object.RecordFlagVisible(ColumnNumb)	Получить свойство (*)
RecordFlagVisible = Object.GetRecordFlagVisible(ColumnNumb)	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(ColumnNumb, &ColumnsCount)	Получить свойство

Входные параметры:

ColumnNumb	- номер колонки записи.

Примечание:

1. Свойство позволяет получить значения видимости колонок внутри записи.
2. Установить значения видимости колонок записи можно с помощью метода IAttribute::SetRecordFlagVisible.
3. Свойство доступно только для чтения.

RowsCount – Получить количество строк в таблице атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = Object.RowsCount	Получить свойство (*)
RowsCount = Object.GetRowsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_RowsCount(&RowsCount) Получить свойство,

Примечание:

1. Свойство позволяет получить количество строк в таблице атрибута.
2. Свойство доступно только для чтения.

Value – Значение ячейки из таблицы атрибута

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Value = Object.Value(RowNumb, ColumnNumb) Получить свойство (*)
Value = Object.GetValue(RowNumb, ColumnNumb) Получить свойство (**)

Синтаксис COM:

Object.get_Value(RowNumb, ColumnNumb, &Value) Получить свойство

Входные параметры:

RowNumb (long) - номер строки атрибута,
ColumnNumb (long) - номер столбца атрибута.

Примечание:

1. Свойство позволяет получать значение ячейки атрибута.
2. Для задания значения ячейки следует использовать метод IAttribute::SetValue.
3. Свойство доступно только для чтения.

Values – Получить значения всех ячеек из таблицы атрибута

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_VARIANT

Синтаксис Automation:

Values = Object.Values Получить свойство (*)
Values = Object.GetValues() Получить свойство (**)

Синтаксис COM:

Object.get_Values(&Values) Получить свойство

Примечание:

1. Свойство позволяет значения всех ячеек из таблицы атрибута.
2. Свойство доступно только для чтения.

IAttribute – методы

AddRow – Добавить строку к табличному атрибуту неопределенной длины

Интерфейс...

Синтаксис Automation:

BOOL AddRow(BSTR Password,
long RowNumb);

Синтаксис COM:

HRESULT AddRow(BSTR Password,
long RowNumb,
BOOL * Result);

Входные параметры:

Password - пароль,
RowNumb - номер строки, после которой нужно вставить строку.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет добавить строку в таблицу атрибута.

Delete – Удалить атрибут

Интерфейс...

Синтаксис Automation:

BOOL Delete(BSTR Password,
VARIANT Objects);

Синтаксис COM:

HRESULT Delete(BSTR Password,
VARIANT Objects
BOOL * Result);

Входные параметры:

Password	- пароль,
Objects	- массив SAFEARRAY типа VT_ARRAY | VT_DISPATCH объектов, у которых надо удалить атрибут.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить атрибут у объектов.

DeleteRow – Удалить строку табличного атрибута неопределенной длины

Интерфейс...

Синтаксис Automation:

```
BOOL DeleteRow( BSTR Password,
long RowNumb );
```

Синтаксис COM:

```
HRESULT DeleteRow( BSTR Password,
long RowNumb,
BOOL * Result );
```

Входные параметры:

Password	- пароль,
RowNumb	- номер удаляемой строки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить строку из таблицы атрибута.

GetKeysInfo – Выдать информацию о ключах атрибута

Интерфейс...

Синтаксис Automation:

```
BOOL GetKeysInfo( long*  Key1,
long*  Key2,
long*  Key3,
```

long* Key4,
double* Numb);

Синтаксис COM:

HRESULT GetKeysInfo(long* Key1,
long* Key2,
long* Key3,
long* Key4,
double* Numb,
BOOL * Result);

Выходные параметры:

Key1, Key2, Key3, Key4	- ключи атрибута,
Numb	- уникальный номер атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить информацию о ключах атрибута.

SetColumnKey – Установить ключи колонок

Интерфейс...

Синтаксис Automation:

BOOL SetColumnKey(BSTR Password,
VARIANT Keys);

Синтаксис COM:

HRESULT SetColumnKey(BSTR Password,
VARIANT Keys,
BOOL *Result);

Входные параметры:

Password	- пароль,
Keys	- массив SAFEARRAY типа VT_ARRAY | VT_I4 значений ключей колонок атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить ключи колонок атрибута.

SetFlagVisible – Установить значения видимости колонок в виде массива SAFEARRAY VT_ARRAY | VT_BOOL

Интерфейс...

Синтаксис Automation:

BOOL SetFlagVisible(BSTR Password,
VARIANT Values);

Синтаксис COM:

HRESULT SetFlagVisible(BSTR Password,
VARIANT Values,
BOOL *Result);

Входные параметры:

Password	- пароль,
Values	- массив SAFEARRAY типа VT_ARRAY I VT_BOOL значений видимости колонок атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить значения видимости колонок атрибута.

SetKeysInfo – Установить информацию о ключах атрибута

Интерфейс...

Синтаксис Automation:

BOOL SetKeysInfo(BSTR Password,
long Key1,
long Key2,
long Key3,
long Key4);

Синтаксис COM:

HRESULT SetKeysInfo(BSTR Password,
long Key1,
long Key2,
long Key3,
long Key4,
BOOL * Result);

Входные параметры:

Password	- пароль,
Key1, Key2, Key3, Key4	- ключи атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет задать ключи атрибута.

SetPassword – Установить новый пароль на изменение данных атрибута

Интерфейс...

Синтаксис Automation:

BOOL SetPassword(BSTR OldPassword,
BSTR NewPassword);

Синтаксис COM:

HRESULT SetPassword(BSTR OldPassword,
BSTR NewPassword,
BOOL * Result);

Входные параметры:

OldPassword	- старый пароль,
NewPassword	- новый пароль.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить новый пароль на изменение данных атрибута.

SetRecordFlagVisible – Установить значения видимости колонок внутри записи в виде массива SAFEARRAY VT_ARRAY | VT_BOOL

Интерфейс...

Синтаксис Automation:

BOOL SetRecordFlagVisible(BSTR Password,
long ColumnNumb,
VARIANT Values);

Синтаксис COM:

HRESULT SetRecordFlagVisible(BSTR Password,
long ColumnNumb,
VARIANT Values,
BOOL * Result);

Входные параметры:

Password	- пароль,
ColumnNumb	- номер колонки записи,
Values	- массив SAFEARRAY типа VT_ARRAY I VT_BOOL значений видимости колонок записи.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить значения видимости колонок записи.

SetValue – Установить значение ячейки

Интерфейс...

Синтаксис Automation:

BOOL SetValue(BSTR Password,
long RowNumb,
long ColumnNumb,
VARIANT Value);

Синтаксис COM:

HRESULT SetValue(BSTR Password,
long RowNumb,
long ColumnNumb,
VARIANT Value,
BOOL* Result);

Входные параметры:

Password	- пароль,
RowNumb	- номер строки атрибута,
ColumnNumb	- номер столбца атрибута,
Value	- значение ячейки атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

Метод позволяет установить значение ячейки атрибута.

SetValues – Установить значения всех ячеек таблицы атрибута

Интерфейс...

Синтаксис Automation:

BOOL SetValues(BSTR Password,
VARIANT Values);

Синтаксис COM:

HRESULT SetValues(BSTR Password,
VARIANT Values,
BOOL *Result);

Входные параметры:

Password	- пароль,
Values	- массив SAFEARRAY типа VT_ARRAY | VT_VARIANT значений ячеек атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить значения всех ячеек таблицы атрибута.

ViewEdit – Открыть диалог для просмотра и редактирования атрибута

Интерфейс...

Синтаксис Automation:

BOOL ViewEdit(OLE_HANDLE Parent,
BSTR Password,
BOOL ReadOnly);

Синтаксис COM:

HRESULT ViewEdit(OLE_HANDLE Parent,
BSTR Password,
BOOL ReadOnly,
BOOL *Result);

Выходные параметры:

Parent	- дескриптор главного окна системы КОМПАС, который можно получить с помощью функции GetHWindow,
Password	- пароль,
ReadOnly	- TRUE - открыть диалог для просмотра - FALSE - открыть диалог для редактирования.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет открыть диалог для просмотра и редактирования атрибута.

Интерфейс IAttributeType

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Интерфейс параметров типа атрибута.

Иерархия:

IKompasAPIObject

IAttributeType

Примечание:

1. Интерфейс позволяет задавать параметры типа атрибута.
2. Интерфейс можно получить следующими способами:

- у менеджера типа атрибутов с помощью методов IAttrTypeMng::CreateAttrType, IAttrTypeMng::GetAttrType, IAttrTypeMng::ChoiceAttrTypes,
- у атрибута с помощью свойства IAttribute::AttributeType.

IAttributeType – свойства

AttrType – Тип данных

Интерфейс...

Тип данных: из перечисления ksAttributeTypeEnum

Синтаксис Automation:

AttrType = Object.AttrType	Получить свойство (*)
Object.AttrType = AttrType	Установить свойство (*)
AttrType = Object.GetAttrType()	Получить свойство (**)
Object.SetAttrType(AttrType)	Установить свойство (**)

Синтаксис COM:

Object.get_AttrType(&AttrType)	Получить свойство,
Object.put_AttrType(AttrType)	Установить свойство.

Примечание:

Свойство позволяет устанавливать и получать тип данных для типа атрибута.

ColumnsCount – Получить количество колонок табличного атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnsCount = Object.ColumnsCount	Получить свойство (*)
ColumnsCount = Object.GetColumnsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(&ColumnsCount)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

ColumnInfo – Получить информацию о столбце атрибута

Интерфейс...

Тип данных: указатель на интерфейс параметров столбца атрибута IColumnInfo

Синтаксис Automation:

ColumnInfo = Object.ColumnInfo (Index)	Получить свойство (*)
ColumnInfo = Object.GetColumnInfo(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnInfo(Index, &ColumnInfo)	Получить свойство,

Входные параметры:

Index (VARIANT)	- индекс столбца.

Примечание:

1. Свойство позволяет получать интерфейс параметров столбца атрибута.
2. Свойство доступно только для чтения.

FileName – Имя библиотеки типов атрибутов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство,
Object.put_RowsCount(RowsCount)	Установить свойство.

Примечание:

1. Свойство позволяет получить имя библиотеки типов атрибутов.
2. Свойство доступно только для чтения.

RowsCount – Количество строк в таблице

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = Object.RowsCount	Получить свойство (*)
Object.RowsCount = RowsCount	Установить свойство (*)
RowsCount = Object.GetRowsCount()	Получить свойство (**)
Object.SetRowsCount(RowsCount)	Установить свойство (**)

Синтаксис COM:

Object.get_RowsCount(&RowsCount)	Получить свойство,
Object.put_RowsCount(RowsCount)	Установить свойство.

Примечание:

Свойство позволяет устанавливать и получать количество строк в таблице.

TypeName – Название типа атрибута

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

TypeName = Object.TypeName	Получить свойство (*)
Object.TypeName = TypeName	Установить свойство (*)
TypeName = Object.GetTypeName()	Получить свойство (**)

Object.SetTypeName(TypeName) Установить свойство (**)

Синтаксис COM:

Object.get_TypeName(&TypeName) Получить свойство,
Object.put_TypeName(TypeName) Установить свойство.

Примечание:

Свойство позволяет устанавливать и получать название типа атрибута.

UniqueNumb – Уникальный номер типа

Интерфейс...

Тип данных: double.

Синтаксис Automation:

UniqueNumb = Object.UniqueNumb Получить свойство (*)
UniqueNumb = Object.GetUniqueNumb() Получить свойство (**)

Синтаксис COM:

Object.get_UniqueNumb(&UniqueNumb) Получить свойство,

Примечание:

1. Свойство позволяет получать уникальный номер типа атрибута.
2. Свойство доступно только для чтения.

IAttributeType – методы

AddColumn – Добавить колонку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddColumn(long Index,
long Type,
BSTR Name);

Синтаксис COM:

HRESULT AddColumn(long Index,
ksValueTypeEnum Type,
BSTR Name,
IColumnInfo** Result);

Входные параметры:

Index - индекс колонки, после которой нужно добавить колонку,

Type	- тип значений колонки,
Name	- название колонки.

Возвращаемое значение:

Указатель на интерфейс IColumnInfo.

Примечание:

Метод позволяет добавить колонку к типу атрибута.

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete (BSTR Password);

Синтаксис COM:

HRESULT Delete(BSTR Password,
BOOL * Result);

Входные параметры:

Password	- пароль.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить тип атрибута.

GetKeysInfo – Получить информацию о ключах атрибута

Интерфейс...

Синтаксис Automation:

BOOL GetKeysInfo(long* Key1,
long* Key2,
long* Key3,
long* Key4);

Синтаксис COM:

HRESULT GetKeysInfo(long* Key1,
long* Key2,
long* Key3,
long* Key4,
BOOL* Result);

Выходные параметры:

Key1, Key2, Key3, Key4	- ключи атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить информацию о ключах атрибута.

SetKeysInfo – Установить информацию о ключах атрибута

Интерфейс...

Синтаксис Automation:

```
BOOL SetKeysInfo( long Key1,
long Key2,
long Key3,
long Key4 );
```

Синтаксис COM:

```
HRESULT SetKeysInfo( long Key1,
long Key2,
long Key3,
long Key4,
BOOL * Result );
```

Входные параметры:

Key1, Key2, Key3, Key4	- ключи атрибута.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить информацию о ключах атрибута.

SetPassword – Установить новый пароль на изменение данных атрибута

Интерфейс...

Синтаксис Automation:

BOOL SetPassword(BSTR OldPassword,
BSTR NewPassword);

Синтаксис COM:

HRESULT SetPassword(BSTR OldPassword,
BSTR NewPassword,
BOOL * Result);

Входные параметры:

OldPassword	- старый пароль,
NewPassword	- новый пароль.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить новый пароль на изменение данных атрибута.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update(BSTR Password);

Синтаксис COM:

HRESULT Update(BSTR Password,
BOOL * Result);

Входные параметры:

Password	- пароль.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет обновить данные в типе атрибута.

ViewEdit – Открыть диалог для просмотра и редактирования типа атрибута

Интерфейс...

Синтаксис Automation:

BOOL ViewEdit(OLE_HANDLE Parent,
BSTR Password,
BOOL ReadOnly);

Синтаксис COM:

HRESULT ViewEdit(OLE_HANDLE Parent,
BSTR Password,
BOOL ReadOnly,
BOOL *Result);

Выходные параметры:

Parent	- дескриптор главного окна системы КОМПАС, который можно получить с помощью функции GetHWindow,
Password	- пароль,
ReadOnly	- TRUE - открыть диалог для просмотра, - FALSE - открыть диалог для редактирования.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет открыть диалог для просмотра и редактирования типа атрибута.

Интерфейс IColumnInfo

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Интерфейс параметров столбца табличного атрибута.

Иерархия:

IKompasAPIObject

IColumnInfo

Примечание:

1. Интерфейс позволяет задавать параметры столбца табличного атрибута.
2. Интерфейс можно получить следующими способами:

- у интерфейса типа атрибута с помощью свойства IAttributeType::ColumnInfo и метода IAttributeType::AddColumn.
- для столбца записи - с помощью свойства IColumnInfo::RecordColumnInfo и метода IColumnInfo::AddRecordColumn.

IColumnInfo – свойства

Caption – Заголовок столбца

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Caption = Object.Caption	Получить свойство (*)
Object.Caption = Caption	Установить свойство (*)
Caption = Object.GetCaption()	Получить свойство (**)
Object.SetCaption(Caption)	Установить свойство (**)

Синтаксис COM:

Object.get_Caption(&Caption)	Получить свойство,
Object.put_Caption(Caption)	Установить свойство.

Примечание:

Свойство позволяет устанавливать и получать заголовок столбца атрибута.

ColType – Тип данных столбца

Интерфейс...

Тип данных: из перечисления ksValueTypeEnum

Синтаксис Automation:

ColType = Object.ColType	Получить свойство (*)
Object.ColType = ColType	Установить свойство (*)
ColType = Object.GetColType()	Получить свойство (**)
Object.SetColType(ColType)	Установить свойство (**)

Синтаксис COM:

Object.get_ColType(&ColType)	Получить свойство,
Object.put_ColType(ColType)	Установить свойство.

Примечание:

1. Свойство позволяет устанавливать и получать тип данных столбца атрибута.
2. При выборе типа ksValueTypeString длина строки задается свойством IColumnInfo::Range (1…4000).

DefValue – Значение по умолчанию

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
DefValue = Object.DefValue                  Получить свойство (* )
Object.DefValue = DefValue                  Установить свойство (* )
DefValue = Object.GetDefValue()             Получить свойство (**)
Object.SetDefValue( DefValue )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DefValue( &DefValue )            Получить свойство,
Object.put_DefValue(  DefValue )            Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать умолчательное значение колонки атрибута.

Key – Ключ колонки

Интерфейс...

Тип данных:long.

Синтаксис Automation:

```
Key = Object.Key                            Получить свойство (* )
Object.Key = Key                            Установить свойство (* )
Key = Object.GetKey()                       Получить свойство (**)
Object.SetKey( Key )                        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Key( &Key )                      Получить свойство,
Object.put_Key(  Key )                      Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать ключ столбца атрибута.

ListValue – Использовать список значений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ListValue = Object.ListValue                Получить свойство (* )
Object.ListValue = ListValue                Установить свойство (* )
ListValue = Object.GetListValue()           Получить свойство (**)
Object.SetListValue( ListValue )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ListValue( &ListValue )    Получить свойство,
Object.put_ListValue(  ListValue )    Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать признак использования списка значений.

Range – Диапазон значений

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Range = Object.Range                  Получить свойство (* )
Object.Range = Range                  Установить свойство (* )
Range = Object.GetRange()             Получить свойство (**)
Object.SetRange( Range )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Range( &Range )            Получить свойство,
Object.put_Range(  Range )            Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать диапазон значений столбца атрибута.

RecordColumnsCount – Получить количество столбцов записи

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
RecordColumnsCount =                  Получить свойство (* )
Object.RecordColumnsCount
RecordColumnsCount =                  Получить свойство (**)
Object.GetRecordColumnsCount()
```

Синтаксис COM:

```
Object.get_RecordColumnsCount(        Получить свойство,
&RecordColumnsCount )
```

Примечание:

Свойство доступно только для чтения.

RecordColumnInfo – Получить информацию о столбце атрибута

Интерфейс...

Тип данных: указатель на интерфейс IColumnInfo

Синтаксис Automation:

```
RecordColumnInfo = Object.RecordColumnInfo      Получить свойство (* )
RecordColumnInfo =                              Получить свойство (**)
Object.GetRecordColumnInfo()
```

Синтаксис COM:

```
Object.get_RecordColumnInfo(                    Получить свойство,
&RecordColumnInfo )
```

Примечание:

1. Свойство позволяет получать информацию о столбце записи.
2. Свойство доступно только для чтения.

SortListValue – Учитывать порядок размещения значений при сортировке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
SortListValue = Object.SortListValue            Получить свойство (* )
Object.SortListValue = SortListValue            Установить свойство (* )
SortListValue = Object.GetSortListValue()       Получить свойство (**)
Object.SetSortListValue( SortListValue )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SortListValue( &SortListValue )      Получить свойство,
Object.put_SortListValue(  SortListValue )      Установить свойство.
```

Примечание:

Свойство позволяет устанавливать и получать признак учета порядка значений при сортировке.

IColumnInfo – методы

AddRecordColumn – Добавить колонку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddRecordColumn(long Index,
long Type,
BSTR Name);

Синтаксис COM:

HRESULT AddRecordColumn(long Index,
ksValueTypeEnum Type,
BSTR Name,
IColumnInfo** Result);

Входные параметры:

Index	- индекс колонки, после которой нужно добавить колонку,
Type	- тип значений колонки,
Name	- название колонки.

Возвращаемое значение:

указатель на интерфейс IColumnInfo.

Примечание:

Метод позволяет добавить колонку к записи.

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить колонку к записи.

Базовые интерфейсы

Интерфейс IKompasDocument

Интерфейс событий документа, работа с файлом...

Базовый класс документов КОМПАС.

Иерархия:

IKompasAPIObject

IKompasDocument

IKompasDocument1

Примечание:

1. Данный интерфейс является базовым для всех интерфейсов документов КОМПАС API.
2. Данный интерфейс может быть получен следующими способами:

- от интерфейса приложения IApplication с помощью свойства IApplication::ActiveDocument,
- от интерфейса коллекции документов IDocuments с помощью свойства IDocuments::Item и методов IDocuments::Add, IDocuments::Open,
- от интерфейса элемента библиотеки документов IInsert с помощью метода IInsert::Edit.

3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить дополнительный интерфейс IKompasDocument1и IProductDataManager.

IKompasDocument – свойства

Active – Активность документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Active = iObject.Active                 Получить свойство (* )
iObject.Active = Active                 Установить свойство (* )
Active = iObject.GetActive()            Получить свойство (**)
iObject.SetActive (Active)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Active (&Active)           Получить свойство
iObject->put_Active (Active)            Установить свойство
```

Примечание:

Позволяет получить активность документа. Установить можно только значение TRUE, т.е. сделать документ активным, если он неактивен. Значение FALSE игнорируется.

Changed – Документ изменен

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Changed = iObject.Changed                Получить свойство (* )
Changed = iObject.GetChanged()           Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Changed (&Changed)          Получить свойство
```

Значения свойства:

```
TRUE      - документ изменен с момента последнего сохранения,
FALSE     - документ не изменялся.
```

DocumentFrames – Интерфейс коллекции окон документа

Интерфейс...

Тип данных: указатель на интерфейс коллекции окон IDocumentFrames

Синтаксис Automation:

```
DocumentFrames = iObject.DocumentFrames    Получить свойство(* )
DocumentFrames =                           Получить свойство (**)
iObject.GetDocumentFrames()
```

Синтаксис COM:

```
iObject->get_DocumentFrames                Получить свойство
(&DocumentFrames)
```

Примечание:

Свойство доступно только для чтения.

DocumentSettings – Настройки документа

Интерфейс...

Тип данных: - указатель на интерфейс IDocumentSettings текущих настроек документа.

Синтаксис Automation:

```
DocumentSettings =                         Получить свойство (* )
iObject.DocumentSettings
```

DocumentSettings = iObject.GetDocumentSettings() — Получить свойство (**)

Синтаксис COM:

iObject->get_DocumentSettings (&DocumentSettings) — Получить свойство

Примечание:

Свойство доступно только для чтения.

DocumentType – Тип документа

Интерфейс...

Тип данных: тип документа из перечисления DocumentTypeEnum

Синтаксис Automation:

DocumentType = iObject.DocumentType — Получить свойство *)
DocumentType = iObject.GetDocumentType() — Получить свойство (**)

Синтаксис COM:

iObject->get_DocumentType (&DocumentType) — Получить свойство

Примечание:

Свойство доступно только для чтения.

LayoutSheets – Листы оформления

Интерфейс...

Тип данных: указатель на интерфейс ILayoutSheets коллекции листов оформления

Синтаксис Automation:

LayoutSheets = iObject.LayoutSheets — Получить свойство
LayoutSheets = iObject.GetLayoutSheets() — Получить свойство (**)

Синтаксис COM:

iObject->get_LayoutSheets (&LayoutSheets) — Получить свойство

Примечание:

1. Свойство используется только для документа Чертеж.
2. Свойство доступно только для чтения.

Name – Имя документа

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

```
Name = iObject.Name                    Получить свойство (* )
Name = iObject.GetName()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name (&Name)              Получить свойство
```

Примечание:

Свойство доступно только для чтения. Возвращает имя файла документа без пути.

Path – Путь к файлу документа

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Path = iObject.Path                    Получить свойство (* )
Path = iObject.GetPath()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Path (&Path)              Получить свойство
```

Примечание:

Свойство доступно только для чтения. Возвращает путь к файлу документа без имени и расширения файла.

PathName – Полное имя документа

Интерфейс...

Тип данных:строка

Синтаксис Automation:

```
PathName = iObject.PathName            Получить свойство (* )
PathName = iObject.GetPathName()       Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_PathName                  Получить свойство
(&PathName)
```

Примечание:

Свойство доступно только для чтения. Возвращает полное имя файла документа, состоящее из пути, имени и расширения файла.

ReadOnly – Только для чтения

Интерфейс...

Тип данных:BOOL

Синтаксис Automation:

```
ReadOnly = iObject.ReadOnly                 Получить свойство (* )
iObject.ReadOnly = ReadOnly                 Установить свойство (* )
ReadOnly = iObject.GetReadOnly()            Получить свойство (**)
iObject.SetReadOnly (ReadOnly)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ReadOnly (&ReadOnly)           Получить свойство
iObject->put_ReadOnly (ReadOnly)            Установить свойство
```

Примечание:

Позволяет получить и изменить свойство документа Только для чтения.

SpecificationDescriptions – Описания спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationDescriptions коллекции описаний спецификации

Синтаксис Automation:

```
SpecificationDescriptions =                 Получить свойство(* )
iObject.SpecificationDescriptions
SpecificationDescriptions =                 Получить свойство (**)
iObject.GetSpecificationDescriptions()
```

Синтаксис COM:

```
iObject->get_SpecificationDescriptions      Получить свойство
(&SpecificationDescriptions)
```

Примечание:

Свойство доступно только для чтения.

Visible – Видимость документа

Интерфейс...

Тип данных:BOOL

Синтаксис Automation:

Visible = iObject.Visible — Получить свойство (*)
Visible = iObject.GetVisible() — Получить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible) — Получить свойство

Значения свойства:

TRUE - документ открыт в видимом режиме,
FALSE - документ открыт в невидимом режиме.

Примечание:

Свойство доступно только для чтения. Возвращает свойство видимости документа.

IKompasDocument – методы

Close – Закрыть документ

Интерфейс...

Синтаксис Automation:

BOOL Close (DocumentCloseOptions closeOptions);

Синтаксис COM:

HRESULT Close ([in, defaultvalue(FALSE)] DocumentCloseOptions closeOptions, [out, retval] VARIANT_BOOL* pRes);

Входные параметры:

closeOptions - действия при закрытии измененного документа из перечисления DocumentCloseOptions.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Если документ не был изменен или его уже сохранили перед закрытием, то значение входного параметра будет проигнорировано.
2. Если документ был изменен, то возможны три варианта:

▼ закрыть документ без сохранения – kdDoNotSaveChanges (в этом случае все изменения будут потеряны),

- закрыть документ с сохранением – kdSaveChanges (в этом случае документ будет сначала сохранен, а потом закрыт),
- предоставить право выбора пользователю – kdPromptToSaveChanges (в этом случае на экран будет выведено сообщение о возможных действиях). Если закрываем в режиме kdPromptToSaveChanges, то пользователь может вообще отказаться от закрытия измененного документа, в этом случае метод вернет FALSE.

Save – Сохранить документ на диске

Интерфейс...

Синтаксис Automation:

void Save();

Синтаксис COM:

HRESULT Save();

Примечание:

Сохраняет все изменения в документе и снимает флаг измененности документа.

SaveAs– Сохранить документ под другим именем

Интерфейс...

Синтаксис Automation:

void SaveAs (LPCTSTR PathName);

Синтаксис COM:

HRESULT SaveAs ([in] BSTR PathName);

Входные параметры:

PathName	- полное имя файла документа, состоящее из пути, имени и расширения файла.

Примечание:

Позволяет сохранить документ на диск под другим именем.

UserDataStoragesMng – Интерфейс Менеджера пользовательских хранилищ

Интерфейс...

Тип данных: указатель на интерфейс IUserDataStoragesMng Менеджера пользовательских хранилищ

Синтаксис Automation:

UserDataStoragesMng = iObject.UserDataStoragesMng	Получить свойство (*)
UserDataStoragesMng = iObject.GetUserDataStoragesMng()	Получить свойство (**)

Синтаксис COM:

iObject->get_UserDataStoragesMng (&UserDataStoragesMng)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IKompasDocument2D

Базовый класс графических документов КОМПАС.

Иерархия:

IKompasAPIObject

IKompasDocument

IKompasDocument2D

IKompasDocument2D1

IPropertyKeeper

Примечание:

1. Данный интерфейс является базовым для интерфейсов графических документов КОМПАС API - чертежа и фрагмента.
2. Интерфейс можно получить из коллекции документов IDocuments.
3. Интерфейс может быть получен с помощью метода IUnknown QueryInterface от

- интерфейса документа IKompasDocument,
- интерфейса чертежа IDrawingDocument,
- интерфейса фрагмента IFragmentDocument.

4. С помощью метода IUnknown::QueryInterface у интерфейса можно получить дополнительные интерфейсы:

- IKompasDocument2D1,
- IPropertyKeeper.

IKompasDocument2D – свойства

ViewsAndLayersManager – Возвращает менеджер видов и слоев документа

Интерфейс...

Тип данных: IViewsAndLayersManager

Синтаксис Automation:

ViewsAndLayersManager = iObject.ViewsAndLayersManager	Получить свойство (*)

ViewsAndLayersManager = iObject.GetViewsAndLayersManager() — Получить свойство (**)

Синтаксис COM:

iObject->get_ViewsAndLayersManager (&ViewsAndLayersManager) — Получить свойство

Примечание:

Свойство доступно только для чтения.

IKompasDocument2D – методы

GetDrawingObjectNotifyResult – Получить интерфейс результатов редактирования объекта при обработке событий

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDrawingObjectNotifyResult();

Синтаксис COM:

HRESULT GetDrawingObjectNotifyResult ([out, retval] IDispatch**Result);

Возвращаемое значение:

указатель на интерфейс ksObject2DNotifyResult

Интерфейс IFragmentDocument

Интерфейс фрагмента.

IKompasAPIObject

IKompasDocument

IKompasDocument

IFragmentDocument

IFragmentDocument – свойства

IsSketch – Документ является эскизом

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - документ является эскизом,
FALSE - документ не является эскизом.

Синтаксис Automation:

IsSketch = iObject.IsSketch	Получить свойство (*)
IsSketch = iObject.GetIsSketch( )	Получить свойство (**)

Синтаксис COM:

iObject->get_IsSketch (&IsSketch)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

Интерфейс IDrawingDocument

Интерфейс чертежа.

Иерархия:

IKompasAPIObject

IKompasDocument

IKompasDocument2D

IDrawingDocument

IDrawingDocument – свойства

ChangeListDescriptions – Описания таблицы изменений

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationDescriptions

Синтаксис Automation:

Descriptions = Object.ChangeListDescriptions	Получить свойство (*)
Descriptions = Object.GetChangeListDescriptions()	Получить свойство (**)

Синтаксис COM:

Object.get_ChangeListDescriptions(&Descriptions)	Получить свойство

Примечание:

1. Позволяет получать и создавать объекты спецификации таблицы изменения.
2. Свойство доступно только для чтения.

SpecRough – Неуказанная шероховатость

Интерфейс...

Тип данных: указатель на интерфейс ISpecRough

Синтаксис Automation:

SpecRough = Object.SpecRough	Получить свойство (*)
SpecRough = Object.GetSpecRough()	Получить свойство (**)

Синтаксис COM:

Object.get_SpecRough(&SpecRough)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс неуказанной шероховатости.
2. Свойство доступно только для чтения.

TechnicalDemand – Технические требования

Интерфейс...

Тип данных: указатель на интерфейс ITechnicalDemand

Синтаксис Automation:

TechnicalDemand = Object.TechnicalDemand	Получить свойство (*)
TechnicalDemand = Object.GetTechnicalDemand()	Получить свойство (**)

Синтаксис COM:

Object.get_TechnicalDemand(&TechnicalDemand)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс технических требований.
2. Свойство доступно только для чтения.

Интерфейс IKompasDocument3D

Справка системы КОМПАС...

`kompas.chm::/260_Glava14_Osnovn_pon_trehmern_modelir.htm`

Базовый класс документов-моделей КОМПАС (скрытый).

Иерархия:

IKompasAPIObject

IKompasDocument

IKompasDocument3D

IExternalTessellationManager

ISelectionManager

IChooseManager

Интерфейс событий...

Синтаксис Automation:

IKompasDocument3D document3D (application.GetActiveDocument());

Синтаксис COM:

pApplication->get_ActiveDocument(&document3D);

Примечание:

1. Данный интерфейс является базовым для интерфейсов документов-моделей КОМПАС API – детали IPartDocument и сборки IAssemblyDocument.
2. Интерфейс можно получить у интерфейса приложения IApplication с помощью свойства IApplication::ActiveDocument или свойства IApplication::Documents.
3. У данного интерфейса посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif) можно получить дополнительные интерфейсы:

- IEmbodimentsManager
- IExternalTessellationManager
- ISelectionManager
- IChooseManager

IKompasDocument3D – свойства

AttributesEx – Массив атрибутов документа в виде массива SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

AttributesEx = Object.AttributesEx(Key1, Key2, Key3, Key4, Numb, Objects, SourcePart)	Получить свойство (*)
AttributesEx = Object.GetAttributesEx(Key1, Key2, Key3, Key4, Numb, Objects, SourcePart)	Получить свойство (**)

Синтаксис COM:

Object.get_AttributesEx(Key1, Key2, Key3, Key4, Numb, Objects, SourcePart, &AttributesEx)	Получить свойство

Входные параметры:

key1, key2, key3, key4	- ключи атрибута,
Numb	- уникальный номер типа атрибута,
Objects	- массив VT_ARRAY I VT_DISPATCH объектов, для которых назначается атрибут,
SourcePart	- указатель на интерфейс вставки детали IPart.

Примечание:

Свойство доступно только для чтения.

ChooseManager – Менеджер выбора (подсветки) объектов

Интерфейс...

Тип данных: указатель на интерфейс менеджера выбора (подсветки) объектов IChooseManager

Синтаксис Automation:

ChooseManager = Object.ChooseManager	Получить свойство (*)
ChooseManager = Object.GetChooseManager()	Получить свойство (**)

Синтаксис COM:

Object.get_ChooseManager(&ChooseManager)	Получить свойство

Свойство позволяет получать указатель на менеджер выбора (подсветки) объектов.

Примечание:

Свойство доступно только для чтения.

CreateObjectsInCurrentLocalCS – Создавать новые объекты в текущей ЛСК

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateObjectsInCurrentLocalCS = Object.CreateObjectsInCurrentLocalCS	Получить свойство (*)
Object.CreateObjectsInCurrentLocalCS = CreateObjectsInCurrentLocalCS	Установить свойство (*)
CreateObjectsInCurrentLocalCS = Object.GetCreateObjectsInCurrentLocalCS()	Получить свойство (**)
Object.SetCreateObjectsInCurrentLocalCS(CreateObjectsInCurrentLocalCS)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateObjectsInCurrentLocalCS(&CreateObjectsInCurrentLocalCS)	Получить свойство
Object.put_CreateObjectsInCurrentLocalCS(CreateObjectsInCurrentLocalCS)	Установить свойство

Примечание:

Свойство позволяет включить режим создания объектов модели в текущей системе координат. Для правильной работы библиотек нужно включить режим, создать нужные объекты и выключить режим.

Если флаг включен, то 3D точка, ломаная, сплайн, эскиз, импортированная поверхность, операция без истории, деталь заготовка через API будут создаваться в текущей локальной системе координат.

Параметры создания в локальной системе координат зависят от настроек системы.

См. также: ISystemSettings::ModelLocalCSCreateInAbsoluteCS и ISystemSettings::ModelLocalCSSetActive.

DrawMode – Режим визуализации

Интерфейс...

Тип данных: long

Значение свойства:

Из перечисления ViewMode

Синтаксис Automation:

long DrawMode = iObject.DrawMode	Получить свойство (*)
iObject.DrawMode = DrawMode	Установить свойство (*)
long DrawMode = iObject.GetDrawMode()	Получить свойство (**)
iObject.SetDrawMode(DrawMode)	Установить свойство (**)

Синтаксис COM:

iObject->get_DrawMode(&DrawMode)	Получить свойство
iObject->put_DrawMode(DrawMode)	Установить свойство

EnableUndo – Включить/отключить добавление команд в Undo

Интерфейс...

Тип данных: BOOL

Синтаксис:

EnableUndo = Object.EnableUndo	Получить свойство (*)
Object.EnableUndo = EnableUndo	Установить свойство (*)
EnableUndo = Object.GetEnableUndo()	Получить свойство (**)

Object.SetEnableUndo(EnableUndo) Установить свойство (**)

Синтаксис COM:

Object.get_EnableUndo(&EnableUndo) Получить свойство
Object.put_EnableUndo(EnableUndo) Установить свойство

HideAllAuxiliaryGeom – Скрыть / показать все вспомогательные объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HideAllAuxiliaryGeom = Object.HideAllAuxiliaryGeom Получить свойство (*)
Object.HideAllAuxiliaryGeom = HideAllAuxiliaryGeom Установить свойство (*)
HideAllAuxiliaryGeom = Object.GetHideAllAuxiliaryGeom() Получить свойство (**)
Object.SetHideAllAuxiliaryGeom(HideAllAuxiliaryGeom) Установить свойство (**)

Синтаксис COM:

Object.get_HideAllAuxiliaryGeom(&HideAllAuxiliaryGeom) Получить свойство
Object.put_HideAllAuxiliaryGeom(HideAllAuxiliaryGeom) Установить свойство

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllAxis – Скрыть / показать конструктивные оси

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - конструктивные оси скрыты,
FALSE - конструктивные оси показаны.

Синтаксис Automation:

BOOL HideAllAxis = iObject.HideAllAxis Получить свойство (*)
iObject.HideAllAxis = HideAllAxis Установить свойство (*)
BOOL HideAllAxis = iObject.GetHideAllAxis() Получить свойство (**)
iObject.SetHideAllAxis (HideAllAxis) Установить свойство (**)

Синтаксис COM:

iObject->get_HideAllAxis (&HideAllAxis) Получить свойство
iObject->put_HideAllAxis (HideAllAxis) Установить свойство

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllControlPoints – Скрыть / показать контрольные точки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - контрольные точки скрыты,
FALSE - контрольные точки показаны.

Синтаксис Automation:

BOOL HideAllControlPoints = iObject.HideAllControlPoints Получить свойство (*)
iObject.HideAllControlPoints = HideAllControlPoints Установить свойство (*)
BOOL HideAllControlPoints = iObject.GetHideAllControlPoints() Получить свойство (**)
iObject.SetHideAllControlPoints (HideAllControlPoints) Установить свойство (**)

Синтаксис COM:

iObject->get_HideAllControlPoints (&HideAllControlPoints) Получить свойство
iObject->put_HideAllControlPoints (HideAllControlPoints) Установить свойство

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllCurves – Скрыть / показать пространственные кривые

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - пространственные кривые скрыты,
FALSE - пространственные кривые показаны.

Синтаксис Automation:

```
BOOL HideAllCurves = iObject.HideAllCurves          Получить свойство (* )
iObject.HideAllCurves = HideAllCurves               Установить свойство (* )
BOOL HideAllCurves = iObject.GetHideAllCurves()     Получить свойство (**)
iObject.SetHideAllCurves (HideAllCurves)            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllCurves (&HideAllCurves)     Получить свойство
iObject->put_HideAllCurves (HideAllCurves)      Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllDesignations – Скрыть / показать условные обозначения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
HideAllDesignations = Object.HideAllDesignations          Получить свойство (* )
Object.HideAllDesignations = HideAllDesignations          Установить свойство (* )
HideAllDesignations =                                     Получить свойство (**)
Object.GetHideAllDesignations()
Object.SetHideAllDesignations( HideAllDesignations )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HideAllDesignations( &HideAllDesignations )     Получить свойство
Object.put_HideAllDesignations(  HideAllDesignations )     Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllDimensions – Скрыть / показать размеры

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
HideAllDimensions = Object.HideAllDimensions                Получить свойство (* )
Object.HideAllDimensions = HideAllDimensions                Установить свойство (* )
HideAllDimensions = Object.GetHideAllDimensions()           Получить свойство (**)
Object.SetHideAllDimensions( HideAllDimensions )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HideAllDimensions( &HideAllDimensions )      Получить свойство
Object.put_HideAllDimensions(  HideAllDimensions )      Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllPlaces – Скрыть / показать начала координат

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - начала координат скрыты,
FALSE       - начала координат показаны.
```

Синтаксис Automation:

```
BOOL HideAllPlaces = iObject.HideAllPlaces                  Получить свойство (* )
iObject.HideAllPlaces = HideAllPlaces                       Установить свойство (* )
BOOL HideAllPlaces = iObject.GetHideAllPlaces()             Получить свойство (**)
iObject.SetHideAllPlaces (HideAllPlaces)                    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllPlaces (&HideAllPlaces)     Получить свойство
iObject->put_HideAllPlaces (HideAllPlaces)      Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllPlanes – Скрыть / показать конструктивные плоскости

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - конструктивные плоскости скрыты,
FALSE - конструктивные плоскости показаны.

Синтаксис Automation:

```
BOOL HideAllPlanes = iObject.HideAllPlanes          Получить свойство (* )
iObject.HideAllPlanes = HideAllPlanes               Установить свойство (* )
BOOL HideAllPlanes = iObject.GetHideAllPlanes()     Получить свойство (**)
iObject.SetHideAllPlanes (HideAllPlanes)            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllPlanes (&HideAllPlanes)     Получить свойство
iObject->put_HideAllPlanes (HideAllPlanes)      Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllSketches – Скрыть / показать эскизы

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - эскизы скрыты,
FALSE - эскизы показаны.

Синтаксис Automation:

```
BOOL HideAllSketches = iObject.HideAllSketches          Получить свойство (* )
iObject.HideAllSketches = HideAllSketches               Установить свойство (* )
BOOL HideAllSketches = iObject.GetHideAllSketches()     Получить свойство (**)
iObject.SetHideAllSketches (HideAllSketches)            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllSketches (&HideAllSketches)     Получить свойство
iObject->put_HideAllSketches (HideAllSketches)      Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllSurfaces – Скрыть / показать поверхности

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - поверхности скрыты,
FALSE - поверхности показаны.

Синтаксис Automation:

```
BOOL HideAllSurfaces = iObject.HideAllSurfaces     Получить свойство (* )
iObject.HideAllSurfaces = HideAllSurfaces          Установить свойство (* )
BOOL HideAllSurfaces =                             Получить свойство (**)
iObject.GetHideAllSurfaces()
iObject.SetHideAllSurfaces (HideAllSurfaces)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllSurfaces (&HideAllSurfaces)    Получить свойство
iObject->put_HideAllSurfaces (HideAllSurfaces)     Установить свойство
```

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllThreads – Резьбы

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - резьбы скрыты,
FALSE - резьбы показаны.

Синтаксис Automation:

```
BOOL HideAllThreads = iObject.HideAllThreads       Получить свойство (* )
iObject.HideAllThreads = HideAllThreads            Установить свойство (* )
BOOL HideAllThreads =                              Получить свойство (**)
iObject.GetHideAllThreads()
iObject.SetHideAllThreads (HideAllThreads)         Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_HideAllThreads (&HideAllThreads)      Получить свойство
```

iObject->put_HideAllThreads (HideAllThreads) Установить свойство

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideInComponentsMode – Режим скрытия вспомогательной геометрии в компонентах

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HideInComponentsMode = Object.HideInComponentsMode	Получить свойство (*)
Object.HideInComponentsMode = HideInComponentsMode	Установить свойство (*)
HideInComponentsMode = Object.GetHideInComponentsMode()	Получить свойство (**)
Object.SetHideInComponentsMode(HideInComponentsMode)	Установить свойство (**)

Синтаксис COM:

Object.get_HideInComponentsMode(&HideInComponentsMode)	Получить свойство
Object.put_HideInComponentsMode(HideInComponentsMode)	Установить свойство

Примечание:

Флаг позволяет для сборки включить режим работы с флагами скрытия вспомогательной геометрии во вставках или в документе.

Если флаг включен, то свойства: HideAllPlanes, HideAllAxis, HideAllSketches, HideAllPlaces, HideAllSurfaces, HideAllThreads, HideAllCurves, HideAllControlPoints, HideAllDimensions, HideAllDesignations, HideAllAuxiliaryGeom
управляют признаками скрытия вспомогательной геометрии во вставках.

Perspective – Скрыть / показать перспективу

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- показывать перспективу,
FALSE	- не показывать перспективу.

Синтаксис Automation:

BOOL Perspective = iObject.Perspective	Получить свойство (*)
iObject.Perspective = Perspective	Установить свойство (*)
BOOL Perspective = iObject.GetPerspective()	Получить свойство (**)
iObject.SetPerspective (Perspective)	Установить свойство (**)

Синтаксис COM:

```
iObject->get_Perspective (&Perspective)      Получить свойство
iObject->put_Perspective (Perspective)       Установить свойство
```

SelectionManager – Менеджер селектированных объектов

Интерфейс...

Тип данных: указатель на интерфейс менеджера селектированных объектов ISelectionManager

Синтаксис Automation:

```
SelectionManager = Object.SelectionManager          Получить свойство (* )
SelectionManager = Object.GetSelectionManager()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SelectionManager (&SelectionManager)     Получить свойство
```

Свойство позволяет получать указатель на менеджер селектированных объектов.

Примечание:

Свойство доступно только для чтения.

ShadedWireframe – Полутоновое изображение с каркасом

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE     - каркас показан,
FALSE    - каркас скрыт.
```

Синтаксис Automation:

```
BOOL ShadedWireframe = iObject.ShadedWireframe      Получить свойство (* )
iObject.ShadedWireframe = ShadedWireframe           Установить свойство (* )
ShadedWireframe = iObject.GetShadedWireframe()      Получить свойство (**)
iObject.SetShadedWireframe (ShadedWireframe)        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ShadedWireframe (&ShadedWireframe)     Получить свойство
iObject->put_ShadedWireframe (ShadedWireframe)      Установить свойство
```

TechnicalDemand3D – Технические требования 3D

Интерфейс...

Тип данных: указатель на интерфейс ITechnicalDemand3D

Синтаксис Automation:

TechnicalDemand3D = Object.TechnicalDemand3D	Получить свойство (*)
TechnicalDemand3D = Object.GetTechnicalDemand3D()	Получить свойство (**)

Синтаксис COM:

Object.get_TechnicalDemand3D(&TechnicalDemand3D)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на интерфейс технических требований 3D.

ToleranceMode – Включить/отключить режим пересчета по допуску

Интерфейс...

Синтаксис Automation:

ToleranceMode = Object.ToleranceMode	Получить свойство (*)
Object.ToleranceMode = ToleranceMode	Установить свойство (*)
ToleranceMode = Object.GetToleranceMode()	Получить свойство (**)
Object.SetToleranceMode(ToleranceMode)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceMode(&ToleranceMode)	Получить свойство
Object.put_ToleranceMode(ToleranceMode)	Установить свойство

Примечание:

Свойство позволяет включать и отключать режим перестроения по допуску.

TopPart – Верхний компонент (сам документ)

Интерфейс...

Тип данных: указатель на интерфейс IPart7

Синтаксис Automation:

TopPart = iObject.TopPart	Получить свойство (*)
TopPart = iObject.GetTopPart()	Получить свойство (**)

Синтаксис COM:

iObject->get_TopPart(&TopPart)	Получить свойство

Примечание:

Свойство доступно только для чтения.

UndoContainer – Включить/отключить объединение команд в Undo

Интерфейс...

Тип данных: BOOL

Синтаксис:

UndoContainer = Object.UndoContainer	Получить свойство (*)
Object.UndoContainer = UndoContainer	Установить свойство (*)
UndoContainer = Object.GetUndoContainer()	Получить свойство (**)
Object.SetUndoContainer(UndoContainer)	Установить свойство (**)

Синтаксис COM:

Object.get_UndoContainer(&UndoContainer)	Получить свойство
Object.put_UndoContainer(UndoContainer)	Установить свойство

IKompasDocument3D – методы

CreateAttrEx – Создать атрибут по номеру типа атрибута из библиотеки libname

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateAttrEx(double AttrID, BSTR Libname, VARIANT Objects, IPart7 * SourcePart);

Синтаксис COM:

HRESULT CreateAttrEx(double AttrID, BSTR Libname, VARIANT Objects, IPart7 * SourcePart, IAttribute ** Result);

Возвращаемое значение:

- указатель на интерфейс IAttribute

Входные параметры:

AttrID	- уникальный номер типа атрибута,
Libname	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется из текущего документа,
Objects	- массив VT_ARRAY | VT_DISPATCH объектов, для которых назначается атрибут,
SourcePart	- указатель на интерфейс вставки детали IPart.

DeleteHistory – Удалить историю построения

Интерфейс...

Синтаксис Automation:

BOOL DeleteHistory();

Синтаксис COM:

HRESULT DeleteHistory([out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет удалить историю построения.

RebuildDocument – Перестроить документ

Интерфейс...

Синтаксис Automation:

BOOL RebuildDocument();

Синтаксис COM:

HRESULT RebuildDocument(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет вызвать перестроение 3D-документа.

Интерфейс IAssemblyDocument

Интерфейс сборки.

IKompasAPIObject

IKompasDocument

IKompasDocument3D

IAssemblyDocument

ILoadCombination

Примечание:

Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс ILoadCombination.

IAssemblyDocument – свойства

DismantleMode – Разнесенная сборка

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
DismantleMode = Object.DismantleMode                Получить свойство (* )
Object.DismantleMode = DismantleMode                Установить свойство (* )
DismantleMode = Object.GetDismantleMode()           Получить свойство (**)
Object.SetDismantleMode( DismantleMode )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DismantleMode( &DismantleMode )          Получить свойство
Object.put_DismantleMode(  DismantleMode )          Установить свойство
```

Свойство позволяет включать и отключать режим разнесенной сборки.

Интерфейс IPartDocument

Интерфейс детали.

IKompasAPIObject

IKompasDocument

IKompasDocument3D

IPartDocument

Интерфейс ITextDocument

Интерфейс текстового документа.

IKompasAPIObject

IKompasDocument

ITextDocument

ITextDocument – свойства

BlocksGabarits – Габариты внутренних областей листов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
BlocksGabarits = Object.BlocksGabarits          Получить свойство (* )
BlocksGabarits = Object.GetBlocksGabarits()     Получить свойство (**)
```

Синтаксис COM:

Object.get_BlocksGabarits(&BlocksGabarits)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BlocksStartLineNumbers – Получить список номеров строк начала страниц

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BlocksStartLineNumbers = Object.BlocksStartLineNumbers	Получить свойство (*)
BlocksStartLineNumbers = Object.GetBlocksStartLineNumbers()	Получить свойство (**)

Синтаксис COM:

Object.get_BlocksStartLineNumbers(&BlocksStartLineNumbers)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SectionsStartLineNumbers – Получить список номеров строк начала разделов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

SectionsStartLineNumbers = Object.SectionsStartLineNumbers	Получить свойство (*)
SectionsStartLineNumbers = Object.GetSectionsStartLineNumbers()	Получить свойство (**)

Синтаксис COM:

Object.get_SectionsStartLineNumbers(&SectionsStartLineNumbers)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Text – Текст

Интерфейс...

Тип данных: Указатель на интерфейс IText

Синтаксис Automation:

```
Text = Object.Text                    Получить свойство (* )
Text = Object.GetText()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Text( &Text )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ITextDocument – методы

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

```
TRUE      - в случае успешного завершения,
FALSE     - в случае неудачи.
```

Интерфейс ISpecificationDocument

Интерфейс документа спецификации.

Иерархия:

IKompasAPIObject

IKompasDocument

ISpecificationDocument

ISpecificationDocument – свойства

AttachedDocuments – Управление сборкой. Присоединенные документы

Интерфейс...

Тип данных: IAttachedDocuments

Синтаксис Automation:

AttachedDocuments = Object.AttachedDocuments	Получить свойство (*)
AttachedDocuments = Object.GetAttachedDocuments()	Получить свойство (**)

Синтаксис COM:

Object.get_AttachedDocuments(&AttachedDocuments)	Получить свойство

Примечание:

Позволяет получать и установить список документов, подключенных в управлении сборкой. Свойство IAttachedDocument_Comment для документов, подключенных через управление сборкой, не используется.

Crossed – Признак необходимости перестроения документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Crossed = Object.Crossed	Получить свойство (*)
Crossed = Object.GetCrossed()	Получить свойство (**)

Синтаксис COM:

Object.get_Crossed(&Crossed)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

ISpecificationDocument – методы

RebuildDocument – Перестроить документ

Интерфейс...

Синтаксис Automation:

BOOL RebuildDocument();

Синтаксис COM:

HRESULT RebuildDocument(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IKompasDocument1

Дополнительный интерфейс IKompasDocument.

Иерархия:

IKompasAPIObject

IKompasDocument1

Примечание:

Интерфейс является дополнительным к интерфейсу документа IKompasDocument. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IKompasDocument1 – свойства

Attributes – Массив атрибутов документа в виде массива SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Attributes = Object.Attributes(long Key1, Получить свойство (*)
long Key2,
long Key3,
long Key4,
double Numb,
VARIANT Objects)

Attributes = Object.GetAttributes(long Key1, Получить свойство (**)
long Key2,
long Key3,
long Key4,
double Numb,
VARIANT Objects)

Синтаксис COM:

Object.get_Attributes(long Key1, long Key2, long Key3, long Key4, double Numb, VARIANT Objects, &Attributes)	Получить свойство

Входные параметры:

Key1, Key2, Key3, Key4	- ключи атрибута,
Numb	- уникальный номер типа атрибута,
Objects	- массив VT_ARRAY | VT_DISPATCH объектов, атрибуты которых нужно получить, или указатель на объект VT_DISPATCH.

Примечание:

1. Свойство позволяет получать атрибуты документа. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.
3. Чтобы получить все объекты с заданным атрибутом, надо в качестве параметра Objects передать VARIANT с типом VT_EMPTY
4. Чтобы получить атрибуты документа, надо в качестве параметра Objects передать указатель на документ (тип VT_DISPATCH).
5. Чтобы получить все атрибуты объекта, надо в качестве параметра Numb передать 0.
6. Если в качестве параметра Objects будет передан массив объектов, то вернутся групповые атрибуты для этих объектов.

Author – Автор

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Author = Object.Author	Получить свойство (*)
Object.Author = Author	Установить свойство (*)
Author = Object.GetAuthor()	Получить свойство (**)
Object.SetAuthor(Author)	Установить свойство (**)

Синтаксис COM:

Object.get_Author(&Author)	Получить свойство
Object.put_Author(Author)	Установить свойство

Comment – Комментарий

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Comment = Object.Comment	Получить свойство (*)
Object.Comment = Comment	Установить свойство (*)
Comment = Object.GetComment()	Получить свойство (**)
Object.SetComment(Author)	Установить свойство (**)

Синтаксис COM:

Object.get_Comment(&Comment)	Получить свойство
Object.put_Comment(Comment)	Установить свойство

CreationDate – Получить дату создания документа

Интерфейс...

Тип данных: double

Синтаксис Automation:

CreationDate = Object.CreationDate	Получить свойство (*)
Object.CreationDate = CreationDate	Установить свойство (*)
CreationDate = Object.GetCreationDate()	Получить свойство (**)
Object.SetCreationDate(CreationDate)	Установить свойство (**)

Синтаксис COM:

Object.get_CreationDate(&CreationDate)	Получить свойство
Object.put_CreationDate(CreationDate)	Установить свойство

DocumentTypeId – Идентификатор типа документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DocumentTypeId = Object.DocumentTypeId	Получить свойство (*)
DocumentTypeId = Object.GetDocumentTypeId()	Получить свойство (**)

Синтаксис COM:

Object.get_DocumentTypeId(&DocumentTypeId)	Получить свойство

ExternalFilesNames – Список внешних файлов

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_BSTR

Синтаксис Automation:

ExternalFilesNames = Object.ExternalFilesNames	Получить свойство (*)
ExternalFilesNames = Object.GetExternalFilesNames()	Получить свойство (**)

Синтаксис COM:

Object.get_ExternalFilesNames(&ExternalFilesNames)	Получить свойство

Примечание:

1. Свойство позволяет получать список внешних файлов.
2. Свойство доступно только для чтения.

LastChangeDate – Получить дату последнего изменения документа

Интерфейс...

Тип данных: double

Синтаксис Automation:

LastChangeDate = Object.LastChangeDate	Получить свойство (*)
Object.LastChangeDate = LastChangeDate	Установить свойство (*)
LastChangeDate = Object.GetLastChangeDate()	Получить свойство (**)
Object.SetLastChangeDate(LastChangeDate)	Установить свойство (**)

Синтаксис COM:

Object.get_LastChangeDate(&LastChangeDate)	Получить свойство
Object.put_LastChangeDate(LastChangeDate)	Установить свойство

Metadata – Метаданные

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Metadata = Object.Metadata	Получить свойство (*)
Object.Metadata = Metadata	Установить свойство (*)
Metadata = Object.GetMetadata()	Получить свойство (**)

Object.SetMetadata(Metadata) Установить свойство (**)

Синтаксис COM:

Object.get_Metadata(&Metadata Получить свойство
Object.put_Metadata(Metadata) Установить свойство

Свойство позволяет получить и задать метаданные документа.

ObjectsByAttr – Массив объектов документа в виде массива SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Attributes = Object.ObjectsByAttr(long Получить свойство (*)
Key1,
long Key2,
long Key3,
long Key4,
double Numb,
VARIANT ObjectType)
Attributes = Object.ObjectsByAttr(long Получить свойство (**)
Key1,
long Key2,
long Key3,
long Key4,
double Numb,
VARIANT ObjectType)

Синтаксис COM:

Object.get_ObjectsByAttr(long Key1, Получить свойство
long Key2,
long Key3,
long Key4,
double Numb,
VARIANT ObjectType,
&Attributes)

Входные параметры:

Key1, Key2, Key3, Key4 - ключи атрибута,
Numb - уникальный номер типа атрибута,
ObjectType - тип объектов.

Примечание:

1. Свойство позволяет получать объекты с заданным типом атрибута. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

OpenVersion – Версия файла, с которой документ был сохранен

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
OpenVersion = Object.OpenVersion                Получить свойство (* )
OpenVersion = Object.GetOpenVersion()           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_OpenVersion( &OpenVersion )          Получить свойство
```

Примечание:

Свойство доступно только для чтения

Свойство возвращает номер версии Компас, с которой документ был сохранен последний раз в случае открытия документа с диска.

Если документ сохранен в официальных версиях дистрибутива Компас, свойство вернет значение из перечисления ksSaveDocumentVersionEnum.

Если документ сохранен в промежуточных версиях, может вернуться числовая версия потока.

Organization – Организация

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Organization = Object.Organization              Получить свойство (* )
Object.Organization = Organization              Установить свойство (* )
Organization = Object.GetOrganization()         Получить свойство (**)
Object.SetOrganization( Organization )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Organization( &Organization )        Получить свойство
Object.put_Organization(  Organization )        Установить свойство
```

IKompasDocument1 – методы

ActivateToolbarSet – Активизировать набор по идентификатору

Интерфейс...

Синтаксис Automation:

BOOL ActivateToolbarSet(BSTR TolbrSetId);

Синтаксис COM:

HRESULT ActivateToolbarSet(BSTR TolbrSetId, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALS E	- в случае неудачи.

Входные параметры:

TolbrSetId	- строковый идентификатор набора панелей.

ApplyMetadataFromFile – Применить метаданные из файла

Интерфейс...

Синтаксис Automation:

BOOL ApplyMetadataFromFile(BSTR FileName);

Синтаксис COM:

HRESULT ApplyMetadataFromFile(BSTR FileName, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

FileName	- имя файла.

CreateAttr– Создать атрибут по номеру типа атрибута из библиотеки libname

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateAttr(double AttrID,
BSTR Libname,
VARIANT Objects);

Синтаксис COM:

HRESULT CreateAttr(AttrID,
Libname,
VARIANT Objects,
IAttribute ** Result);

Входные параметры:

AttrID	- уникальный номер типа атрибута,
Libname	- имя библиотеки типов атрибутов,
Objects	- объект или массив объектов, к которым нужно добавить атрибут (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

указатель на интерфейс атрибута IAttribute.

Примечание:

Метод позволяет создать атрибут по номеру типа из заданной библиотеки.

Delete – Удалить объект или объекты

Интерфейс...

Синтаксис Automation:

BOOL Delete(VARIANT Objects);

Синтаксис COM:

HRESULT Delete(VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов, которые нужно удалить (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить объект или объекты.

GetExternalFilesNamesEx – Получить список внешних файлов в виде SAFEARRAY BSTR – (VT_ARRAY | VT_BSTR) и их типов в виде SAFEARRAY long – (VT_ARRAY | VT_I4)

Интерфейс...

Синтаксис Automation:

BOOL GetExternalFilesNamesEx(BOOL allFiles, VARIANT * Files, VARIANT * FilesTypes);

Синтаксис COM:

BOOL GetExternalFilesNamesEx(BOOL allFiles, VARIANT * Files, VARIANT * FilesTypes);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

allFiles - TRUE - рекурсивно по всем файлам,
- FALSE - только по внешнему уровню.

Выходные параметры:

Files - массив VT_ARRAY I VT_BSTR имен файлов,
FilesTypes - массив VT_ARRAY I VT_I4 типов файлов (из перечисления ksExternalFilesTypesEnum).

GetInterface – Получить вспомогательный интерфейс

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetInterface(long Type);

Синтаксис COM:

HRESULT GetInterface(KompasAPIObjectTypeEnum Type,
IKompasAPIObject ** Result);

Входные параметры:

Type - тип интерфейса из перечисления KompasAPIObjectTypeEnum.

Возвращаемое значение:

- указатель на интерфейс IKompasAPIObject.

Примечание:

Метод позволяет получить вспомогательный интерфейс.

В зависимости от значения параметра Type возвращается указатель на интерфейс:

- в 2D документе:
  - ksObjectCopyObjectParam - параметры копирования ICopyObjectParam
  - ksObjectCurveCopyObjectParam - параметры копирования по кривой ICurveCopyObjectParam
  - ksObjectCircleCopyObjectParam - параметры копирования по окружности ICircleCopyObjectParam
  - ksObjectCircularCopyObjectParam - параметры копирования по концентрической сетке ICircularCopyObjectParam
  - ksObjectMeshCopyObjectParam - параметры копирования по сетке IMeshCopyObjectParam
  - ksObjectFindObjectParameters - интерфейс расширенного поиска объектов IFindObjectParameters
- в 3D документе:
  - case ksObjectReportProcess - интерфейс для управления процессом **Создать отчет** IReportProcess

RedrawDocument – Перерисовать окна документа

Интерфейс...

Синтаксис Automation:

BOOL RedrawDocument(ksRedrawDocumentModeEnum Mode);

Синтаксис COM:

HRESULT RedrawDocument(ksRedrawDocumentModeEnum Mode, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Mode - режим перерисовки окон документа.

ReportPropertiesMultieditMode – Включить/Выключить режим массовой работы со свойствами объектов

Интерфейс...

Синтаксис Automation:

BOOL ReportPropertiesMultieditMode(BOOL On, BOOL UpdateProps);

Синтаксис COM:

HRESULT ReportPropertiesMultieditMode(BOOL On, BOOL UpdateProps, BOOL * Result);

Входные параметры:

On - включить режим,
UpdateProps - применить изменение свойств.

SaveAsEx – Сохранить документ под другим именем

Интерфейс...

Синтаксис Automation:

BOOL SaveAsEx(BSTR PathName, long saveMode);

Синтаксис COM:

HRESULT SaveAsEx(BSTR PathName, long saveMode, BOOL * Result);

Входные параметры:

PathName - новое имя файла документа,
saveMode - версия для сохранения.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ViewEditAttr – Открыть диалог для просмотра атрибутов объекта

Интерфейс...

Синтаксис Automation:

BOOL ViewEditAttr(OLE_HANDLE HWnd,
VARIANT Objects);

Синтаксис COM:

HRESULT ViewEditAttr(OLE_HANDLE HWnd,
VARIANT Objects,
BOOL * Result);

Входные параметры:

HWnd - дескриптор главного окна КОМПАС (можно получить с помощью функции GetHWindow,
Objects - объект или массив объектов, атрибуты которых нужно просмотреть (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

Метод позволяет вывести диалог для просмотра атрибутов объекта.

WriteMetadataToFile – Записать метаданные в файл

Интерфейс...

Синтаксис Automation:

BOOL WriteMetadataToFile(BSTR FileName);

Синтаксис COM:

HRESULT WriteMetadataToFile(BSTR FileName, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

FileName - имя файла.

Интерфейс IKompasDocument2D1

Дополнительный интерфейс IKompasDocument2D.

Иерархия:

IKompasAPIObject

IKompasDocument2D1

Примечание:

Интерфейс является дополнительным к интерфейсу документа IKompasDocument2D. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IKompasDocument2D1 – свойства

ChooseManager – Менеджера выбора (подсветки) объектов

Интерфейс...

Тип данных: указатель на интерфейс менеджера выбора IChooseManager

Синтаксис Automation:

ChooseManager = Object.ChooseManager Получить свойство (*)
ChooseManager = Object.GetChooseManager() Получить свойство (**)

Синтаксис COM:

Object.get_ChooseManager(&ChooseManager) — Получить свойство

Примечание:

1. Свойство позволяет получать указатель на менеджер выбора.
2. Свойство доступно только для чтения.

CurrentGroup – Текущая группа

Интерфейс...

Тип данных: указатель на интерфейс группы IDrawingGroup

Синтаксис Automation:

CurrentGroup = Object.CurrentGroup — Получить свойство (*)
CurrentGroup = Object.GetCurrentGroup() — Получить свойство (**)

Синтаксис COM:

Object.get_CurrentGroup(&CurrentGroup) — Получить свойство

Примечание:

1. Свойство позволяет получать указатель на интерфейс текущей группы.
2. Свойство доступно только для чтения.

DrawingGroups – Коллекция групп

Интерфейс...

Тип данных: указатель на интерфейс коллекции групп IDrawingGroups

Синтаксис Automation:

DrawingGroups = Object.DrawingGroups — Получить свойство (*)
DrawingGroups = Object.GetDrawingGroups() — Получить свойство (**)

Синтаксис COM:

Object.get_DrawingGroups(&DrawingGroups) — Получить свойство

Примечание:

1. Свойство позволяет получать указатель на коллекцию групп.
2. Свойство доступно только для чтения.

EditMacroObject – Редактируемый макроэлемент

Интерфейс...

Тип данных: указатель на интерфейс макроэлемента IMacroObject

Синтаксис Automation:

EditMacroObject = Object.EditMacroObject	Получить свойство (*)
EditMacroObject = Object.GetEditMacroObject()	Получить свойство (**)

Синтаксис COM:

Object.get_EditMacroObject(&EditMacroObject)	Получить свойство

Примечание:

1. Свойство позволяет получать указатель на редактируемый макроэлемент.
2. Свойство доступно только для чтения.

EditMacroVisibleRegime – Находится ли документ в режиме редактирования макроэлемента

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EditMacroVisibleRegime Object.EditMacroVisibleRegime	=	Получить свойство (*)
EditMacroVisibleRegime Object.GetEditMacroVisibleRegime()	=	Получить свойство (**)

Синтаксис COM:

Object.get_EditMacroVisibleRegime(&EditMacroVisibleRegime)	Получить свойство

Возвращаемое значение:

TRUE	- режим редактирования макроэлемента включен,
FALSE	- режим редактирования макроэлемента отключен.

Примечание:

1. Свойство доступно только для чтения.
2. Свойство используется при редактировании макроэлементов.

LibProcess – Получить объект процесса

Интерфейс...

Тип данных: Указатель на интерфейс IProcess2D

Синтаксис Automation:

LibProcess = Object.LibProcess(ProcessType)	Получить свойство (*)
LibProcess = Object.GetLibProcess(ProcessType)	Получить свойство (**)

Синтаксис COM:

Object.get_LibProcess(ProcessType, &LibProcess)	Получить свойство

Примечание:

Свойство доступно только для чтения.

NamedGroups – Коллекция именованных групп

Интерфейс...

Тип данных: указатель на интерфейс коллекции групп IDrawingGroups

Синтаксис Automation:

NamedGroups = Object.NamedGroups	Получить свойство (*)
NamedGroups = Object.GetNamedGroups()	Получить свойство (**)

Синтаксис COM:

Object.get_NamedGroups(&NamedGroups)	Получить свойство

Примечание:

1. Свойство позволяет получать указатель на коллекцию именованных групп.
2. Свойство доступно только для чтения.

SelectionManager – Менеджер выделенных объектов

Интерфейс...

Тип данных: - указатель на интерфейс менеджера выделенных объектов ISelectionManager

Синтаксис Automation:

SelectionManager = Object.SelectionManager	Получить свойство (*)
SelectionManager = Object.GetSelectionManager()	Получить свойство (**)

Синтаксис COM:

Object.get_SelectionManager(&SelectionManager)	Получить свойство

Примечание:

1. Свойство позволяет получать указатель на менеджер выделенных объектов.
2. Свойство доступно только для чтения.

Variable – Получить параметрическую переменную по имени, индексу или указателю на размер

Интерфейс...

Тип данных: указатель на интерфейс параметрической переменной IVariable7

Синтаксис Automation:

Variable = Object.Variable (VARIANT Index)	Получить свойство (*)
Variable = Object.GetVariable (VARIANT Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Variable (VARIANT Index, &Variable)	Получить свойство

Входные параметры:

Index	- индекс, имя или указатель на размер параметрической переменной.

Свойство позволяет получать интерфейс параметрической переменной.

Примечание:

Свойство доступно только для чтения.

Variables – Получить массив параметрических переменных

Интерфейс...

Тип данных: SAFEARRAY VT_DISPATCH

Синтаксис Automation:

Variables = Object.Variables Получить свойство (*)
Variables = Object.GetVariables() Получить свойство (**)

Синтаксис COM:

Object.get_Variables(&Variables) Получить свойство

Свойство позволяет получать массив параметрических переменных.

Примечание:

Свойство доступно только для чтения.

VariablesCount – Получить количество параметрических переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

VariablesCount = Object.VariablesCount Получить свойство (*)
VariablesCount = Object.GetVariablesCount() Получить свойство (**)

Синтаксис COM:

Object.get_VariablesCount(&VariablesCount) Получить свойство

Свойство позволяет получать количество параметрических переменных.

Примечание:

Свойство доступно только для чтения.

VariableTable – Таблица переменных

Интерфейс...

Тип данных: указатель на интерфейс таблицы переменных IVariableTable

Синтаксис Automation:

VariableTable = Object.VariableTable Получить свойство (*)
VariableTable = Object.GetVariableTable() Получить свойство (**)

Синтаксис COM:

Object.get_VariableTable(&VariableTable) Получить свойство

Свойство позволяет получать интерфейс таблицы переменных.

Примечание:

Свойство доступно только для чтения.

IKompasDocument2D1 – методы

AddVariable – Создать переменную

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddVariable();

Синтаксис COM:

HRESULT AddVariable(IVariable7 ** Result);

Возвращаемое значение:

- указатель на интерфейс параметрической переменной IVariable7

Метод позволяет создать переменную (добавить переменную в массив переменных и документ).

CopyObjects – Копировать объекты

Интерфейс...

Синтаксис Automation:

VARIANT CopyObjects(VARIANT Objects,
LPDISPATCH * params);

Синтаксис COM:

HRESULT CopyObjects(VARIANT Objects,
ICopyObjectParam * params,
VARIANT * Result);

Входные параметры:

Objects - объект или массив объектов для копирования (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH),

params - указатель на интерфейс параметров копирования ICopyObjectParam.

Возвращаемое значение:

Объект или массив копий объектов (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Примечание:

Метод позволяет создать атрибут по номеру типа атрибута.

Способ копирования зависит от типа интерфейса параметров копирования:

- ICopyObjectParam - параметры копирования,
- ICurveCopyObjectParam - параметры копирования по кривой,
- ICircleCopyObjectParam - параметры копирования по окружности,
- ICircularCopyObjectParam - параметры копирования по концентрической сетке,
- IMeshCopyObjectParam - параметры копирования по сетке.

Нужный интерфейс параметров копирования можно получить с помощью функции IKompasDocument1::GetInterface.

CreateHyperLink – Создать гиперссылку

Интерфейс...

Синтаксис Automation:

BOOL CreateHyperLink(VARIANT Objects,
ksHyperLinkTypeEnum Type,
BSTR Text,
IDrawingObject * LinkObject,
longLevel);

Синтаксис COM:

HRESULT CreateHyperLink(VARIANT Objects,
ksHyperLinkTypeEnum Type,
BSTR Text,
IDrawingObject * LinkObject,
long Level,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- VT_DISPATCH или массив VT_ARRAY | VT_DISPATCH объектов, содержащих ссылку,
Type	- тип ссылки,
Text	- текст ссылки,
LinkObject	- объект, на который сделана ссылка,
Level	- уровень.

DeleteHyperLinks – Удалить гиперссылки

Интерфейс...

Синтаксис Automation:

BOOL DeleteHyperLinks(VARIANT Objects);

Синтаксис COM:

HRESULT DeleteHyperLinks(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- объекты, содержащие гиперссылку.

GetHyperLinkObjects – Получить объекты, имеющие гиперссылку

Интерфейс...

Синтаксис Automation:

VARIANT GetHyperLinkObjects(ksHyperLinkTypeEnum Type,
IDrawingObject * LinkObject,
long Level,
BSTR Text);

Синтаксис COM:

HRESULT GetHyperLinkObjects(ksHyperLinkTypeEnum Type,
IDrawingObject * LinkObject,
long Level,
BSTR Text,
VARIANT * Result);

Возвращаемое значение:

VT_DISPATCH или массив VT_ARRAY | VT_DISPATCH объектов, содержащих ссылку

Входные параметры:

Type	- тип ссылки,
Text	- текст ссылки,
LinkObject	- объект, на который сделана ссылка,
Level	- уровень.

GetObjectById – Получить объект по его уникальному идентификатору

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObjectById(int64 Id);

Синтаксис COM:

HRESULT GetObjectById(int64 Id, IDrawingObject ** Result);

Возвращаемое значение:

- указатель на интерфейс объекта IDrawingObject.

Входные параметры:

Id - идентификатор объекта.

FindObject – Найти объект, ближайший к заданной точке, удовлетворяющий параметрам поиска

Интерфейс...

Синтаксис Automation:

LPDISPATCH FindObject(double X, double Y, double Limit, IFindObjectParameters * Param);

Синтаксис COM:

HRESULT FindObject(double X, double Y, double Limit, IFindObjectParameters * Param, IDrawingObject * * Result);

Возвращаемое значение:

- указатель на интерфейс объекта IDrawingObject.

Входные параметры:

X - первая координата точки,
Y - вторая координата точки,
Limit - предел поиска и найденное расстояние,
Param - параметры поиска объектов IFindObjectParameters.

Примечание:

Для поиска объектов в точке могут быть заданы расширенные параметры поиска.

Интерфейс расширенных параметров поиска IFindObjectParameters можно получить с помощью метода IKompasDocument1::GetInterface в 2D - документах при использовании константы ksObjectFindObjectParameters.

Если расширенные параметры поиска не требуются, необходимо передать NULL в функцию.

FindObjects – Найти объекты в заданной точке, удовлетворяющие параметрам поиска

Интерфейс...

Синтаксис Automation:

VARIANT FindObjects(double X, double Y, double Limit, IFindObjectParameters * Param);

Синтаксис COM :

HRESULT FindObjects(double X, double Y, double Limit, IFindObjectParameters * Param, VARIANT * Result);

Возвращаемое значение:

VARIANT - объект типа VARIANT на безопасный массив SAFEARRAY: VT_ARRAY|VT_DISPATCH.

Входные параметры:

X - первая координата точки,
Y - вторая координата точки,
Limit - предел поиска и найденное расстояние,
Param - параметры поиска объектов IFindObjectParameters.

Примечание:

Для поиска объектов в точке могут быть заданы расширенные параметры поиска.

Интерфейс расширенных параметров поиска IFindObjectParameters можно получить с помощью метода IKompasDocument1::GetInterface в 2D - документах при использовании константы ksObjectFindObjectParameters.

Если расширенные параметры поиска не требуются, необходимо передать NULL в функцию.

SelectObjects – Отобрать объекты рамкой

Интерфейс...

Синтаксис Automation:

VARIANT SelectObjects(ksRegionTypeEnum RegionType, double XMin, double YMin, double XMax, double YMax);

Синтаксис COM :

HRESULT SelectObjects(ksRegionTypeEnum RegionType, double XMin, double YMin, double XMax, double YMax, VARIANT * Result);

Возвращаемое значение:

VARIANT - объект типа VARIANT на безопасный массив SAFEARRAY: VT_ARRAY|VT_DISPATCH.

Входные параметры:

RegionType - тип региона,
XMin - минимальная граница поиска по x
YMin - минимальная граница поиска по y,
XMax - максимальная граница поиска по x,
YMax - максимальная граница поиска по y.

IsVariableNameValid – Проверить допустимость создания новой переменной с данным именем

Интерфейс...

Синтаксис Automation:

BOOL IsVariableNameValid(BSTR Name);

Синтаксис COM:

HRESULT IsVariableNameValid(BSTR Name, BOOL * Result);

Входные параметры:

Name - имя переменной.

Возвращаемое значение:

TRUE - переменную с данным именем создать можно,
FALSE - переменную с данным именем создать нельзя.

Метод позволяет проверить, можно ли создать переменную с данным именем (проверяются синтаксис имени и наличие в документе переменной с данным именем).

RebuildDocument – Перестроить документ

Интерфейс...

Синтаксис Automation:

BOOL RebuildDocument();

Синтаксис COM:

HRESULT RebuildDocument(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Метод позволяет перестроить документ.

UpdateVariables – Установить новые значения внешних переменных

Интерфейс...

Синтаксис Automation:

BOOL UpdateVariables()

Синтаксис COM:

HRESULT UpdateVariables(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Метод позволяет применить новые значения внешних переменных.

Интерфейс IKompasDocument3D1

Дополнительный интерфейс для 3D документа.

Иерархия:

IDispatch

IKompasDocument3D1

IKompasDocument3D1 – свойства

Document3DManager – Менеджер 3D документа

Тип данных: Указатель на интерфейс IDocument3DManager

Синтаксис Automation:

Document3DManager = Object.Document3DManager	Получить свойство (*)
Document3DManager = Object.GetDocument3DManager()	Получить свойство (**)

Синтаксис COM:

Object.get_Document3DManager(&Document3DManager)	Получить свойство

Примечание:

Свойство доступно только для чтения.

HideLayoutGeometry – Скрыть / показать компоновочную геометрию

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HideLayoutGeometry = Object.HideLayoutGeometry	Получить свойство (*)
Object.HideLayoutGeometry = HideLayoutGeometry	Установить свойство (*)

HideLayoutGeometry = Object.GetHideLayoutGeometry()	Получить свойство (**)
Object.SetHideLayoutGeometry(HideLayoutGeometry)	Установить свойство (**)

Синтаксис COM:

Object.get_HideLayoutGeometry(&HideLayoutGeometry)	Получить свойство
Object.put_HideLayoutGeometry(HideLayoutGeometry)	Установить свойство

Примечание:

В зависимости от флага HideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

EditObject – Редактируемый объект

Интерфейс...

Тип данных: Указатель на интерфейс IFeature7

Синтаксис Automation:

EditObject = Object.EditObject	Получить свойство (*)
Object.EditObject = EditObject	Установить свойство (*)
EditObject = Object.GetEditObject()	Получить свойство (**)
Object.SetEditObject(EditObject)	Установить свойство (**)

Синтаксис COM:

Object.get_EditObject(&EditObject))	Получить свойство
Object.put_EditObject(EditObject)	Установить свойство

LibProcess – Получить объект процесса

Интерфейс...

Тип данных: Указатель на интерфейс IProcess3D

Синтаксис Automation:

LibProcess = Object.LibProcess(ProcessType)	Получить свойство (*)
LibProcess = Object.GetLibProcess(ProcessType)	Получить свойство (**)

Синтаксис COM:

Object.get_LibProcess(ProcessType, &LibProcess) — Получить свойство

Примечание:

Свойство доступно только для чтения.

MateConstraints – Коллекция сопряжений

Интерфейс...

Тип данных: Указатель на интерфейс IMateConstraints3D

Синтаксис Automation:

MateConstraints = Object.MateConstraints — Получить свойство (*)
MateConstraints = Object.GetMateConstraints() — Получить свойство (**)

Синтаксис COM:

Object.get_MateConstraints(&MateConstraints) — Получить свойство

Примечание:

Свойство доступно только для чтения.

SpecRough – Неуказанная шероховатость 3D

Интерфейс...

Тип данных: Указатель на интерфейс ISpecRough3D

Синтаксис Automation:

SpecRough = Object.SpecRough — Получить свойство (*)
SpecRough = Object.GetSpecRough() — Получить свойство (**)

Синтаксис COM:

Object.get_SpecRough(&SpecRough) — Получить свойство

Примечание:

Свойство доступно только для чтения.

IKompasDocument3D1 – методы

ClearUndo – Очистить очередь Undo

Интерфейс...

Синтаксис Automation:

BOOL ClearUndo();

Синтаксис COM:

HRESULT ClearUndo(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ExcludeObjects – Исключить из расчета объект или объекты

Интерфейс...

Синтаксис Automation:

BOOL ExcludeObjects(VARIANT Objects, BOOL Excl);

Синтаксис COM:

HRESULT ExcludeObjects(VARIANT Objects, BOOL Excl, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- объект (VT_DISPATCH) или Список объектов VT_ARRAY | VT_DISPATCH,
Excl	- TRUE - исключить из расчета - FALSE - включить в расчет.

ExecuteProcessOfInsertComponentFromFile – Запустить процесс вставки компонента из файла или библиотеки моделей

Интерфейс...

Синтаксис Automation:

BOOL ExecuteProcessOfInsertComponentFromFile(BSTR FileName, ProcessTypeEnum ProcessType);

Синтаксис COM:

HRESULT ExecuteProcessOfInsertComponentFromFile(BSTR FileName, ProcessTypeEnum ProcessType, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE - в случае неудачи.

Входные параметры:

FileName	- имя файла,
ProcessType	- тип процесса,
prAddPartFromFile	- добавить компонент из файла,
prAddLocalPartFromFile	- добавить локальную деталь из файла,
prAddLayoutGeometryFromFile	- добавить компоновочную геометрию,
prAddBilletPartFromFile	- добавить деталь-заготовку.

Подсветка и выделение объектов

Интерфейс IChooseManager

Справка системы КОМПАС...

kompas.chm::/98_8_5_Vydelenie_obwektov.htm

Интерфейс менеджера выбора (подсветки) объектов.

Иерархия:

IDispatch

IKompasAPIObject

IChooseManager

Описание:

Интерфейс позволяет осуществлять выбор (подсветку) объектов. Используется при создании отчета по выбранным объектам.

Примечание:

Интерфейс является дополнительным для интерфейса таблицы отчетов IReport. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) и у интерфейса документа 3D с помощью свойства IKompasDocument3D::ChooseManager, а также у дополнительного интерфейса документа 2D с помощью свойства IKompasDocument2D1::ChooseManager.

IChooseManager – свойства

ChoosenObjects – Получить массив выделенных объектов в виде SAFEARRAY | VT_DISPATCH.

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

ChoosenObjects = Object.ChoosenObjects	Получить свойство (*)
ChoosenObjects = Object.GetChoosenObjects()	Получить свойство (**)

Синтаксис COM:

Object.get_ChoosenObjects(&ChoosenObjects)	Получить свойство

Примечание:

1. Свойство позволяет получать массив выделенных объектов. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

CurrentManagerType – Получить тип текущего менеджера

Интерфейс...

Тип данных: из перечисления ksChooseManagerTypeEnum

Синтаксис Automation:

CurrentManagerType = Object.CurrentManagerType	=	Получить свойство (*)
Object.CurrentManagerType = CurrentManagerType	=	Установить свойство (*)
CurrentManagerType = Object.GetCurrentManagerType()	=	Получить свойство (**)
Object.SetCurrentManagerType(CurrentManagerType)		Установить свойство (**)

Синтаксис COM:

Object.get_CurrentManagerType(&CurrentManagerType)	Получить свойство
Object.put_CurrentManagerType(CurrentManagerType)	Установить свойство

IChooseManager – методы

Choose – Выделить объект

Интерфейс...

Синтаксис Automation:

BOOL Choose(VARIANT Objects);

Синтаксис COM:

HRESULT Choose (VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects - объект или массив объектов, которые нужно выделить (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет выделить объект.

GetManagerIndex – Получить индекс менеджера по указателю на выбранный объект

Интерфейс...

Синтаксис Automation:

ksChooseManagerTypeEnum GetManagerIndex (LPDISPATCH Object);

Синтаксис COM:

HRESULT GetManagerIndex(IKompasAPIObject * Object, ksChooseManagerTypeEnum * Result);

Входные параметры:

Object - указатель на выбранный объект.

Возвращаемое значение:

тип текущего менеджера из перечисления ksChooseManagerTypeEnum

IsChoosen – Выделен ли объект

Интерфейс...

Синтаксис Automation:

BOOL IsChoosen(LPDISPATCH Object);

Синтаксис COM:

HRESULT IsChoosen(IKompasAPIObject * Object,
BOOL * Result);

Входные параметры:

Objects - объект.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет определить, выделен ли объект.

Unchoose – Снять выделение объекта

Интерфейс...

Синтаксис Automation:

BOOL Unchoose(VARIANT Objects);

Синтаксис COM:

HRESULT Unchoose(VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects - объект или массив объектов, с которых нужно снять выделение (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет снять выделение объекта.

UnchooseAll – Снять выделение всех объектов

Интерфейс...

Синтаксис Automation:

BOOL UnchooseAll();

Синтаксис COM:

HRESULT UnchooseAll(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет снять выделение со всех объектов.

Интерфейс ISelectionManager

Справка системы КОМПАС...

kompas.chm::/98_8_5_Vydelenie_obwektov.htm

Интерфейс менеджера выделенных объектов.

Иерархия:

IDispatch

IKompasAPIObject

ISelectionManager

Описание:

Интерфейс позволяет осуществлять выделение объектов.

Примечание:

Интерфейс можно получить у интерфейса документа 3D с помощью свойства IKompasDocument3D::SelectionManager и у дополнительного интерфейса документа 2D с помощью свойства IKompasDocument2D1::SelectionManager.

ISelectionManager – свойства

SelectedObjects – Получить массив выделенных объектов в виде SAFEARRAY | VT_DISPATCH

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Интерфейс...

Синтаксис Automation:

SelectedObjects = Object.SelectedObjects	Получить свойство (*)
SelectedObjects = Object.GetSelectedObjects()	Получить свойство (**)

Синтаксис COM:

Object.get_SelectedObjects(&SelectedObjects)	Получить свойство

Примечание:

1. Свойство позволяет получать массив выделенных объектов. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

ISelectionManager – методы

IsSelected – Выделен ли объект

Интерфейс...

Синтаксис Automation:

BOOL IsSelected(LPDISPATCH Object);

Синтаксис COM:

HRESULT IsSelected(IKompasAPIObject * Object,
BOOL * Result);

Входные параметры:

Objects - объект.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет определить, выделен ли объект.

Select – Выделить объект

Интерфейс...

Синтаксис Automation:

BOOL Select(VARIANT Objects);

Синтаксис COM:

HRESULT Select(VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects - объект или массив объектов, которые нужно выделить (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет выделить объект.

Unselect – Снять выделение объекта

Интерфейс...

Синтаксис Automation:

BOOL Unselect(VARIANT Objects);

Синтаксис COM:

HRESULT Unselect(VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов, с которых нужно снять выделение (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет снять выделение объекта.

UnselectAll – Снять выделение всех объектов

Интерфейс...

Синтаксис Automation:

BOOL UnselectAll();

Синтаксис COM:

HRESULT UnselectAll(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет снять выделение со всех объектов.

Окна документа

Интерфейс IDocumentFrame

Интерфейс событий...

Справка системы КОМПАС...

kompas.chm::/63_1_2_1_okno_sistemy.htm

Интерфейс окна документа.

IKompasAPIObject

IDocumentFrame

IFrameTreesManager

Примечание:

1. Данный интерфейс может быть получен от интерфейса окон документа IDocumentFrames с помощью свойства IDocumentFrames::Item.
2. У данного интерфейса посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif) можно получить дополнительный интерфейс IFrameTreesManager.

IDocumentFrame – свойства

Active – Активность окна

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Active = iObject.Active	Получить свойство (*)
iObject.Active = Active	Установить свойство (*)
Active = iObject.GetActive()	Получить свойство (**)
iObject.SetActive (Active)	Установить свойство (**)

Синтаксис COM:

iObject->get_Active (&Active)	Получить свойство,
iObject->put_Active (Active)	Установить свойство

Примечание:

С помощью данного свойства можно получить и изменить активность данного окна.

Caption – Заголовок окна

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Caption = iObject.Caption	Получить свойство (*)
Caption = iObject.GetCaption()	Получить свойство (**)

Синтаксис COM:

iObject->get_Caption (&Caption)	Получить свойство,

Примечание:

1. С помощью данного свойства можно получить заголовок данного окна.
2. Свойство доступно только для чтения.

CurrentCursorStep – Текущий шаг дискретного перемещения курсора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CurrentCursorStep = Object.CurrentCursorStep( )	Получить свойство (*)
CurrentCursorStep = Object.GetCurrentCursorStep()	Получить свойство (**)

Синтаксис COM:

Object.get_CurrentCursorStep(&CurrentCursorStep)	Получить свойство,

Примечание

При включенном режиме округления IDocumentFrame::RoundModeOn значения параметров округляются до ближайшего значения, кратного текущему шагу курсора.

Regime – Режим отображения окна

Интерфейс...

Тип данных: режим отображения окна из перечисления FrameRegimeEnum

Синтаксис Automation:

Regime = iObject.Regime	Получить свойство (*)
iObject.Regime = Regime	Установить свойство (*)
Regime = iObject.GetRegime()	Получить свойство (**)
iObject.SetRegime (Regime)	Установить свойство (**)

Синтаксис COM:

```
iObject->get_Regime (&Regime)        Получить свойство,
iObject->put_Regime (Regime)         Установить свойство
```

Примечание:

С помощью данного свойства можно получить и изменить режим отображения данного окна.

RoundModeOn – Состояние режима округления линейных величин до значений, кратных шагу курсора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
RoundModeOn = Object.RoundModeOn( )      Получить свойство (* )
RoundModeOn = Object.GetRoundModeOn()    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_RoundModeOn(                  Получить свойство,
&RoundModeOn )
```

Примечание

При включенном режиме значения параметров округляются до ближайшего значения, кратного текущему шагу курсора IDocumentFrame::CurrentCursorStep.

IDocumentFrame – методы

ConvertCoordinates – Преобразовать оконные координаты в координаты СК согласно ConvertCoordTypeEnum

Интерфейс...

Синтаксис Automation:

```
BOOL ConvertCoordinates (ConvertCoordTypeEnum type,
long lx,
long ly,
double* x,
double* y,
double* z);
```

Синтаксис COM:

HRESULT ConvertCoordinates ([in]ConvertCoordTypeEnum type,
[in]long lx,
[in]long ly,
[out]double* x,
[out]double* y,
[out]double* z,
[out, retval]
VARIANT_BOOL* Result);

Входные параметры:

lx, ly	- координаты в экранных пикселах,
type	- тип СК в которую нужно преобразовать оконные координаты.

Выходные параметры:

x, y, z	- координаты в мм в СК согласно ConvertCoordTypeEnum.

Возвращаемое значение:

TRUE	- преобразование прошло успешно,
FALSE	- в случае неудачи.

Примечание:

1. Получив координаты мыши в оконных координатах при обработке событий из ksDocumentFrameNotify:

frMouseDown	3	Нажатие кнопки мыши.
frMouseUp	4	Отпускание кнопки мыши.
frMouseDblClick	5	Двойной щелчок мыши.
frMouseMove	12	Перемещение мыши.

их можно преобразовать в системы координат, указанные в ConvertCoordTypeEnum:

- для графических документов можно получить координаты в СК листа и текущего вида,
- для СП и текстового документа в СК первого листа,
- для модели в СК документа и в СК текущей плоской грани. Грань должна быть выделена в дереве.

2. Если курсор находится в нерабочей области окна активного документа, то конвертация координат не производится, возвращается FALSE. Нерабочей областью окна является, например, панель команд или панель свойств в этом же активном документе.

ExecuteKompasCommand – Выполнить команду системы КОМПАС

Интерфейс...

Синтаксис Automation:

BOOL ExecuteKompasCommand (long commandID, BOOL post);

Синтаксис COM:

HRESULT ExecuteKompasCommand (long commandID, BOOL post, BOOL * retval);

Входные параметры:

commandID	- константа из перечисления ProcessTypeEnum или ksKompasCommandEnum,
post	- true - запуск команды через PostMessage, - false - через SendMessage.

Возвращаемое значение:

TRUE	- в случае удачи,
FALSE	- в случае ошибки.

Примечание:

1. Проверить доступность команды можно с помощью функции IDocumentFrame::IsKompasCommandEnable.
2. Проверить, нажата ли в данный момент кнопка команды, можно с помощью IDocumentFrame::IsKompasCommandCheck.

GetHWND – Получить дескриптор окна

Интерфейс...

Синтаксис Automation:

long GetHWND();

Синтаксис COM:

HRESULT GetHWND ([out, retval] OLE_HANDLE* hwnd);

Возвращаемое значение:

Дескриптор окна

GetPickRay – Преобразовать оконные координаты в луч взгляда

Интерфейс...

Синтаксис Automation:

BOOL GetPickRay (long lx, long ly,
double* x, double* y, double* z,
double* zx, double* zy, double* zz);

Синтаксис COM:

```
HRESULT GetPickRay( long lx, long ly,
                    double* x, double* y, double* z,
                    double* zx, double* zy, double* zz,
                     VARIANT_BOOL* Result);
```

Входные параметры:

lx, ly - координаты в экранных пикселах.

Выходные параметры:

x, y, z - координаты точки на ближайший к лучу взгляда объект, в координатах компонента,
zx, zy, zz - направления вектора взгляда.

Возвращаемое значение:

TRUE - преобразование прошло успешно,
FALSE - в случае неудачи.

Примечание:

Луч смотрит перпендикулярно экрану. Луч строится с учетом перспективной проекции, если она включена.

GetZoomScale – Получить масштаб и центр окна документа

Интерфейс...

Синтаксис Automation:

void GetZoomScale (double* x, double* y, double* scale);

Синтаксис COM:

HRESULT GetZoomScale ([out]double* x, [out]double* y, [out]double* scale);

Выходные параметры:

x, y - координаты (в СК вида) точки центра изменения масштаба,
scale - коэффициент изменения масштаба изображения.

Примечание:

1. Метод позволяет получить текущий масштаб отображения документа в окне с центром в точке x, y (точка x, y в центре окна).
2. Метод работает только для графических документов.

IsKompasCommandCheck – Проверить, нажата ли кнопка команды

Интерфейс...

Синтаксис Automation:

long IsKompasCommandCheck (long commandID);

Синтаксис COM:

HRESULT IsKompasCommandCheck (long commandID, long * retVal);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

1 - кнопка нажата,
0 - кнопка отжата.

IsKompasCommandEnable – Проверить доступность выполнения команды

Интерфейс...

Синтаксис Automation:

BOOL IsKompasCommandEnable (long commandID);

Синтаксис COM:

HRESULT IsKompasCommandEnable (long commandID, BOOL * retVal);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

TRUE - команда доступна,
FALSE - команда недоступна.

RefreshWindow – Обновить изображение окна

Интерфейс...

Синтаксис Automation:

void RefreshWindow();

Синтаксис COM:

HRESULT RefreshWindow();

SetGabaritModifying – Сообщить окну документа, что библиотека меняет габарит документа

Интерфейс...

Синтаксис Automation:

void SetGabaritModifying();

Синтаксис COM:

SetGabaritModifying();

Примечание:

Использование метода может понадобиться для собственного рисования в окне. Если габарит изменен, то это приведет к генерации события frAddGabarit (определение габаритов документа) из ksDocumentFrameNotify. В обработчике события frAddGabarit библиотека может указать свой габаритный прямоугольник, если документ является чертежом или фрагментом или свой габаритный куб, если документ является моделью. Значение габарита библиотеки сложится с габаритом документа.

Zoom – Увеличить масштаб окна рамкой

Интерфейс...

Синтаксис Automation:

void Zoom (double x1, double y1, double x2, double y2);

Синтаксис COM:

HRESULT Zoom ([in]double x1, [in]double y1, [in]double x2, [in]double y2);

Входные параметры:

x1, y1 - координаты (в СК вида) первой точки диагонали рамки,
x2, y2 - координаты (в СК вида) второй точки диагонали рамки.

Примечание:

1. Метод позволяет растянуть часть документа, заданную рамкой, во все окно.
2. Метод работает только для графических документов.

ZoomPrevNextOrAll – Отобразить предыдущий/следующий масштаб или показать весь документ

Интерфейс...

Синтаксис Automation:

void ZoomPrevNextOrAll (long Type);

Синтаксис COM:

HRESULT ZoomPrevNextOrAll ([in]ZoomTypeEnum type);

Входные параметры:

type - тип изменения масштаба из перечисления ZoomTypeEnum.

Примечание:

1. Установка следующего масштаба возможно только после отката к предыдущему масштабу.
2. Метод работает в полном объеме только для графических документов. Для документов-моделей работает только режим ksZoomAll (Показать весь документ).
3. В текстовом документе и в спецификации метод не работает.

ZoomScale – Изменить масштаб изображения окна

Интерфейс...

Синтаксис Automation:

void ZoomScale (double x, double y, double scale);

Синтаксис COM:

HRESULT ZoomScale ([in]double x, [in]double y, [in]double scale);

Входные параметры:

x, y - координаты (в СК вида) точки центра изменения масштаба,
scale - коэффициент изменения масштаба изображения.

Примечание:

1. Метод позволяет установить новый масштаб отображения документа в окне с центром в точке x,y.
2. Метод работает только для графических документов.

Интерфейс IDocumentFrames

Справка системы КОМПАС...

KOMPAS.chm::/63_1_2_1_okno_sistemy.htm

Коллекция окон документа.

Иерархия:

IKompasAPIObject

IKompasCollection

IDocumentFrames

Примечание:

Данный интерфейс может быть получен от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::DocumentFrames.

IDocumentFrames – свойства

Item – Окно, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс окна IDocumentFrame

Синтаксис Automation:

frame = iObject.Item (Index)	Получить свойство (*)
frame = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (Index, &Frame)	Получить свойство

Входные параметры:

Index	- индекс в коллекции, тип VARIANT.

Примечание:

Свойство доступно только для чтения.

Интерфейс IFrameTreesManager

Интерфейс менеджера вкладок окна документа.

Описание:

Дополнительный интерфейс для IDocumentFrame.

Данный интерфейс можно получить у интерфейса IDocumentFrame посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

Примечание:

1. Позволяет создавать вкладку в окне дерева документа.
2. Добавить вкладку можно в окно чертежа, 3D модели и сборки.
3. Для создания закладки используется метод AddTab.

IFrameTreesManager – свойства

ActiveTab – Активная закладка

Интерфейс...

Тип данных: IDispatch

Синтаксис Automation:

ActiveTab = Object.ActiveTab	Получить свойство (*)
Object.ActiveTab = ActiveTab	Установить свойство (*)

ActiveTab = Object.GetActiveTab()	Получить свойство (**)
Object.SetActiveTab(ActiveTab)	Установить свойство (**)

Синтаксис COM:

Object.get_ActiveTab(&ActiveTab)	Получить свойство
Object.put_ActiveTab(ActiveTab)	Установить свойство

Примечание:

1. Позволяет получить и установить активность вкладки окна дерева документа.
2. Для активизации закладки дерева документа системы КОМПАС нужно передать NULL.
3. Для активизации закладки ActiveX библиотеки нужно передать указатель на интерфейс созданного функцией AddTab ActiveX.

TabVisible – Видимость закладки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TabVisible = Object.TabVisible(ocx)	Получить свойство (*)
Object.TabVisible(ocx) = TabVisible	Установить свойство (*)
TabVisible = Object.GetTabVisible(ocx)	Получить свойство (**)
Object.SetTabVisible(ocx, TabVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_TabVisible(ocx, &TabVisible)	Получить свойство
Object.put_TabVisible(ocx, TabVisible)	Установить свойство

Входные параметры:

ocx	- указатель на интерфейс ActiveX контрола, добавленного на закладку.

TabsVisible – Видимость вкладок документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TabsVisible = Object.TabsVisible	Получить свойство (*)
Object.TabsVisible = TabsVisible	Установить свойство (*)
TabsVisible = Object.GetTabsVisible()	Получить свойство (**)
Object.SetTabsVisible(TabsVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_TabsVisible(&TabsVisible)	Получить свойство
Object.put_TabsVisible(TabsVisible)	Установить свойство

Примечание:

Позволяет получить и установить видимость окна дерева документа.

TreeCaption – Заголовок окна Дерева документа, устанавливаемый при активизации закладки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

TreeCaption = Object.TreeCaption(Ocx)	Получить свойство (*)
Object.TreeCaption(Ocx) = TreeCaption	Установить свойство (*)
TreeCaption = Object.GetTreeCaption(Ocx)	Получить свойство (**)
Object.SetTreeCaption(Ocx, TreeCaption)	Установить свойство (**)

Синтаксис COM:

Object.get_TreeCaption(Ocx, &TreeCaption)	Получить свойство
Object.put_TreeCaption(Ocx, TreeCaption)	Установить свойство

Входные параметры:

Ocx - Указатель на интерфейс ActiveX-окна, возвращаемого методом IFrameTreesManager::AddTab

IFrameTreesManager – методы

AddTab – Добавить вкладку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddTab(BSTR tabCaption, BSTR ocxProgID);

Синтаксис COM:

HRESULT AddTab(BSTR tabCaption, BSTR ocxProgID, IDispatch ** PVal);

Входные параметры:

tabCaption	- заголовок для новой закладки,
ocxProgID	- ProgID - ActiveX элемента (ActiveX дерева); Например, «VCTree.VCTreeCtrl.1» - для создания дерева, из примеров SDK. см SDK\lib\VCTree.ocx, проект SDK\OCX\VCTree. Пример использования SDK\C++\Visualc\TestVCTree.

Возвращаемое значение:

- Указатель на интерфейс созданного ActiveX элемента.

AddTabEx – Добавить закладку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddTabEx(BSTR TabCaption, BSTR TreeCaption, BSTR OcxClassID, BOOL Active, PropertyManagerLayout Layout);

Синтаксис COM:

HRESULT AddTabEx(BSTR TabCaption, BSTR TreeCaption, BSTR OcxClassID, BOOL Active, PropertyManagerLayout Layout, IDispatch* * Result);

Возвращаемое значение:

- Указатель на интерфейс созданного ActiveX элемента.

Входные параметры:

tabCaption	- имя закладки,
TreeCaption	- заголовок дерева ,
ocxProgID	- ProgID - ActiveX элемента (ActiveX дерева); Например, «VCTree.VCTreeCtrl.1» - для создания дерева, из примеров SDK. см SDK\lib\VCTree.ocx, проект SDK\OCX\VCTree. Пример использования SDK\C++\Visualc\TestVCTree,
Active	- активизировать закладку при создании,
Layout	- умолчательное расположение.

Примечание:

При повторном создании закладки она будет создана в том же положении, где ее разместил пользователь.

RemoveTab – Удалить вкладку

Интерфейс...

Синтаксис Automation:

BOOL RemoveTab(LPDISPATCH ocx);

Синтаксис COM:

HRESULT RemoveTab(LPDISPATCH ocx, BOOL * Result);

Входные параметры:

o c x	- указатель на интерфейс ActiveX окна.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

OpenGL

Интерфейс IExternalGDIObject

Интерфейс внешнего GDI- объекта.

Иерархия:

IKompasAPIObject

IExternalGDIObject

Примечание:

1. Интерфейс позволяет реализовать отрисовку внешних GDI- объектов (для V11 только текстов).
2. Управление интерфейсом (создание и удаление) осуществляется через менеджер IExternalTessellationManager.

IExternalGDIObject – свойства

AlwaysDrawInScreenPlane – Способ преобразования координат внешней триангуляции

Интерфейс...

Тип данных: из перечисления ksDrawInScreenPlaneEnum

Синтаксис Automation:

AlwaysDrawInScreenPlane = Object.AlwaysDrawInScreenPlane	Получить свойство (*)
Object.AlwaysDrawInScreenPlane = AlwaysDrawInScreenPlane	Установить свойство (*)
AlwaysDrawInScreenPlane = Object.GetAlwaysDrawInScreenPlane()	Получить свойство (**)
Object.SetAlwaysDrawInScreenPlane(AlwaysDrawInScreenPlane)	Установить свойство (**)

Синтаксис COM:

Object.get_AlwaysDrawInScreenPlane(&AlwaysDrawInScreenPlane)	Получить свойство

Object.put_AlwaysDrawInScreenPlane(AlwaysDrawInScreenPlane)	Установить свойство

NonScalableX – Не масштабировать по X

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonScalableX = Object.NonScalableX	Получить свойство (*)
Object.NonScalableX = NonScalableX	Установить свойство (*)
NonScalableX = Object.GetNonScalableX()	Получить свойство (**)
Object.SetNonScalableX(NonScalableX)	Установить свойство (**)

Синтаксис COM:

Object.get_NonScalableX(&NonScalableX)	Получить свойство
Object.put_NonScalableX(NonScalableX)	Установить свойство

NonScalableY – Не масштабировать по Y

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonScalableY = Object.NonScalableY	Получить свойство (*)
Object.NonScalableY = NonScalableY	Установить свойство (*)
NonScalableY = Object.GetNonScalableY()	Получить свойство (**)
Object.SetNonScalableY(NonScalableY)	Установить свойство (**)

Синтаксис COM:

Object.get_NonScalableY(&NonScalableY)	Получить свойство
Object.put_NonScalableY(NonScalableY)	Установить свойство

NonScalableZ – Не масштабировать по Z

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonScalableZ = Object.NonScalableZ	Получить свойство (*)
Object.NonScalableZ = NonScalableZ	Установить свойство (*)
NonScalableZ = Object.GetNonScalableZ()	Получить свойство (**)
Object.SetNonScalableZ(NonScalableZ)	Установить свойство (**)

Синтаксис COM:

```
Object.get_NonScalableZ( &NonScalableZ )      Получить свойство
Object.put_NonScalableZ(  NonScalableZ )      Установить свойство
```

ObjectID – Идентификатор объекта

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
ObjectID = Object.ObjectID                    Получить свойство (* )
ObjectID = Object.GetObjectID()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ObjectID( &ObjectID )              Получить свойство,
```

Примечание:

1. Позволяет получить идентификатор объекта. По идентификатору объекта осуществляется установка видимости и удаление объектов.
2. Свойство доступно только для чтения.

ScalableText – Признак масштабирования текста

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ScalableText = Object.ScalableText            Получить свойство (* )
Object.ScalableText = ScalableText            Установить свойство (* )
ScalableText = Object.GetScalableText()       Получить свойство (**)
Object.SetScalableText( ScalableText )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ScalableText( &ScalableText )      Получить свойство
Object.put_ScalableText(  ScalableText )      Установить свойство
```

Visible – Показать\скрыть объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Object.Visible(frame) = Visible — Установить свойство (*)
Object.SetVisible(frame, Visible) — Установить свойство (**)

Синтаксис COM:

Object.put_Visible(frame, Visible) — Установить свойство,

Входные параметры:

frame - указатель на интерфейс окна документа IDocumentFrame.

Примечание:

1. Позволяет изменить видимость объекта в заданном окне.
2. Если frame == NULL, то объект будет отображаться во всех окнах документа.
3. Свойство доступно только для изменения.

IExternalGDIObject – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete( );

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Перед удалением объекта нужно выключить видимость объекта в окнах документа.
2. Для удаления группы объектов используется функция IExternalTessellationManager::DeleteObjects.
3. Для удаления всех внешних объектов используется метод IExternalTessellationManager::Clear.

SetBkColors – Установить цвета фона текстов с Alpha каналом

Интерфейс...

Синтаксис Automation:

BOOL SetBkColors(VARIANT BkColors);

Синтаксис COM:

HRESULT SetBkColors(VARIANT BkColors, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае ошибки.

Входные параметры:

BkColors	- массив SafeArray VT_ARRAY | VT_I4.

SetPlace – Установить матрицу для отрисовки объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPlace(VARIANT Place);

Синтаксис COM:

HRESULT SetPlace(VARIANT Place, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Входные параметры:

Place	- массив SAFEARRAY double (VT_ARRAY | VT_R8).

SetTextColors – Установить цвета текстов

Интерфейс...

Синтаксис Automation:

BOOL SetTextColors(VARIANT textColors);

Синтаксис COM:

HRESULT SetTextColors(VARIANT textColors, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

textColors - массив цветов - SafeArray целых чисел VT_ARRAY | VT_I4.

SetTextOrientation – Установить наклон символов текстов

Интерфейс...

Синтаксис Automation:

BOOL SetTextOrientation(VARIANT textOrients);

Синтаксис COM:

HRESULT SetTextOrientation(VARIANT textOrients, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

textOrients - массив наклонов.

SetTexts – Установить тексты и их параметры

Интерфейс...

Синтаксис Automation:

BOOL SetTexts(VARIANT Texts,
VARIANT Points,
VARIANT Colors,
VARIANT Fonts,
BOOL Is3DPoints);

Синтаксис COM:

HRESULT SetTexts(VARIANT Texts,
VARIANT Points,
VARIANT Colors,
VARIANT Fonts,
BOOL Is3DPoints,
BOOL * Result);

Входные параметры:

Points	- массив SafeArray VT_ARRAY | VT_R8 координат точек отрисовки текстов. Координаты в массиве могут лежать в последовательности : - для Is3DPoints == TRUE x0, y0, z0, x1, y1, z1, ... xi, yi, zi - для Is3DPoints == FALSE x0, y0, x1, y1, ... xi, yi,
Texts	- массив SafeArray VT_ARRAY | VT_BSTR текстов,
Colors	- массив цветов текстов. Массив может быть не задан - тип VARIANT-а VT_EMPTY, в этом случае используется черный цвет. Может быть задан один цвет - тип VARIANT-а VT_I4. Может быть задан цвет для каждого текста тип VARIANT-а VT_ARRAY | VT_I4,
Fonts	- массив шрифтов текстов. Массив может быть не задан - тип VARIANT-а VT_EMPTY. В этом случае используется шрифт, установленный в модели. Может быть задан один шрифт - тип VARIANT-а VT_DISPATCH. Может быть задан шрифт для каждого текста тип VARIANT-а VT_ARRAY | VT_DISPATCH. Передаваемый интерфейс - интерфейс IFontDispPtr, который можно получить от объекта MFC- класса CFontHolder,
Is3DPoints	- признак использования 3D-координат отрисовки текстов.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

SetTextsAlign – Установить выравнивание текста

Интерфейс...

Синтаксис Automation:

BOOL SetTextsAlign(VARIANT TextsAlign);

Синтаксис COM:

HRESULT SetTextsAlign(VARIANT TextsAlign, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае ошибки.

Входные параметры:

TextsAlign	- массив SafeArray VT_ARRAY | VT_I4 констант из перечисления ksTextAlignEnum.

Интерфейс IExternalTessellationManager

Менеджер объектов с внешней триангуляцией.

IExternalTessellationManager - **интерфейс Automation**

Описание:

Дополнительный интерфейс для IKompasDocument3D.

Данный интерфейс можно получить у интерфейса IKompasDocument3D посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

Примечание:

Позволяет создавать объекты с внешней триангуляцией, отрисовываемые треугольниками и ребрами.

IExternalTessellationManager – свойства

DisableModelRotation – Запретить поворот модели и изменение ориентации вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableModelRotation = Object.DisableModelRotation	=	Получить свойство (*)
Object.DisableModelRotation = DisableModelRotation	=	Установить свойство (*)
DisableModelRotation = Object.GetDisableModelRotation()	=	Получить свойство (**)
Object.SetDisableModelRotation(DisableModelRotation)		Установить свойство (**)

Синтаксис COM:

Object.get_DisableModelRotation(&DisableModelRotation)	Получить свойство
Object.put_DisableModelRotation(DisableModelRotation)	Установить свойство

GDIObject – Получить внешний GDI объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

GDIObject = Object.GDIObject(Id) Получить свойство (*)
GDIObject = Object.GetGDIObject(Id) Получить свойство (**)

Синтаксис COM:

Object.get_GDIObject(Ids, &GDIObject) Получить свойство

Входные параметры:

Id - идентификатор объекта.

ObjectsVisible – Показать\скрыть объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Object.ObjectsVisible(Ids, frame) = Visible Установить свойство (*)
Object.SetObjectsVisible(Ids, frame, Visible) Установить свойство (**)

Синтаксис COM:

Object.put_ObjectsVisible(Ids, frame, Visible) Установить свойство

Входные параметры:

Ids - Идентификаторы объектов тип VARIANT. Можно передать один идентификатор тип VARIANT-а VT_I4 или массив SafeArray идентификаторов тип VARIANT-а VT_ARRAY | VT_I4
frame - указатель на интерфейс окна документа IDocumentFrame.

Примечание:

1. Позволяет изменить видимость объектов в заданном окне. Если frame == NULL, то объект будет отображаться во всех окнах документа.
2. Свойство доступно только для изменения.

TessellationObject – Получить объект по идентификатору

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TessellationObject = Object.TessellationObject(Id)		Получить свойство (*)
TessellationObject = Object.GetTessellationObject(Id)		Получить свойство (**)

Синтаксис COM:

Object.get_TessellationObject(Ids, &TessellationObject)	Получить свойство

Входные параметры:

Id	- идентификатор объекта.

IExternalTessellationManager – методы

Add – Добавить объект с внешней триангуляцией

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IExternalTessellationObject ** PVal);

Возвращаемое значение:

- Указатель на интерфейс объекта с внешней триангуляцией IExternalTessellationObject.

AddGDIObject – Добавить внешний GDI– объект

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddGDIObject();

Синтаксис COM:

HRESULT AddGDIObject(IExternalGDIObject ** PVal);

Возвращаемое значение:

- Указатель на интерфейс внешнего GDI- объекта IExternalGDIObject.

Clear – Удалить все объекты

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * res);

Возвращаемое значение:

TRUE - в случае успешного завершения.

CreateTextureImage – Создать изображение текстуры

Интерфейс...

Синтаксис Automation:

```
long CreateTextureImage ( long   Width, long Heigh,
                          BOOL       RGBA,
                          BOOL       WrapMode,
                          BOOL       FiltMode,
                          VARIANT      ImageData );
```

Синтаксис COM:

```
HRESULT CreateTextureImage ( long   Width, long Heigh,
                             BOOL       RGBA,
                             BOOL       WrapMode,
                             BOOL       FiltMode,
                             VARIANT      ImageData,
                             long *  Result );
```

Входные параметры:

Width, Heigh	- Размер изображения,
RGBA	- TRUE - RGBA, FALSE - RGB,
WrapMode	- Режим повторения TRUE - GL_CLAMP FALSE - GL_REPEAT,
FiltMode	- Режим повторения TRUE - GL_LINEAR FALSE - GL_NEAREST,
ImageData	Массив байтов изображения текстуры VT_ARRAY|VT_UI1.

Возвращаемое значение:

- Идентификатор текстуры.

Возвращаемое значение:

TRUE - в случае удачи,
FALSE - в случае ошибки.

Примечание:

Созданная текстура используется в методе IExternalTessellationObj::SelectTextureImage.

DisableModelDrawing – Запретить отрисовку модели для указанных элементов

Интерфейс...

Синтаксис Automation

BOOL DisableModelDrawing(long ForElements);

Синтаксис COM:

HRESULT DisableModelDrawing(long ForElements, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

ForElements - типы объектов ksModelDrawingElementsEnum.

IsModelDrawingEnabled – Проверить, разрешена ли отрисовка для указанных элементов

Интерфейс...

Синтаксис Automation:

BOOL IsModelDrawingEnabled(long ForElements);

Синтаксис COM:

HRESULT IsModelDrawingEnabled(long ForElements, BOOL * Result);

Возвращаемое значение:

TRUE - если отрисовка разрешена.

Входные параметры:

ForElements - типы объектов ksModelDrawingElementsEnum.

EnableModelDrawing – Отменить последний запрет на отрисовку элементов модели

Интерфейс...

Синтаксис Automation:

BOOL EnableModelDrawing();

Синтаксис COM:

HRESULT EnableModelDrawing(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DeleteObjects – Удалить объекты

Интерфейс...

Синтаксис Automation:

BOOL DeleteObjects();

Синтаксис COM:

HRESULT DeleteObjects(VARIANT Id, BOOL * res);

Входные параметры:

Ids	- Идентификаторы объектов тип VARIANT. Можно передать один идентификатор тип VARIANT-а VT_I4 или массив SafeArray идентификаторов тип VARIANT-а VT_ARRAY | VT_I4.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

DeleteTextureImage – Удалить изображение текстуры

Интерфейс...

Синтаксис Automation:

BOOL DeleteTextureImage(long TexImgId);

Синтаксис COM:

HRESULT DeleteTextureImage(long TexImgId, BOOL * Result);

Входные параметры:

TexImgId	- идентификатор текстуры.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

PickObjects – Собрать объекты по лучу

Интерфейс...

Синтаксис Automation:

BOOL PickObjects(LPDISPATCH Frame, VARIANT PickRay, BOOL Visible, VARIANT * PickedObjs, VARIANT * PickedPars);

Синтаксис COM:

HRESULT PickObjects(IDocumentFrame * Frame, VARIANT PickRay, BOOL Visible, VARIANT * PickedObjs, VARIANT * PickedPars, BOOL * Result ;

Входные параметры:

Frame - Указатель на интерфейс окна документа IDocumentFrame,
PickRay - Параметры луча (VT_ARRAY | VT_R8), массив из шести элементов (Точка начала луча и единичный вектор задающий направление луча),
Visible - Признак учета видимых объектов:
TRUE - пересекать только с видимыми объектами,
FALSE - со всеми объектами.

Выходные параметры:

PickedObjs - Идентификаторы найденных внешних объектов триангуляции (VT_ARRAY | VT_UI4),
PickedPars - Параметры луча в точке пересечения с объектами (VT_ARRAY | VT_R4).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IExternalTessellationObject

Интерфейс объекта с внешней триангуляцией.

IExternalTessellationObject - **интерфейс Automation**

Примечание:

Интерфейс позволяет реализовать отрисовку внешних объектов с триангуляцией.

IExternalTessellationObject – свойства

AlwaysDrawInScreenPlane – Способ преобразования координат внешней триангуляции

Интерфейс...

Тип данных: из перечисления ksDrawInScreenPlaneEnum

Синтаксис Automation:

AlwaysDrawInScreenPlane = Object.AlwaysDrawInScreenPlane	Получить свойство (*)
Object.AlwaysDrawInScreenPlane = AlwaysDrawInScreenPlane	Установить свойство (*)
AlwaysDrawInScreenPlane = Object.GetAlwaysDrawInScreenPlane()	Получить свойство (**)
Object.SetAlwaysDrawInScreenPlane(AlwaysDrawInScreenPlane)	Установить свойство (**)

Синтаксис COM:

Object.get_AlwaysDrawInScreenPlane(&AlwaysDrawInScreenPlane)	Получить свойство
Object.put_AlwaysDrawInScreenPlane(AlwaysDrawInScreenPlane)	Установить свойство

DisableDepthTest – Запрет проверки глубины

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableDepthTest = Object.DisableDepthTest	Получить свойство (*)
Object.DisableDepthTest = DisableDepthTest	Установить свойство (*)
DisableDepthTest = Object.GetDisableDepthTest()	Получить свойство (**)
Object.SetDisableDepthTest(DisableDepthTest)	Установить свойство (**)

Синтаксис COM:

Object.get_DisableDepthTest(&DisableDepthTest)	Получить свойство
Object.put_DisableDepthTest(DisableDepthTest)	Установить свойство

Свойство позволяет устанавливать и получать признак запрета глубины.

NonGeometry – Не геометрический

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonGeometry = Object.NonGeometry	Получить свойство (*)
Object.NonGeometry = NonGeometry	Установить свойство (*)
NonGeometry = Object.GetNonGeometry()	Получить свойство (**)
Object.SetNonGeometry(NonGeometry)	Установить свойство (**)

Синтаксис COM:

Object.get_AlwaysDrawInScreenPlane(&AlwaysDrawInScreenPlane)	Получить свойство
Object.put_AlwaysDrawInScreenPlane(AlwaysDrawInScreenPlane)	Установить свойство

NonPickable – Не выбираемый

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonPickable = Object.NonPickable	Получить свойство (*)
Object.NonPickable = NonPickable	Установить свойство (*)
NonPickable = Object.GetNonPickable()	Получить свойство (**)
Object.SetNonPickable(NonPickable)	Установить свойство (**)

Синтаксис COM:

Object.get_NonPickable(&NonPickable)	Получить свойство
Object.put_NonPickable(NonPickable)	Установить свойство

NonScalableX – Не масштабировать по X

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NonScalableX = Object.NonScalableX	Получить свойство (*)
Object.NonScalableX = NonScalableX	Установить свойство (*)
NonScalableX = Object.GetNonScalableX()	Получить свойство (**)
Object.SetNonScalableX(NonScalableX)	Установить свойство (**)

Синтаксис COM:

```
Object.get_NonScalableX(&NonScalableX )        Получить свойство
Object.put_NonScalableX( NonScalableX )        Установить свойство
```

NonScalableY – Не масштабировать по Y

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
NonScalableY = Object.NonScalableY             Получить свойство (* )
Object.NonScalableY = NonScalableY             Установить свойство (* )
NonScalableY = Object.GetNonScalableY()        Получить свойство (**)
Object.SetNonScalableY( NonScalableY )         Установить свойство (**)
```

Синтаксис COM:

```
Object.get_NonScalableY(&NonScalableY )        Получить свойство
Object.put_NonScalableY( NonScalableY )        Установить свойство
```

NonScalableZ – Не масштабировать по Z

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
NonScalableZ = Object.NonScalableZ             Получить свойство (* )
Object.NonScalableZ = NonScalableZ             Установить свойство (* )
NonScalableZ = Object.GetNonScalableZ()        Получить свойство (**)
Object.SetNonScalableZ( NonScalableZ )         Установить свойство (**)
```

Синтаксис COM:

```
Object.get_NonScalableZ(&NonScalableZ )        Получить свойство
Object.put_NonScalableZ( NonScalableZ )        Установить свойство
```

ObjectID – Идентификатор объекта

Интерфейс...

Тип данных:long

Синтаксис Automation:

```
ObjectID = Object.ObjectID                     Получить свойство (* )
ObjectID = Object.GetObjectID()                Получить свойство (**)
```

Синтаксис COM:

Object.get_ObjectID(&ObjectID) — Получить свойство

Примечание:

1. Позволяет получить идентификатор объекта. По идентификатору объекта осуществляется установка видимости и удаление объектов.
2. Свойство доступно только для чтения.

Visible – Идентификатор объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Object.Visible(frame) = Visible — Установить свойство (*)
Object.SetVisible(frame, Visible) — Установить свойство (**)

Синтаксис COM:

Object.put_Visible(frame, Visible) — Установить свойство

Входные параметры:

frame - указатель на интерфейс окна документа IDocumentFrame.

Примечание:

1. Позволяет изменить видимость объекта в заданном окне. Если frame == NULL, то объект будет отображаться во всех окнах документа.
2. Свойство доступно только для изменения.

IExternalTessellationObject – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete( );

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Перед удалением объекта нужно выключить видимость объекта в окнах документа.
1. Для удаления группы объектов используется метод IExternalTessellationManager::DeleteObjects. Для удаления всех внешних объектов используется метод IExternalTessellationManager::Clear.

SelectTextureImage – Задать изображение текстуры по идентификатору

Интерфейс...

Синтаксис Automation:

BOOL SelectTextureImage(long TexImgId);

Синтаксис COM:

HRESULT SelectTextureImage(long TexImgId, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачи,
FALSE	- в случае ошибки.

Примечание:

Метод позволяет установить для объекта текстуру, созданную функцией IExternalTessellationManager::CreateTextureImage.

SetAdvancedColor – Установить параметры цвета объекта

Интерфейс...

Синтаксис Automation:

BOOL SetAdvancedColor (long color,

double ambient,

double diffuse,

double specularity,

double shininess,

double transparency,

double emission);

Синтаксис COM:

HRESULT SetAdvancedColor (COLORREF color,

double ambient,

double diffuse,

double specularity,

double shininess,

double transparency,

double emission,

BOOL * result);

Входные параметры:

color	- цвет,
ambient	- общий свет,
diffuse	- диффузия,
specularity	- зеркальность,
shininess	- блеск,
transparency	- прозрачность,
emission	- излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetEdges – Установить ребра сетки

Интерфейс...

Синтаксис Automation:

BOOL SetEdges(VARIANT points,
VARIANT edges,
VARIANT colors);

Синтаксис COM:

HRESULT SetEdges(VARIANT points,
VARIANT edges,
VARIANT colors,
BOOL * res);

Входные параметры:

points	- массив SafeArray VT_ARRAY | VT_R8 координат вершин. Если массив координат уже задан функцией SetTessellation, то нужно передать пустой Variant VT_ARRAY | VT_R8,
edges	- массив SafeArray типа VT_ARRAY | VT_I4 индексов координат вершин,

colors - массив цветов.
Массив может быть не задан - тип VARIANT-a VT_EMPTY. В этом случае используется цвет, заданный функцией SetAdvancedColor.
Может быть задан один цвет - тип VARIANT-a VT_I4.
Может быть задан цвет для каждой точки - тип VARIANT-a VT_ARRAY | VT_I4.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

1. Массив ребер может содержать одно или несколько ребер. Каждое ребро может иметь две или более вершины.
2. Для описания ребер используется последовательность:

   [[count0, i00, i01, ...] - первое ребро

   [count1, i10, i11, ...] - второе ребро

   ...

   [counti, ii0, ii1, ...]] - i-тое ребро,

   где count0, count1, counti - количество вершин в ребре, i00, i01, i10, i11, ii0, ii1 - индексы координат точек из массива points.

SetEdgeColors – Установить цвет ребрам сетки

Интерфейс...

Синтаксис Automation :

BOOL SetEdgeColors(VARIANT Colors);

Синтаксис COM :

HRESULT SetEdgeColors(VARIANT Colors, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Colors - безопасный массив цветов.

SetEdgeStyles – Установить стили рёбер сетки

Интерфейс...

Синтаксис Automation:

BOOL SetEdgeStyles(VARIANT Styles);

Синтаксис COM:

HRESULT SetEdgeStyles(VARIANT Styles, BOOL * Result);

Входные параметры:

Styles - массив VT_ARRAY | VT_I4 стилей ребер.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetEdgeWidths – Установить толщины ребер сетки

Интерфейс...

Синтаксис Automation:

BOOL SetEdgeWidths(VARIANT Widths);

Синтаксис COM:

HRESULT SetEdgeWidths(VARIANT Widths, BOOL * Result);

Входные параметры:

Widths - массив VT_ARRAY | VT_R8 значений толщины ребер.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetFacetMode – Установить режим отображения граней

Интерфейс...

Синтаксис Automation:

BOOL SetFacetMode(BOOL SideMode,
ksFacetCullingMode CullingMode);

Синтаксис COM:

HRESULT SetFacetMode(BOOL SideMode,
ksFacetCullingMode CullingMode, BOOL * Res);

Входные параметры:

SideMode - TRUE - объект является двусторонним,
- FALSE - объект является односторонним,
CullingMode - режим отображения граней ksFacetCullingMode.

Возвращаемое значение:

TRUE - в случае удачи.

SetPlaces – Установить матрицы для отрисовки объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPlaces(VARIANT places);

Синтаксис COM:

HRESULT SetPlaces(VARIANT places, BOOL * res);

Входные параметры:

places - массив SafeArray типа VT_ARRAY | VT_R8. В массиве может быть передана одна или несколько матриц размером 4*4. Если передано несколько матриц, объект будет отрисован столько раз, сколько передано матриц. Используется для многократной отрисовки одинаковых объектов. Матрицы передаются в виде одномерного массива.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetTessellation – Установить параметры триангуляции

Интерфейс...

Синтаксис Automation:

BOOL SetTessellation(VARIANT points,
VARIANT indexes,
VARIANT normals,
VARIANT colors);

Синтаксис COM:

HRESULT SetTessellation(VARIANT points,
VARIANT indexes,
VARIANT normals,
VARIANT colors,
BOOL * Result);

Входные параметры:

points - массив SafeArray VT_ARRAY | VT_R8 координат точек триангуляционной сетки; координаты в массиве лежат в последовательности x0, y0, z0, x1, y1, z1, ... xi, yi, zi,

indexes	- массив SafeArray VT_ARRAY | VT_I4 индексов вершин треугольников; по индексу из массива indexes достаются координаты вершин из массива points,
normals	- массив SafeArray VT_ARRAY | VT_R8 единичных векторов нормалей для вершин. Если направление нормалей во всех вершинах совпадают, можно добавить направление только одной нормали. Иначе количество нормалей должно быть равно количеству точек в массиве points,
colors	- массив цветов в точках. Массив может быть не задан - тип VARIANT-а VT_EMPTY. В этом случае используется цвет, заданный функцией SetAdvancedColor. Может быть задан один цвет - тип VARIANT-а VT_I4. Может быть задан цвет для каждой точки - тип VARIANT-а VT_ARRAY | VT_I4.

SetTextureImage – Установить изображение текстуры

Интерфейс...

Синтаксис Automation:

```
BOOL SetTextureImage( long    Width,
long    Heigh,
BOOL    RGBA,
BOOL    WrapMode,
BOOL    FiltMode,
VARIANT ImageData );
```

Синтаксис COM:

```
HRESULT SetTextureImage( long    Width,
long    Heigh,
BOOL    RGBA,
BOOL    WrapMode,
BOOL    FiltMode,
VARIANT ImageData,
BOOL *  Res );
```

Входные параметры:

Width, Heigh	- размер изображения, являющегося элементом текстуры; Рекомендуется использовать изображения, значения размеров которых являются целой положительной степенью числа 2; в противном случае, объекты будут неправильно выведены на печать и конвертированы в растр,
RGBA	- TRUE - RGBA, - FALSE - RGB,
WrapMode	режим повторения: - TRUE - GL_CLAMP, - FALSE - GL_REPEAT,

FiltMode режим фильтрации:
- TRUE - GL_LINEAR,
- FALSE - GL_NEAREST,
ImageData - массив байтов изображения текстуры VT_ARRAY|VT_UI1.

Возвращаемое значение:

TRUE - в случае удачи.

SetTexturePoints – Установить координаты вывода текстуры

Интерфейс...

Синтаксис Automation:

BOOL SetTexturePoints(BOOL Image2D, VARIANT ImagePoints);

Синтаксис COM:

HRESULT SetTexturePoints(BOOL Image2D, VARIANT ImagePoints, BOOL * Res);

Входные параметры:

Image2D - TRUE - двумерные координаты,
- FALSE - одномерные координаты,
ImagePoints - Координаты в диапазоне 0...1.

Возвращаемое значение:

TRUE - в случае удачи.

Интерфейс IGabaritObject

Вспомогательные данные для изменения габарита документа.

Иерархия:

IDispatch

IGabaritObject

Примечание:

Данный интерфейс может быть получен в событиях интерфейса событий окна документа ksDocumentFrameNotify, ksDocumentFrameNotify::AddGabarit.

IGabaritObject – методы

AddGabarit – Добавить дополнительный габарит к габариту документа

Интерфейс...

Синтаксис Automation:

void AddGabarit (double x1,
double y1,
double z1,
double x2,
double y2,
double z2);

Синтаксис COM:

HRESULT AddGabarit ([in]double x1,
[in]double y1,
[in]double z1,
[in]double x2,
[in]double y2,
[in]double z2);

Выходные параметры:

x1, y1, z1, x2, y2, z2 - координаты габаритной рамки добавляемого дополнительного габарита.

GetCurrentGabarit – Получить текущий габарит документа

Интерфейс...

Синтаксис Automation:

void GetCurrentGabarit (long nPage,
double* p1X,
double* p1Y,
double* p1Z,
double* p2X,
double* p2Y,
double* p2Z);

Синтаксис COM:

HRESULT GetCurrentGabarit ([in]long nPage,
[out]double *p1X,
[out]double *p1Y,
[out]double *p1Z,
[out]double *p2X,
[out]double *p2Y,
[out]double *p2Z)

Входные параметры:

nPage - номер листа, начиная с единицы.

Выходные параметры:

p1X, p1Y, p1Z, p2X, p2Y, p2Z - координаты габаритной рамки документа.

Примечание:

Метод позволяет получить текущий габарит документа в миллиметрах. Для многолистовых документов можно получить габарит конкретного листа.

GetGabaritModifying – Изменился ли габарит документа

Интерфейс...

Синтаксис Automation:

BOOL GetGabaritModifying();

Синтаксис COM:

HRESULT GetGabaritModifying ([out, retval]VARIANT_BOOL * DocGabaritModify);

Возвращаемое значение:

TRUE - габарит документа изменен,
FALSE - габарит документа не изменен.

Примечание:

В момент отработки события **Определение габаритов документа** метод позволяет узнать менялся ли габарит документа или нет.

Интерфейс ksGLObject

Вспомогательные данные для создания листа в контексте OpenGL.

Примечание:

1. Данный интерфейс нужно использовать при отрисовке с помощью OpenGL из внешнего приложения (контроллера). При работе из подключаемой библиотеки (dll) данный интерфейс использовать нельзя. Более подробное описание методов смотрите в описании библиотеки OpenGL.
2. Данный интерфейс может быть получен в событиях интерфейса событий окна документа ksDocumentFrameNotify:

- ksDocumentFrameNotify::BeginPaintGL,
- ksDocumentFrameNotify::ClosePaintGL.

ksGLObject – методы

Синтаксис Automation:

BOOL glBegin(long mode);

BOOL glEnd();

BOOL glEnable(long cap);

BOOL glDisable(long cap);

BOOL glColor3d(double r, double g, double b);

BOOL glLineWidth(double w);

BOOL glLineStipple(long factor, short pattern);

BOOL glPointSize(double w);

BOOL glPolygonMode(long face, long mode);

BOOL glVertex2d(double x, double y);

BOOL glVertex2dv(double* pData, long countDouble);

BOOL glVertex3d(double x, double y, double z);

BOOL glVertex3dv(double* pData, long countDouble);

BOOL glVertex4d(double x, double y, double z, double w);

BOOL glVertex4dv(double* pData, long countDouble);

Интерфейс IPaintObject

Интерфейс вспомогательных данные для отрисовки документа.

Иерархия:

IDispatch

IPaintObject

Примечание:

Данный интерфейс может быть получен в событиях интерфейса событий окна документа ksDocumentFrameNotify:

- ksDocumentFrameNotify::BeginPaint,
- ksDocumentFrameNotify::ClosePaint.

IPaintObject – методы

GetDIBForOutput – Создать растровое изображение документа в объект «проекция файла» для дорисовки

Интерфейс...

Синтаксис Automation:

BSTR GetDIBForOutput();

Синтаксис COM:

HRESULT GetDIBForOutput ([out, retval] BSTR* mapFileName);

Возвращаемое значение:

Имя созданного объекта «проекция файла». - в случае успешного завершения.

Примечание:

Создает объект «проекция файла» (объект ядра в Windows, создаваемый для организации совместного использования несколькими процессами одних и тех же блоков данных) для возможности дорисовки своей геометрии к изображению документа.

GetHWND – Получить дескриптор окна

Интерфейс...

Синтаксис Automation:

long GetHWND();

Синтаксис COM:

HRESULT GetHWND ([out, retval] OLE_HANDLE* hwnd);

Возвращаемое значение:

Дескриптор окна в котором идет отрисовка - в случае успешного завершения.

GetTransformMatrix – Получить коэффициенты для матрицы преобразования координат

Интерфейс...

Синтаксис Automation:

void GetTransformMatrix (double* a11,
double* a12,
double* a13,
double* a14,
double* a21,
double* a22,
double* a23,
double* a24);

Синтаксис COM:

HRESULT GetTransformMatrix ([out]double* a11,
[out]double* a12,
[out]double* a13,
[out]double* a14,
[out]double* a21,
[out]double* a22,
[out]double* a23,
[out]double* a24);

Выходные параметры:

a11, a12, a13, a14, a21, a22, a23, a24 - коэффициенты для матрицы преобразования координат.

Примечание:

Матрица для преобразования координат mm -> LP:

Xlp = Xmm * a11 + Ymm * a12 + Zmm * a13 + a14;

Ylp = Xmm * a21 + Ymm * a22 + Zmm * a23 + a24;

Отчет

Интерфейс IReportProcess

Справка системы КОМПАС...

KOMPAS.chm::/1454_136_2_Sozdanie_otcheta.htm

Интерфейс для управления процессом Создать отчет.

Иерархия:

IKompasAPIObject

IReportProcess

Описание:

Интерфейс позволяет задавать параметры для процесса **Создать отчет**.

Примечание:

Данный интерфейс можно получить с помощью свойства IKompasDocument1::GetInterface с типом ksObjectReportProcess.

IReportProcess – свойства

ActiveStyleIndex – Активный стиль отчета

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ActiveStyleIndex = Object.ActiveStyleIndex	Получить свойство (*)
Object.ActiveStyleIndex = ActiveStyleIndex	Установить свойство (*)
ActiveStyleIndex = Object.GetActiveStyleIndex()	Получить свойство (**)
Object.SetActiveStyleIndex(ActiveStyleIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_ActiveStyleIndex(&ActiveStyleIndex)	Получить свойство
Object.put_ActiveStyleIndex(ActiveStyleIndex)	Установить свойство

Свойство позволяет устанавливать и получать активный стиль отчета.

Примечание:

Тип VT_I4 - номер стиля в списке стилей. При установке стиля можно также использовать VT_R8 - уникальный номер стиля.

ReportFilter – Интерфейс управления фильтрами

Интерфейс...

Тип данных: указатель на интерфейс свойства IReportFilter

Синтаксис Automation:

ReportFilter = Object.ReportFilter	Получить свойство (*)
ReportFilter = Object.GetReportFilter()	Получить свойство (**)

Синтаксис COM:

Object.get_ReportFilter(&ReportFilter)	Получить свойство

Свойство позволяет получить интерфейс управления фильтрами.

Примечание:

Свойство только для чтения.

ReportFilterUse – Использовать фильтр

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReportFilterUse = Object.ReportFilterUse	Получить свойство (*)
Object.ReportFilterUse = ReportFilterUse	Установить свойство (*)
ReportFilterUse = Object.GetReportFilterUse()	Получить свойство (**)
Object.SetReportFilterUse(ReportFilterUse)	Установить свойство (**)

Синтаксис COM:

Object.get_ReportFilterUse(&ReportFilterUse)	Получить свойство
Object.put_ReportFilterUse(ReportFilterUse)	Установить свойство

Свойство позволяет устанавливать и получать признак использования фильтра.

StylesCount – Количество стилей отчета

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
StylesCount = Object.StylesCount          Получить свойство (* )
StylesCount = Object.GetStylesCount()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_StylesCount(                   Получить свойство
&StylesCount )
```

Свойство позволяет получить количество стилей отчета.

Примечание:

Свойство только для чтения.

Интерфейс IReportFilter

Интерфейс для управления фильтрами в процессе Создать отчет.

Иерархия

IDispatch

IReportFilter

Описание:

Позволяет управлять фильтром в процессе Создать отчет.

Примечание:

Данный интерфейс можно получить с помощью свойства IReportProcess::ReportFilter.

IReportFilter – свойства

ConditionCount – Количество условий

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
ConditionCount = Object.ConditionCount    Получить свойство (* )
ConditionCount =                          Получить свойство (**)
Object.GetConditionCount()
```

Синтаксис COM:

```
Object.get_ConditionCount(                Получить свойство
&ConditionCount)
```

Свойство позволяет получить количество условий.

Примечание:

Свойство только для чтения.

IReportFilter – методы

Clear – Удалить все условия фильтра

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetCondition – Получить условия фильтра по индексу

Интерфейс...

Синтаксис Automation:

BOOL GetCondition(long index, VARIANT * uniqId, ksReportFiltersTypeEnum * type, VARIANT * val);

Синтаксис COM:

HRESULT GetCondition(long Index, VARIANT * UniqId, ksReportFiltersTypeEnum * Type, VARIANT * Val, BOOL * Result);

Входные параметры:

index	- индекс в массиве.

Выходные параметры:

uniqId	- уникальный номер свойства VT_R8,
type	- условие фильтрации,
val	- значение фильтра; тип значения фильтра зависит от типа свойства, для которого задается фильтр.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Свойство только для чтения.

RemoveCondition – Удалить условия фильтра по индексу

Интерфейс...

Синтаксис Automation:

BOOL RemoveCondition(long index)

Синтаксис COM:

HRESULT RemoveCondition(long Index, BOOL * Result);

Входные параметры:

index	- индекс в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetCondition – Установить условия фильтра по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetCondition(long index, VARIANT uniqId, ksReportFiltersTypeEnum type, VARIANT Val);

Синтаксис COM:

HRESULT SetCondition(long Index, VARIANT UniqId, ksReportFiltersTypeEnum Type, VARIANT Val, BOOL * Result);

Входные параметры:

index	- индекс в массиве; -1 - добавить в конец,
uniqId	- уникальный номер свойства VT_R8,
type	- условие фильтрации,
val	- значение фильтра; тип значения фильтра зависит от типа свойства, для которого задается фильтр.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IReportObjectsFilter

Интерфейс фильтра объектов при создании отчета.

Иерархия

IDispatch

IReportObjectsFilter

Примечание:

Дополнительный интерфейс для таблицы отчетов IReport.

Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IReportObjectsFilter – свойства

Bodies – Собирать тела

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Bodies = Object.Bodies	Получить свойство(*)
Object.Bodies = Bodies	Установить свойство (*)
Bodies = Object.GetBodies()	Получить свойство (**)
Object.SetBodies(Bodies)	Установить свойство (**)

Синтаксис COM:

Object.get_Bodies(&Bodies)	Получить свойство
Object.put_Bodies(Bodies)	Установить свойство

InsertionFragments – Собирать вставки фрагментов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InsertionFragments = Object.InsertionFragments	Получить свойство(*)
Object.InsertionFragments = InsertionFragments	Установить свойство (*)
InsertionFragments = Object.GetInsertionFragments()	Получить свойство (**)
Object.SetInsertionFragments(InsertionFragments)	Установить свойство (**)

Синтаксис COM:

Object.get_InsertionFragments(&InsertionFragments)	Получить свойство
Object.put_InsertionFragments(InsertionFragments)	Установить свойство

InsertionViews – Собирать вставки видов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InsertionViews = Object.InsertionViews	Получить свойство(*)
Object.InsertionViews = InsertionViews	Установить свойство (*)
InsertionViews = Object.GetInsertionViews()	Получить свойство (**)
Object.SetInsertionViews(InsertionViews)	Установить свойство (**)

Синтаксис COM:

Object.get_InsertionViews(&InsertionViews)	Получить свойство
Object.put_InsertionViews(InsertionViews)	Установить свойство

LocalParts – Собирать локальные вставки компонентов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LocalParts = Object.LocalParts	Получить свойство(*)
Object.LocalParts = LocalParts	Установить свойство (*)
LocalParts = Object.GetLocalParts()	Получить свойство (**)
Object.SetLocalParts(LocalParts)	Установить свойство (**)

Синтаксис COM:

Object.get_LocalParts(&LocalParts)	Получить свойство
Object.put_LocalParts(LocalParts)	Установить свойство

MacroObjects2D – Собирать макроэлементы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MacroObjects2D = Object.MacroObjects2D	Получить свойство(*)
Object.MacroObjects2D = MacroObjects2D	Установить свойство (*)
MacroObjects2D = Object.GetMacroObjects2D()	Получить свойство (**)
Object.SetMacroObjects2D(MacroObjects2D)	Установить свойство (**)

Синтаксис COM:

Object.get_MacroObjects2D(&MacroObjects2D)	Получить свойство
Object.put_MacroObjects2D(MacroObjects2D)	Установить свойство

ModelObjects – Собирать объекты моделей с ассоциативных видов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ModelObjects = Object.ModelObjects	Получить свойство(*)
Object.ModelObjects = ModelObjects	Установить свойство (*)
ModelObjects = Object.GetModelObjects()	Получить свойство (**)
Object.SetModelObjects(ModelObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_ModelObjects(&ModelObjects)	Получить свойство
Object.put_ModelObjects(ModelObjects)	Установить свойство

Parts – Собирать компоненты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Parts = Object.Parts	Получить свойство(*)
Object.Parts = Parts	Установить свойство (*)
Parts = Object.GetParts()	Получить свойство (**)
Object.SetParts(Parts)	Установить свойство (**)

Синтаксис COM:

Object.get_Parts(&Parts)	Получить свойство
Object.put_Parts(Parts)	Установить свойство

Views – Собирать виды (Объекты группируются по видам)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Views = Object.Views	Получить свойство(*)
Object.Views = Views	Установить свойство (*)
Views = Object.GetViews()	Получить свойство (**)
Object.SetViews(Views)	Установить свойство (**)

Синтаксис COM:

Object.get_Views(&Views)	Получить свойство
Object.put_Views(Views)	Установить свойство

Таблицы отчетов

Интерфейс IReport

Интерфейс таблицы отчетов.

Иерархия:

IDispatch

IKompasAPIObject

IReport

Интерфейс можно получить у интерфейса ассоциативной таблицы при помощи свойства IAssociationTable::Report или у интерфейса Менеджера свойств при помощи метода IPropertyMng::GetReport.

Посредством вызова метода Unknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы:

IReportParam – Интерфейс параметров отчета,

IReportTable – Интерфейс таблицы данных,

IChooseManager – Интерфейс менеджера выбора (подсветки) объектов.

IReport – свойства

CurrentReportStyle – Стиль отчета

Интерфейс...

Тип данных: Указатель на интерфейс IReportStyle

Синтаксис Automation:

CurrentReportStyle = Object.CurrentReportStyle	Получить свойство (*)
CurrentReportStyle = Object.GetCurrentReportStyle()	Получить свойство (**)

Синтаксис COM:

Object.get_CurrentReportStyle(&CurrentReportStyle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CurrentStyleIndex – Индекс текущего стиля

Интерфейс...

Тип данных: long

Синтаксис Automation:

CurrentStyleIndex = Object.CurrentStyleIndex	Получить свойство(*)
Object.CurrentStyleIndex = CurrentStyleIndex	Установить свойство (*)
CurrentStyleIndex = Object.GetCurrentStyleIndex()	Получить свойство (**)
Object.SetCurrentStyleIndex(CurrentStyleIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentStyleIndex(&CurrentStyleIndex))	Получить свойство
Object.put_CurrentStyleIndex(CurrentStyleIndex)	Установить свойство

ReportFilter – Интерфейс управления фильтрами

Интерфейс...

Тип данных: Указатель на интерфейс IReportFilter

Синтаксис Automation:

ReportFilter = Object.ReportFilter	Получить свойство (*)
ReportFilter = Object.GetReportFilter()	Получить свойство (**)

Синтаксис COM:

Object.get_ReportFilter(&ReportFilter)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ReportStyle – Стиль отчета по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IReportStyle

Синтаксис Automation:

ReportStyle = Object.ReportStyle(Index)	Получить свойство (*)
ReportStyle = Object.GetReportStyle(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_ReportStyle(Index, &ReportStyle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ReportType – Тип отчета (по свойствам, по исполнениям, по переменным)

Интерфейс...

Тип данных: из перечисления ksReportTypeEnum

Синтаксис Automation:

ReportType = Object.ReportType	Получить свойство (*)
ReportType = Object.GetReportType()	Получить свойство (**)

Синтаксис COM:

Object.get_ReportType(&ReportType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ShowAllObjects – Показывать все объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowAllObjects = Object.ShowAllObjects	Получить свойство(*)
Object.ShowAllObjects = ShowAllObjects	Установить свойство (*)
ShowAllObjects = Object.GetShowAllObjects( )	Получить свойство (**)
Object.SetShowAllObjects(ShowAllObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowAllObjects(&ShowAllObjects)	Получить свойство
Object.put_ShowAllObjects(ShowAllObjects)	Установить свойство

SourceFileName – Имя файла

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

SourceFileName = Object.SourceFileName	Получить свойство (*)
SourceFileName = Object.GetSourceFileName()	Получить свойство (**)

Синтаксис COM:

Object.get_SourceFileName(&SourceFileName)	Получить свойство

Примечание:

Свойство доступно только для чтения.

StylesCount – Количество стилей

Тип данных: long

Синтаксис Automation:

StylesCount = Object.StylesCount	Получить свойство (*)

StylesCount = Object.GetStylesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_StylesCount(&StylesCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

UseReportFilter – Использовать фильтр

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseReportFilter = Object.UseReportFilter	Получить свойство(*)
Object.UseReportFilter = UseReportFilter	Установить свойство (*)
UseReportFilter = Object.GetUseReportFilter( )	Получить свойство (**)
Object.SetUseReportFilter(UseReportFilter)	Установить свойство (**)

Синтаксис COM:

Object.get_UseReportFilter (&UseReportFilter)	Получить свойство
Object.put_UseReportFilter (UseReportFilter)	Установить свойство

IReport – методы

AddStyle – Добавить новый стиль

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddStyle(BSTR FileName, double Id);

Синтаксис COM:

HRESULT AddStyle(BSTR FileName, double Id, IReportStyle ** Result);

Возвращаемое значение:

- Указатель на интерфейс стиля отчета IReportStyle.

Входные параметры:

FileName	- имя файла библиотеки стилей отчетов,
Id	- идентификатор стиля.

LoadSourceDocument – Документ

Интерфейс...

Синтаксис Automation:

LPDISPATCH LoadSourceDocument(BOOL Visible, BOOL ReadOnly);

Синтаксис COM:

HRESULT LoadSourceDocument(BOOL Visible, BOOL ReadOnly, IKompasDocument ** Result);

Возвращаемое значение:

Указатель на документ IKompasDocument.

Входные параметры:

Visible	- видимый режим,
ReadOnly	- режим только для чтения.

Rebuild – Перестроить отчет

Интерфейс...

Синтаксис Automation:

BOOL Rebuild();

Синтаксис COM:

HRESULT Rebuild(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SaveAs – Отчет сохранить как

Интерфейс...

Синтаксис Automation:

BOOL SaveAs(VARIANT FileName);

Синтаксис COM:

HRESULT SaveAs(VARIANT FileName, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

FileName	- имя сохраняемого файла или указатель на открытый документ.

Интерфейс IReportStyle

Интерфейс стиля отчета.

Иерархия:

IDispatch

IKompasAPIObject

IReportStyle

Интерфейс можно получить у интерфейса таблицы отчетов с помощью метода IReport::AddStyle или свойств IReport::ReportStyle и IReport::CurrentReportStyle.

IReportStyle – свойства

Column – Колонка отчета

Интерфейс...

Тип данных:Указатель на интерфейс IReportStyleColumn.

Синтаксис Automation:

Column = Object.Column(Index)	Получить свойство (*)
Column = Object.GetColumn(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Column(Index, &Column)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ColumnsCount – Количество колонок

Интерфейс...

Тип данных:long

Синтаксис Automation:

ColumnsCount = Object.ColumnsCount	Получить свойство (*)
ColumnsCount = Object.GetColumnsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(&ColumnsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ColumnNumberingType – Формат нумерации столбцов отчета

Интерфейс...

Тип данных: из перечисления ksNumberingTypeEnum

Синтаксис Automation:

ColumnNumberingType = Object.ColumnNumberingType	Получить свойство(*)
Object.ColumnNumberingType = ColumnNumberingType	Установить свойство (*)
ColumnNumberingType = Object.GetColumnNumberingType()	Получить свойство (**)
Object.SetColumnNumberingType(ColumnNumberingType)	Установить свойство (**)

Синтаксис COM:

Object.get_ColumnNumberingType(&ColumnNumberingType)	Получить свойство

Object.put_ColumnNumberingType(ColumnNumberingType)	Установить свойство

ColumnNumberingInitVal – Нумеровать колонки, начиная с

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnNumberingInitVal = Object.ColumnNumberingInitVal	Получить свойство(*)
Object.ColumnNumberingInitVal = ColumnNumberingInitVal	Установить свойство (*)
ColumnNumberingInitVal = Object.GetColumnNumberingInitVal()	Получить свойство (**)
Object.SetColumnNumberingInitVal(ColumnNumberingInitVal)	Установить свойство (**)

Синтаксис COM:

Object.get_ColumnNumberingInitVal(&ColumnNumberingInitVal)	Получить свойство
Object.put_ColumnNumberingInitVal(ColumnNumberingInitVal)	Установить свойство

DivideIntoPage – Разбивать на страницы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DivideIntoPage = Object.DivideIntoPage	Получить свойство(*)
Object.DivideIntoPage = DivideIntoPage	Установить свойство (*)
DivideIntoPage = Object.GetDivideIntoPage( )	Получить свойство (**)
Object.SetDivideIntoPage(DivideIntoPage)	Установить свойство (**)

Синтаксис COM:

Object.get_DivideIntoPage (&DivideIntoPage)	Получить свойство
Object.put_DivideIntoPage (DivideIntoPage)	Установить свойство

DrawBottom – true – сверху вниз (как обычно), false – отрисовка снизу вверх

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DrawBottom = Object.DrawBottom	Получить свойство(*)
Object.DrawBottom = DrawBottom	Установить свойство (*)
DrawBottom = Object.GetDrawBottom()	Получить свойство (**)
Object.SetDrawBottom(DrawBottom)	Установить свойство (**)

Синтаксис COM:

Object.get_DrawBottom(&DrawBottom)	Получить свойство
Object.put_DrawBottom(DrawBottom)	Установить свойство

Id – Уникальный номер

Интерфейс...

Тип данных: double

Синтаксис Automation:

Id = Object.Id	Получить свойство (*)
Id = Object.GetId()	Получить свойство (**)

Синтаксис COM:

Object.get_Id(&Id)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Name – Имя

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство(*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Примечание:

Свойство доступно только для чтения.

RowHeight – Высота строки ячейки таблицы, мм

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowHeight = Object.RowHeight	Получить свойство(*)
Object.RowHeight = RowHeight	Установить свойство (*)
RowHeight = Object.GetRowHeight()	Получить свойство (**)
Object.SetRowHeight(RowHeight)	Установить свойство (**)

Синтаксис COM:

Object.get_RowHeight(&RowHeight)	Получить свойство
Object.put_RowHeight(RowHeight)	Установить свойство

RowsCount – Количество строк таблицы

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = Object.RowsCount	Получить свойство(*)
Object.RowsCount = RowsCount	Установить свойство (*)
RowsCount = Object.GetRowsCount()	Получить свойство (**)
Object.SetRowsCount(RowsCount)	Установить свойство (**)

Синтаксис COM:

Object.get_RowsCount(&RowsCount)	Получить свойство
Object.put_RowsCount(RowsCount)	Установить свойство

RowsNumberingType – Нумеровать строки

Интерфейс...

Тип данных: из перечисления ksRowsNumberingTypeEnum

Синтаксис Automation:

RowsNumberingType = Object.RowsNumberingType	Получить свойство(*)
Object.RowsNumberingType = RowsNumberingType	Установить свойство (*)
RowsNumberingType = Object.GetRowsNumberingType()	Получить свойство (**)
Object.SetRowsNumberingType(RowsNumberingType)	Установить свойство (**)

Синтаксис COM:

Object.get_RowsNumberingType(&RowsNumberingType)	Получить свойство
Object.put_RowsNumberingType(RowsNumberingType)	Установить свойство

ShowEmptyRows – Выводить пустые строки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowEmptyRows = Object.ShowEmptyRows	Получить свойство(*)

Object.ShowEmptyRows = ShowEmptyRows	Установить свойство (*)
ShowEmptyRows = Object.GetShowEmptyRows()	Получить свойство (**)
Object.SetShowEmptyRows(ShowEmptyRows)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowEmptyRows(&ShowEmptyRows)	Получить свойство
Object.put_ShowEmptyRows(ShowEmptyRows)	Установить свойство

ShowTitle – Отображать заголовок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowTitle = Object.ShowTitle	Получить свойство(*)
Object.ShowTitle = ShowTitle	Установить свойство (*)
ShowTitle = Object.GetShowTitle()	Получить свойство (**)
Object.SetShowTitle(ShowTitle)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowTitle(&ShowTitle)	Получить свойство
Object.put_ShowTitle(ShowTitle)	Установить свойство

TitleHeight – Высота заголовка, мм

Интерфейс...

Тип данных: long

Синтаксис Automation:

TitleHeight = Object.TitleHeight	Получить свойство(*)
Object.TitleHeight = TitleHeight	Установить свойство (*)

TitleHeight = Object.GetTitleHeight() — Получить свойство (**)

Object.SetTitleHeight(TitleHeight) — Установить свойство (**)

Синтаксис COM:

Object.get_TitleHeight(&TitleHeight) — Получить свойство

Object.put_TitleHeight(TitleHeight) — Установить свойство

IReportStyle – методы

AddColumn – Добавить колонку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddColumn(long IndexAt, BSTR Name, IProperty * Property);

Синтаксис COM:

HRESULT AddColumn(long IndexAt, BSTR Name, IProperty * Property, IReportStyleColumn ** Result);

Возвращаемое значение:

- указатель на интерфейс колонки стиля.

Входные параметры:

IndexAt - индекс добавляемой колонки,
Name - имя колонки,
Property - указатель на интерфейс свойства, которое будет отображаться в колонке. Можно передать NULL.

Clear – Очистить стиль

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

ClearSortParameters – Очистить параметры сортировки

Интерфейс...

Синтаксис Automation:

BOOL ClearSortParameters();

Синтаксис COM:

HRESULT ClearSortParameters(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DeleteColumn – Удалить колонку

Интерфейс...

Синтаксис Automation:

BOOL DeleteColumn(VARIANT Index);

Синтаксис COM:

HRESULT DeleteColumn(VARIANT Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс удаляемой колонки.

GetSortParameters – Параметры сортировки

Интерфейс...

Синтаксис Automation:

BOOL GetSortParameters(long Level, long * ColumnNumber, ksSortTypeEnum * SortType);

Синтаксис COM:

HRESULT GetSortParameters(long Level, long * ColumnNumber, ksSortTypeEnum * SortType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Level - уровень сортировки.

Выходные параметры:

ColumnNumber - номер колонки,
SortType - тип сортировки.

Init – Инициализация стиля параметрами по умолчанию

Интерфейс...

Синтаксис Automation:

BOOL Init(ksReportStyleInitEnum Type);

Синтаксис COM:

HRESULT Init(ksReportStyleInitEnum Type, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Type - тип инициализации стиля.

SetSortParameters – Параметры сортировки

Интерфейс...

Синтаксис Automation:

BOOL SetSortParameters(long Level, long ColumnNumber, ksSortTypeEnum SortType);

Синтаксис COM:

HRESULT SetSortParameters(long Level, long ColumnNumber, ksSortTypeEnum SortType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Level - уровень сортировки,
ColumnNumber - номер колонки,
SortType - тип сортировки.

SwapColumn – Поменять колонки местами

Интерфейс...

Синтаксис Automation:

BOOL SwapColumn(long Index1, long Index2);

Синтаксис COM:

HRESULT SwapColumn(long Index1, long Index2, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Index1 - индекс 1-й колонки,
Index2 - индекс 2-й колонки.

Интерфейс IReportStyleColumn

Интерфейс колонки стиля отчета.

Иерархия:

IDispatch

IKompasAPIObject

IReportStyleColumn

Интерфейс можно получить у интерфейса таблицы данных с помощью свойства IReportTable::StyleColumn, у интерфейса стиля отчета с помощью свойства IReportStyle::Column или с помощью метода IReportStyle::AddColumn.

IReportStyleColumn – свойства

CombineCells – Объединять ячейки с одинаковыми значениями в колонке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CombineCells = Object.CombineCells                  Получить свойство (* )
Object.CombineCells = CombineCells                  Установить свойство (* )
CombineCells = Object.GetCombineCells()             Получить свойство (**)
Object.SetCombineCells( CombineCells )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CombineCells( &CombineCells )    Получить свойство
Object.put_CombineCells(  CombineCells )    Установить свойство
```

GroupeType – Тип группировки колонки отчета

Интерфейс...

Тип данных: из перечисления ksGroupeTypeEnum

Синтаксис Automation:

```
GroupeType = Object.GroupeType                      Получить свойство (* )
Object.GroupeType = GroupeType                      Установить свойство (* )
GroupeType = Object.GetGroupeType()                 Получить свойство (**)
Object.SetGroupeType( GroupeType )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_GroupeType( &GroupeType )        Получить свойство
Object.put_GroupeType(  GroupeType )        Установить свойство
```

Header – Имя колонки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Header = Object.Header                              Получить свойство (* )
Object.Header = Header                              Установить свойство (* )
Header = Object.GetHeader()                         Получить свойство (**)
Object.SetHeader( Header )                          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Header( &Header )    Получить свойство
Object.put_Header(  Header )    Установить свойство
```

Id – Уникальный номер

Интерфейс...

Тип данных: double

Синтаксис Automation:

Id = Object.Id	Получить свойство(*)
Id = Object.GetId()	Получить свойство (**)

Синтаксис COM:

Object.get_Id(&Id)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Property – Описание типа свойства

Интерфейс...

Тип данных: Указатель на интерфейс IProperty

Синтаксис Automation:

Property = Object.Property	Получить свойство (*)
Object.Property = Property	Установить свойство (*)
Property = Object.GetProperty()	Получить свойство (**)
Object.SetProperty(Property)	Установить свойство (**)

Синтаксис COM:

Object.get_Property(&Property)	Получить свойство
Object.put_Property(Property)	Установить свойство

PropertyLevel – Уровень, с которого требуется получать значение свойства

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

PropertyLevel = Object.PropertyLevel	Получить свойство (*)
Object.PropertyLevel = PropertyLevel	Установить свойство (*)
PropertyLevel = Object.GetPropertyLevel()	Получить свойство (**)
Object.SetPropertyLevel(PropertyLevel)	Установить свойство (**)

Синтаксис COM:

Object.get_PropertyLevel(&PropertyLevel)	Получить свойство
Object.put_PropertyLevel(PropertyLevel)	Установить свойство

TextAlign – Выравнивание в колонке отчета

Интерфейс...

Тип данных: из перечисления ksAlignEnum

Синтаксис Automation:

TextAlign = Object.TextAlign	Получить свойство (*)
Object.TextAlign = TextAlign	Установить свойство (*)
TextAlign = Object.GetTextAlign()	Получить свойство (**)
Object.SetTextAlign(TextAlign)	Установить свойство (**)

Синтаксис COM:

Object.get_TextAlign(&TextAlign)	Получить свойство
Object.put_TextAlign(TextAlign)	Установить свойство

UserColumn – Пользовательская колонка без свойства

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UserColumn = Object.UserColumn	Получить свойство(*)
UserColumn = Object.GetUserColumn()	Получить свойство (**)

Синтаксис COM:

Object.get_UserColumn(&UserColumn)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ValueFormat – Формат значения

Интерфейс...

Тип данных: из перечисления ksValueFormatEnum

Синтаксис Automation:

ValueFormat = Object.ValueFormat	Получить свойство (*)
Object.ValueFormat = ValueFormat	Установить свойство (*)
ValueFormat = Object.GetValueFormat()	Получить свойство (**)
Object.SetValueFormat(ValueFormat)	Установить свойство (**)

Синтаксис COM:

```
Object.get_ValueFormat( &ValueFormat )      Получить свойство
Object.put_ValueFormat(  ValueFormat )      Установить свойство
```

Visible – Отображать в отчете

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Visible = Object.Visible                    Получить свойство (* )
Object.Visible = Visible                    Установить свойство (* )
Visible = Object.GetVisible()               Получить свойство (**)
Object.SetVisible( Visible )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Visible( &Visible )              Получить свойство
Object.put_Visible(  Visible )              Установить свойство
```

Width – Ширина колонки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width                        Получить свойство (* )
Object.Width = Width                        Установить свойство (* )
Width = Object.GetWidth()                   Получить свойство (**)
Object.SetWidth( Width )                    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width )                  Получить свойство
Object.put_Width(  Width )                  Установить свойство
```

WordWrap – Переносить по словам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
WordWrap = Object.WordWrap                  Получить свойство (* )
Object.WordWrap = WordWrap                  Установить свойство (* )
WordWrap = Object.GetWordWrap()             Получить свойство (**)
Object.SetWordWrap( WordWrap )              Установить свойство (**)
```

Синтаксис COM:

Object.get_WordWrap(&WordWrap)	Получить свойство
Object.put_WordWrap(WordWrap)	Установить свойство

Интерфейс IReportTable

Интерфейс таблицы данных.

Иерархия:

IDispatch

IReportTable

IReportTable является дополнительным для интерфейса таблицы отчетов IReport. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IReportTable – свойства

CellValue – Значение в ячейке

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

CellValue = Object.CellValue(NRow, NCol)	Получить свойство (*)
Object.CellValue(NRow, NCol) = CellValue	Установить свойство (*)
CellValue = Object.GetCellValue(NRow, NCol)	Получить свойство (**)
Object.SetCellValue(NRow, NCol, CellValue)	Установить свойство (**)

Синтаксис COM:

Object.get_CellValue(NRow, NCol, &CellValue)	Получить свойство
Object.put_CellValue(NRow, NCol, CellValue)	Установить свойство

Входные параметры:

long NRow	- номер строки,
long NCol	- номер колонки.

ColumnsCount – Количество столбцов

Интерфейс...

Тип данных: long.

Синтаксис Automation:

ColumnsCount = Object.ColumnsCount	Получить свойство(*)
ColumnsCount = Object.GetColumnsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(&ColumnsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DisplayMode – TRUE – Работает с отображаемой таблицей (скрытые колонки и строки не выдаются)

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

DisplayMode = Object.DisplayMode	Получить свойство (*)
Object.DisplayMode = DisplayMode	Установить свойство (*)
DisplayMode = Object.GetDisplayMode()	Получить свойство (**)
Object.SetDisplayMode(DisplayMode)	Установить свойство (**)

Синтаксис COM:

Object.get_DisplayMode(&DisplayMode)	Получить свойство
Object.put_DisplayMode(DisplayMode)	Установить свойство

FormatCellValue – форматированное значение в ячейке

Интерфейс...

Тип данных: BSTR.

Синтаксис Automation:

FormatCellValue = Object.FormatCellValue(NRow, NCol)	Получить свойство(*)
FormatCellValue = Object.GetFormatCellValue (NRow, NCol)	Получить свойство (**)

Синтаксис COM:

Object.get_FormatCellValue(NRow, NCol, &FormatCellValue)	Получить свойство

Входные параметры:

long NRow	- номер строки,
long NCol	- номер колонки.

Примечание:

Свойство доступно только для чтения.

PropertyKeeper – Объект отчета

Интерфейс...

Тип данных: Указатель на интерфейс IPropertyKeeper

Синтаксис Automation:

PropertyKeeper = Object.PropertyKeeper(RowIndex)	Получить свойство(*)
PropertyKeeper = Object.GetPropertyKeeper(RowIndex)	Получить свойство (**)

Синтаксис COM:

Object.get_PropertyKeeper(RowIndex, &PropertyKeeper)	Получить свойство

Примечание:

Свойство доступно только для чтения.

RowsCount – Количество строк

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = Object.RowsCount	Получить свойство(*)
RowsCount = Object.GetRowsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_RowsCount(&RowsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

StyleColumn – Параметры отображаемой в отчете колонки

Интерфейс...

Тип данных: Указатель на интерфейс IReportStyleColumn

Синтаксис Automation:

StyleColumn = Object.StyleColumn(NCol)	Получить свойство(*)
StyleColumn = Object.GetStyleColumn(NCol)	Получить свойство (**)

Синтаксис COM:

Object.get_StyleColumn(NCol, &StyleColumn)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Text – Текст в ячейке

Интерфейс...

Тип данных: Указатель на интерфейс IText.

Синтаксис Automation:

Text = Object.Text(NRow, NCol)	Получить свойство(*)
Text = Object.GetText(NRow, NCol)	Получить свойство (**)

Синтаксис COM:

Object.get_Text(NRow, NCol, &Text)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IReportParam

Интерфейс параметров отчета.

Иерархия:

IDispatch

IReportTable

IReportParam является дополнительным для интерфейса таблицы отчетов IReport. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IReportParam – свойства

BuildingType – Способ выбора объектов

Интерфейс...

Тип данных: из перечисления ksReportBuildingTypeEnum

Синтаксис Automation:

```
BuildingType = Object.BuildingType          Получить свойство (* )
Object.BuildingType = BuildingType          Установить свойство (* )
BuildingType = Object.GetBuildingType()     Получить свойство (**)
Object.SetBuildingType( BuildingType )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType( &BuildingType )    Получить свойство
Object.put_BuildingType(  BuildingType )    Установить свойство
```

LevelsCount – Число уровней

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
LevelsCount = Object.LevelsCount            Получить свойство (* )
Object.LevelsCount = LevelsCount            Установить свойство (* )
LevelsCount = Object.GetLevelsCount()       Получить свойство (**)
Object.SetLevelsCount( LevelsCount )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LevelsCount( &LevelsCount )      Получить свойство
Object.put_LevelsCount(  LevelsCount )      Установить свойство
```

PagesColumnsCount – Число колонок таблиц при установке таблиц в 2D документ

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PagesColumnsCount = Object.PagesColumnsCount                Получить свойство (* )
Object.PagesColumnsCount = PagesColumnsCount                Установить свойство (* )
PagesColumnsCount = Object.GetPagesColumnsCount()           Получить свойство (**)
Object.SetPagesColumnsCount( PagesColumnsCount )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PagesColumnsCount( &PagesColumnsCount )      Получить свойство
Object.put_PagesColumnsCount(  PagesColumnsCount )      Установить свойство
```

PageLayoutType – Тип компоновки таблиц отчета

Интерфейс...

Тип данных: из перечисления ksPageLayoutTypeEnum

Синтаксис Automation:

```
PageLayoutType = Object.PageLayoutType                      Получить свойство (* )
Object.PageLayoutType = PageLayoutType                      Установить свойство (* )
PageLayoutType = Object.GetPageLayoutType()                 Получить свойство (**)
Object.SetPageLayoutType( PageLayoutType )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PageLayoutType( &PageLayoutType )      Получить свойство
Object.put_PageLayoutType(  PageLayoutType )      Установить свойство
```

PagesRowsCount – Число рядов таблиц при установке таблиц в 2D документ

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PagesRowsCount = Object.PagesRowsCount                  Получить свойство (* )
Object.PagesRowsCount = PagesRowsCount                  Установить свойство (* )
PagesRowsCount = Object.GetPagesRowsCount()             Получить свойство (**)
Object.SetPagesRowsCount( PagesRowsCount )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PagesRowsCount( &PagesRowsCount )            Получить свойство
Object.put_PagesRowsCount(  PagesRowsCount )            Установить свойство
```

PagesSpace – Зазор между таблицами, мм

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
PagesSpace = Object.PagesSpace                          Получить свойство (* )
Object.PagesSpace = PagesSpace                          Установить свойство (* )
PagesSpace = Object.GetPagesSpace()                     Получить свойство (**)
Object.SetPagesSpace( PagesSpace )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PagesSpace( &PagesSpace )                    Получить свойство
Object.put_PagesSpace(  PagesSpace )                    Установить свойство
```

UseHyperText – Тип формирования текстов (true – ссылками, false – обычный текст)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
UseHyperText = Object.UseHyperText                      Получить свойство (* )
Object.UseHyperText = UseHyperText                      Установить свойство (* )
UseHyperText = Object.GetUseHyperText()                 Получить свойство (**)
Object.SetUseHyperText( UseHyperText )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_UseHyperText( &UseHyperText )                Получить свойство
Object.put_UseHyperText(  UseHyperText )                Установить свойство
```

Переменные

Интерфейс IVariable7

Справка системы КОМПАС...

KOMPAS.chm::/490_Glava57_Zadanie_zavisimoste.htm

Интерфейс параметрической переменной модели.

Иерархия:

IKompasAPIObject

IVariable7

Описание:

Интерфейс позволяет изменять параметры объектов модели, не прибегая к их прямому редактированию. Выражения дают возможность устанавливать зависимости между параметрами объектов.

Примечание:

Интерфейс можно получить с помощью свойств:

- IInsertionFragment::Variable,
- IInsertionFragment::Variables,
- IKompasDocument2D1::Variable,
- IKompasDocument2D1::Variables,
- IView::Variable,
- IView::Variables

и метода:

- IKompasDocument2D1::AddVariable.

IVariable7 – свойства

DeviationExpression – Выражение для переменной отклонения

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DeviationExpression = Object.DeviationExpression(HighDeviation)	Получить свойство (*)
Object.DeviationExpression(HighDeviation) = DeviationExpression	Установить свойство (*)
DeviationExpression = Object.GetDeviationExpression(HighDeviation)	Получить свойство (**)

Object.SetDeviationExpression(HighDeviation, DeviationExpression) — Установить свойство (**)

Синтаксис COM:

Object.get_DeviationExpression(HighDeviation, &DeviationExpression) — Получить свойство
Object.put_DeviationExpression(HighDeviation, DeviationExpression) — Установить свойство

Входные параметры:

BOOL — - HighDeviation - верхнее (TRUE) или нижнее (FALSE) отклонение.

DeviationOn – Включить отклонения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DeviationOn = Object.DeviationOn — Получить свойство (*)
Object.DeviationOn = DeviationOn — Установить свойство (*)
DeviationOn = Object.GetDeviationOn() — Получить свойство (**)
Object.SetDeviationOn(DeviationOn) — Установить свойство (**)

Синтаксис COM:

Object.get_DeviationOn(&DeviationOn) — Получить свойство
Object.put_DeviationOn(DeviationOn) — Установить свойство

DeviationType – Тип отклонений

Интерфейс...

Тип данных: из перечисления ksDimensionDeviationEnum

Синтаксис Automation:

DeviationType = Object.DeviationType — Получить свойство (*)
Object.DeviationType = DeviationType — Установить свойство (*)
DeviationType = Object.GetDeviationType() — Получить свойство (**)
Object.SetDeviationType(DeviationType) — Установить свойство (**)

Синтаксис COM:

Object.get_DeviationType(&DeviationType) — Получить свойство
Object.put_DeviationType(DeviationType) — Установить свойство

DisplayName – Отображаемое имя переменной

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DisplayName = Object.DisplayName	Получить свойство (*)
DisplayName = Object.GetDisplayName()	Получить свойство (**)

Синтаксис COM:

Object.get_DisplayName(&DisplayName)	Получить свойство

Свойство позволяет получить отображаемое имя переменной.

Примечание:

Свойство доступно только для чтения.

Отображаемое имя переменной может отличаться от имени переменной IVariable7::Name для внешних переменных вставки детали в сборку.

К имени переменной из источника добавляется префикс, равный имени переменной «исключить из расчета».

Expression – Выражение

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Expression = Object.Expression	Получить свойство (*)
Object.Expression = Information	Установить свойство (*)
Expression = Object.GetExpression()	Получить свойство (**)
Object.SetExpression(Expression)	Установить свойство (**)

Синтаксис COM:

Object.get_Expression(&Expression)	Получить свойство
Object.put_Expression(Expression)	Установить свойство

Свойство позволяет устанавливать и получать выражение.

External – Внешняя переменная

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
External = Object.External      Получить свойство (* )
Object.External = External      Установить свойство (* )
External =                      Получить свойство (**)
Object.GetExternal()
Object.SetExternal(             Установить свойство (**)
External )
```

Синтаксис COM:

```
Object.get_External( &External )    Получить свойство
Object.put_External(  External )    Установить свойство
```

Свойство позволяет устанавливать и получать признак внешней переменной.

HasTolerance – Отключить допуск

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
HasTolerance = Object.HasTolerance      Получить свойство (* )
Object.HasTolerance = HasTolerance      Установить свойство (* )
HasTolerance =                          Получить свойство (**)
Object.GetHasTolerance()
Object.SetHasTolerance( HasTolerance )  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HasTolerance( &HasTolerance )    Получить свойство
Object.put_HasTolerance(  HasTolerance )    Установить свойство
```

Свойство позволяет устанавливать и получать признак отключения допуска.

HighDeviation – Верхнее отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

HighDeviation = Object.HighDeviation	Получить свойство (*)
Object.HighDeviation = HighDeviation	Установить свойство (*)
HighDeviation = Object.GetHighDeviation()	Получить свойство (**)
Object.SetHighDeviation(HighDeviation)	Установить свойство (**)

Синтаксис COM:

Object.get_HighDeviation(&HighDeviation)	Получить свойство
Object.put_HighDeviation(HighDeviation)	Установить свойство

Information – Информационная переменная

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Information = Object.Information	Получить свойство (*)
Object.Information = Information	Установить свойство (*)
Information = Object.GetInformation()	Получить свойство (**)
Object.SetInformation(Information)	Установить свойство (**)

Синтаксис COM:

Object.get_Information(&Information)	Получить свойство
Object.put_Information(Information)	Установить свойство

Свойство позволяет устанавливать и получать признак информационной переменной.

IsAngular – Свойство задано в градусах

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsAngular = Object.IsAngular	Получить свойство (*)
IsAngular = Object.GetIsAngular()	Получить свойство (**)

Синтаксис COM:

Object.get_IsAngular(&IsVariable)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IsCreatedDeviationVariables – Созданы ли переменные для отклонений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCreatedDeviationVariables = Object.IsCreatedDeviationVariables	Получить свойство (*)
IsCreatedDeviationVariables = Object.GetIsCreatedDeviationVariables()	Получить свойство (**)

Синтаксис COM:

Object.get_IsCreatedDeviationVariables(&IsCreatedDeviationVariables)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IsVariable – Свойство связано с переменной

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsVariable = Object.IsVariable	Получить свойство (*)
IsVariable = Object.GetIsVariable()	Получить свойство (**)

Синтаксис COM:

Object.get_IsVariable(&IsVariable)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LinkDocumentName – Документ, в котором хранится ссылочная переменная

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
LinkDocumentName =                    Получить свойство (* )
Object.LinkDocumentName
LinkDocumentName =                    Получить свойство (**)
Object.GetLinkDocumentName()
```

Синтаксис COM:

```
    Object.get_LinkDocumentName(      Получить свойство
    &LinkDocumentName )
```

Свойство позволяет получать имя документа, в котором хранится ссылочная переменная.

Примечание:

Свойство доступно только для чтения.

LinkVariableName – Имя переменной, на которую сделана ссылка

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
LinkVariableName =                    Получить свойство (* )
Object.LinkVariableName
LinkVariableName =                    Получить свойство (**)
Object.GetLinkVariableName()
```

Синтаксис COM:

```
    Object.get_LinkVariableName(      Получить свойство
    &LinkVariableName )
```

Свойство позволяет получить имя переменной, на которую сделана ссылка.

Примечание:

Свойство доступно только для чтения.

Переменная может ссылаться на переменную из другого документа.

Ссылка формируется из имени документа IVariable7::LinkDocumentName и имени переменной.

Ссылку можно создать с помощью функции IVariable7::SetLink.

LowDeviation – Нижнее отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
LowDeviation = Object.LowDeviation          Получить свойство (* )
Object.LowDeviation = LowDeviation          Установить свойство (* )
LowDeviation =                              Получить свойство (**)
Object.GetLowDeviation()
Object.SetLowDeviation( LowDeviation )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LowDeviation( &LowDeviation )    Получить свойство
Object.put_LowDeviation(  LowDeviation )    Установить свойство
```

Name – Имя переменной

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = Object.Name          Получить свойство (* )
Name = Object.GetName()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Name( &Name )          Получить свойство
```

Свойство позволяет получить имя переменной.

Примечание:

Свойство доступно только для чтения

Note – Комментарий к переменной

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Note = Object.Note	Получить свойство (*)
Object.Note = Note	Установить свойство (*)
Note = Object.GetNote()	Получить свойство (**)
Object.SetNote(Note)	Установить свойство (**)

Синтаксис COM:

Object.get_Note(&Note)	Получить свойство
Object.put_Note(Note)	Установить свойство

Свойство позволяет устанавливать и получать комментарий к переменной.

ParameterNote – Имя параметра переменной

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ParameterNote = Object.ParameterNote	Получить свойство (*)
Object.ParameterNote = ParameterNote	Установить свойство (*)
ParameterNote = Object.GetParameterNote()	Получить свойство (**)
Object.SetParameterNote(ParameterNote)	Установить свойство (**)

Синтаксис COM:

Object.get_ParameterNote(&ParameterNote)	Получить свойство
Object.put_ParameterNote(ParameterNote)	Установить свойство

Свойство позволяет устанавливать и получать имя параметра переменной.

Property – Получить интерфейс свойства

Интерфейс...

Тип данных: Указатель на интерфейс IProperty

Синтаксис Automation:

Property = Object.Property	Получить свойство (*)
Property = Object.GetProperty()	Получить свойство (**)

Синтаксис COM:

Object.get_Property(&Property)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ReadOnly – Только чтение

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = Object.GetReadOnly()	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

Примечание:

Свойство задает доступность изменения выражения переменной.

Свойство используется для переменных документа.

Rectangle – Текст в рамочке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Rectangle = Object.Rectangle	Получить свойство (*)
Object.Rectangle = Rectangle	Установить свойство (*)
Rectangle = Object.GetRectangle()	Получить свойство (**)
Object.SetRectangle(Rectangle)	Установить свойство (**)

Синтаксис COM:

Object.get_Rectangle(&Rectangle)	Получить свойство
Object.put_Rectangle(Rectangle)	Установить свойство

Tolerance – Квалитет

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Tolerance = Object.Tolerance	Получить свойство (*)
Object.Tolerance = Tolerance	Установить свойство (*)
Tolerance = Object.GetTolerance()	Получить свойство (**)
Object.SetTolerance(Tolerance)	Установить свойство (**)

Синтаксис COM:

Object.get_Tolerance(&Tolerance)	Получить свойство
Object.put_Tolerance(Tolerance)	Установить свойство

ToleranceOn – Включить квалитет

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ToleranceOn = Object.ToleranceOn	Получить свойство (*)
Object.ToleranceOn = ToleranceOn	Установить свойство (*)
ToleranceOn = Object.GetToleranceOn()	Получить свойство (**)
Object.SetToleranceOn(ToleranceOn)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceOn(&ToleranceOn)	Получить свойство
Object.put_ToleranceOn(Tolerance On)	Установить свойство

Value – Значение переменной

Интерфейс...

Тип данных: double

Синтаксис Automation:

Value = Object.Value	Получить свойство (*)

Object.Value = Value	Установить свойство (*)
Value = Object.GetValue()	Получить свойство (**)
Object.SetValue(Value)	Установить свойство (**)

Синтаксис COM:

Object.get_Value(&Value)	Получить свойство
Object.put_Value(Value)	Установить свойство

Свойство позволяет устанавливать и получать значение переменной.

IVariable7 – методы

AddProperty – Создать свойство переменная

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddProperty();

Синтаксис COM:

HRESULT AddProperty(IProperty ** Result);

Возвращаемое значение:

- указатель на интерфейс свойства IProperty.

CreateDeviationVariables – Создать переменные для отклонений

Интерфейс...

Синтаксис Automation:

BOOL CreateDeviationVariables();

Синтаксис COM:

HRESULT CreateDeviationVariables(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачи,
FALSE	- в случае неудачи.

Delete – Удалить переменную

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Res)

Возвращаемое значение:

TRUE	- в случае удачи.

SetLink – Установить ссылку на переменную

Интерфейс...

Синтаксис Automation:

BOOL SetLink(BSTR DocumentName, BSTR VariableName)

Синтаксис COM:

HRESULT SetLink(BSTR DocumentName, BSTR VariableName, BOOL * Res)

Входные параметры:

DocumentName	- имя документа,
VariableName	- имя переменной.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс IVariableTable

Справка системы КОМПАС...

`kompas.chm::/DLG_EXTRN_VAR_TABLE.htm`

Интерфейс таблицы переменных.

Иерархия:

IKompasAPIObject

IVariableTable

Описание:

Позволяет считывать, изменять, добавлять параметры таблицы переменных компонента, применять параметры из таблицы переменных к компоненту.

Примечание:

1. Данный интерфейс можно получить:

- от интерфейса IPart7, используя свойство IPart7::VariableTable,
- от интерфейса IKompasDocument2D1, используя свойство IKompasDocument2D1::VariableTable.

2. Доступно редактирование таблицы переменных только верхнего компонента.

IVariableTable – свойства

Cell – Значение в ячейке таблицы переменных

Интерфейс...

Тип данных: double.

Синтаксис Automation:

Cell = iObject.Cell (rIndex,cIndex)	Получить свойство (*)
iObject.Cell (rIndex,cIndex) = Cell	Установить свойство (*)
Cell = iObject.GetCell (rIndex,cIndex)	Получить свойство (**)
iObject.SetCell (rIndex,cIndex,Cell)	Установить свойство (**)

Синтаксис COM:

iObject->get_Cell(rIndex,cIndex,&Cell)	Получить свойство
iObject->put_Cell (rIndex,cIndex,Cell)	Установить свойство

Входные параметры:

long rIndex	- индекс строки таблицы переменных,
long cIndex	- индекс колонки таблицы переменных.

Примечание:

Установка свойства доступна только для верхнего компонента.

ColumnsCount – Количество столбцов в таблице переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnsCount = iObject.ColumnsCount	Получить свойство (*)
ColumnsCount = iObject.GetColumnsCount()	Получить свойство (**)

Синтаксис COM:

iObject->get_ColumnsCount(&ColumnsCount)	Получить свойство

Значение свойства:

Количество столбцов	в случае успеха,
-1	в случае неудачи.

Примечание:

Свойство только для чтения.

Comment – Комментарий к строке в таблице переменных

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Comment = iObject.Comment	Получить свойство (*)
iObject.Comment = Comment	Установить свойство (*)
Comment = iObject.GetComment()	Получить свойство (**)
iObject.SetComment (Comment)	Установить свойство (**)

Синтаксис COM:

iObject->get_Comment(&Comment)	Получить свойство
iObject->put_Comment (Comment)	Установить свойство

Примечание:

Установка свойства доступна только для верхнего компонента.

RowsCount – Количество строк в таблице переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = iObject.RowsCount	Получить свойство (*)
RowsCount = iObject.GetRowsCount()	Получить свойство (**)

Синтаксис COM:

iObject->get_RowsCount(&RowsCount)	Получить свойство

Значение свойства:

Количество строк	в случае успеха,
-1	в случае неудачи.

Примечание:

Свойство только для чтения.

TableRow – Строка переменных в таблице переменных

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Row = iObject.TableRow                  Получить свойство (* )
iObject.TableRow = Row                  Установить свойство (* )
Row = iObject.GetRableRow()             Получить свойство (**)
iObject.SetTableRow (Row)               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_TableRow(&TableRow)        Получить свойство
iObject->put_TableRow (Row)             Установить свойство
```

Примечание:

Массив значений переменных передаётся через массив SAFEARRAY значений типа double.

VarName – Имя переменной в колонке таблицы переменных

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

```
Name = iObject.VarName (column)         Получить свойство (* )
iObject.VarName (column) = Name         Установить свойство (* )
Name = iObject.GetVarName (column)      Получить свойство (**)
iObject.SetVarName (column, Name)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_VarName (column, &Name)    Получить свойство
iObject->put_VarName (column, Name)     Установить свойство
```

Входные параметры:

```
long column        - индекс колонки таблицы переменных.
```

Примечание:

Установка свойства доступна только для верхнего компонента.

VarNames – Массив имен переменных в таблице переменных

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

VarNames = iObject.VarNames	Получить свойство (*)
iObject.VarNames = VarNames	Установить свойство (*)
VarNames = iObject.GetVarNames()	Получить свойство (**)
iObject.SetVarNames (VarNames)	Установить свойство (**)

Синтаксис COM:

iObject->get_VarNames(&VarNames)	Получить свойство
iObject->put_VarNames (VarNames)	Установить свойство

Примечание:

1. Массив имён переменных передаётся через массив SAFEARRAY значений типа BSTR.
2. Установка свойства доступна только для верхнего компонента.

VisualTable – Выдать таблицу в визуальном режиме (получить индекс выбранной строки)

Интерфейс...

Тип данных: long

Синтаксис Automation:

rIndex = iObject.VisualTable(WinVal,Select)	Получить свойство (*)
rIndex = iObject.GetVisualTable(WinVal,Select)	Получить свойство (**)

Синтаксис COM:

iObject->get_VisualTable (WinVal, Select, rIndex)	Получить свойство

Входные параметры:

OLE_HANDLE WinVal	- дескриптор окна (может быть задан ноль - признак главного окна),
BOOL Select	- признак открытия окна только для чтения и выбора строки из таблицы.

Возвращаемое значение:

индекс выбранной строки - в случае успеха,
-1 - в случае неудачи.

Примечание:

1. Свойство доступно только для чтения.
2. Если задан признак выбора Select = TRUE, по нажатию **OK** в диалоге таблицы переменных возвращается индекс выбранной строки.
3. Если задан признак выбора Select = FALSE, доступно редактирование и по нажатию **OK** в диалоге таблицы переменных возвращается -1.
4. Если задан признак выбора Select = FALSE, доступно редактирование и по нажатию **Присвоить значения параметрам** в диалоге таблицы переменных возвращается индекс выбранной строки.
5. Редактирование таблицы переменных доступно только для верхнего компонента.

IVariableTable – методы

AddColumn – Добавить колонку в таблицу переменных

Интерфейс...

Синтаксис Automation:

long AddColumn (LPCTSTR VarName);

Синтаксис COM:

HRESULT AddColumn (BSTR VarName, long * ColumnIndex);

Входные параметры:

VarName - имя добавляемой переменной.

Возвращаемое значение:

ColumnIndex - индекс добавленной колонки в таблице переменных - в случае успеха,
-1 - в случае неудачи.

Примечание:

1. Метод доступен только для верхнего компонента.
2. Если компонент не имел таблицы переменных, она создается.

AddRow – Добавить строку в таблицу переменных

Интерфейс...

Синтаксис Automation:

long AddRow (LPCTSTR Comment);

Синтаксис COM:

HRESULT AddRow (BSTR Comment, long * index);

Входные параметры:

Comment - комментарий к добавляемой строке.

Возвращаемое значение:

index - индекс добавленной строки в таблице переменных - в случае успеха,
-1 - в случае неудачи.

Примечание:

1. Метод доступен только для верхнего компонента.
2. Если компонент не имел таблицы переменных, она создается.

ApplyVars – Применить параметры строки к компоненту

Интерфейс...

Синтаксис Automation:

BOOL ApplyVars (VARIANT index);

Синтаксис COM:

HRESULT ApplyVars (VARIANT index, VARIANT_BOOL * result);

Входные параметры:

index - индекс строки типа long или комментарий к строке типа BSTR.

Возвращаемое значение:

TRUE - значения параметров из заданной строки успешно применены к компоненту,
FALSE - значения параметров из заданной строки не применены к компоненту.

Примечание:

Метод доступен только для верхнего компонента.

Clear – Очистить всю таблицу – удалить все строки и колонки

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear (VARIANT_BOOL * result);

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Метод доступен только для верхнего компонента.

DeleteColumn – Удалить колонку из таблицы переменных

Интерфейс...

Синтаксис Automation:

BOOL DeleteColumn (VARIANT Column);

Синтаксис COM:

HRESULT DeleteColumn (VARIANT Column, VARIANT_BOOL * result);

Входные параметры:

Column - индекс колонки типа long или имя переменной типа BSTR.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Метод доступен только для верхнего компонента.

DeleteRow – Удалить строку из таблицы переменных

Интерфейс...

Синтаксис Automation:

BOOL DeleteRow (VARIANT Row);

Синтаксис COM:

HRESULT DeleteRow (VARIANT Row, VARIANT_BOOL * result);

Входные параметры:

Row - индекс строки типа long или комментарий типа BSTR.

Возвращаемое значение:

TRUE - в случае успеха,

FALSE - в случае неудачи.

Примечание:

Метод доступен только для верхнего компонента.

FindColumnIndex – Найти индекс колонки по имени переменной

Интерфейс...

Синтаксис Automation:

long FindColumnIndex (LPCTSTR val);

Синтаксис COM:

HRESULT FindColumnIndex ([in] BSTR Val, [out, retval] long * PVal);

Входные параметры:

val - имя переменной.

Возвращаемое значение:

Индекс колонки - в случае успеха,
-1 - в случае неудачи.

Свойства

Интерфейс IProperty

Справка системы КОМПАС...

`kompas.chm::/812_87_chapter_rabota_so_svoistvami.htm`

Интерфейс «свойства».

Иерархия

IKompasAPIObject

IProperty

Описание:

Позволяет задавать параметры свойства. Номера системных свойств...

Примечание:

1. Интерфейс можно получить у менеджера свойств IPropertyMng.
2. После задания параметров требуется вызвать метод IProperty::Update.

IProperty – свойства

Comment – Комментарий

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Comment = Object.Comment	Получить свойство (*)
Object.Comment = Comment	Установить свойство (*)
Comment = Object.GetComment()	Получить свойство (**)
Object.SetComment(Comment)	Установить свойство (**)

Синтаксис COM:

Object.get_Comment(&Comment)	Получить свойство
Object.put_Comment(Comment)	Установить свойство

Свойство позволяет задавать и получать комментарий к свойству.

DataType – Тип свойства

Интерфейс...

Тип данных:ksPropertyTypeEnum

Синтаксис Automation:

DataType = Object.DataType	Получить свойство (*)
Object.DataType = DataType	Установить свойство (*)
DataType = Object.GetDataType()	Получить свойство (**)
Object.SetDataType(DataType)	Установить свойство (**)

Синтаксис COM:

Object.get_DataType(&DataType)	Получить свойство
Object.put_DataType(DataType)	Установить свойство

Свойство позволяет устанавливать и получать тип свойства.

Примечание:

Свойство можно установить только для вновь создаваемых свойств.

Id – Уникальный номер свойства

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Id = Object.Id              Получить свойство (* )
Id = Object.GetId()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Id( &Id )        Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Свойство позволяет получить уникальный номер свойства. Номера системных свойств...

IsAngular – Свойство задано в градусах

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsAngular = Object.IsAngular                Получить свойство (* )
IsAngular = Object.GetIsAngular()           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsAngular( &IsAngular )          Получить свойство
```

Примечание:

Свойство только для чтения.

IsVariable – Свойство связано с переменной

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsVariable = Object.IsVariable              Получить свойство (* )
IsVariable = Object.GetIsVariable()         Получить свойство (**)
```

Синтаксис COM:

Object.get_IsVariable(&IsVariable)	Получить свойство

Примечание:

Свойство только для чтения.

ListVal – Список значений

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ListVal = Object.ListVal	Получить свойство (*)
Object.ListVal = ListVal	Установить свойство (*)
ListVal = Object.GetListVal()	Получить свойство (**)
Object.SetListVal(ListVal)	Установить свойство (**)

Синтаксис COM:

Object.get_ListVal(&ListVal)	Получить свойство
Object.put_ListVal(ListVal)	Установить свойство

Свойство позволяет задать и получить список значений свойства.

Примечание:

Для свойства с типом данных ksPropertyDataTypeDouble значение должно иметь тип VT_ARRAY|VT_R8.

Для свойства с типом данных ksPropertyDataTypeLong значение должно иметь тип VT_ARRAY|VT_I4.

Для свойства с типом данных ksPropertyDataTypeString значение должно иметь тип VT_ARRAY|VT_BSTR.

MaxValue – Максимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

MaxValue = Object.MaxValue	Получить свойство (*)
MaxValue = Object.GetMaxValue()	Получить свойство (**)

Синтаксис COM:

Object.get_MaxValue(&MaxValue)	Получить свойство

Свойство позволяет получить максимальное значение свойства.

Примечание:

Для свойства с типом данных ksPropertyDataTypeDouble значение будет иметь тип VT_R8.

Для свойства с типом данных ksPropertyDataTypeLong значение будет иметь тип VT_I4.

Свойство только для чтения.

MinValue – Минимальное значение

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

MinValue = Object.MinValue	Получить свойство (*)
MinValue = Object.GetMinValue()	Получить свойство (**)

Синтаксис COM:

Object.get_MinValue(&MinValue)	Получить свойство

Свойство позволяет получить минимальное значение свойства.

Примечание:

Для свойства с типом данных ksPropertyDataTypeDouble значение будет иметь тип VT_R8.

Для свойства с типом данных ksPropertyDataTypeLong значение будет иметь тип VT_I4.

Свойство доступно только для чтения.

Name – Имя свойства

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Свойство позволяет устанавливать и получать имя свойства.

PropertyTypeId – Идентификатор сущности свойства

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

PropertyTypeId = Object.PropertyTypeId	Получить свойство (*)
Object.PropertyTypeId = PropertyTypeId	Установить свойство (*)
PropertyTypeId = Object.GetPropertyTypeId()	Получить свойство (**)
Object.SetPropertyTypeId(PropertyTypeId)	Установить свойство (**)

Синтаксис COM:

Object.get_PropertyTypeId (&PropertyTypeId)	Получить свойство
Object.put_PropertyTypeId (PropertyTypeId)	Установить свойство

Свойство позволяет устанавливать и получать идентификатор сущности свойства.

ReadOnly – Доступность редактирования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = Object.GetReadOnly()	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

Свойство позволяет включать и отключать доступность редактирования свойства.

SignificantDigitsCount – Количество значащих цифр после запятой

Интерфейс...

Тип данных:long

Синтаксис Automation:

SignificantDigitsCount = Object.SignificantDigitsCount	Получить свойство (*)
Object.SignificantDigitsCount = SignificantDigitsCount	Установить свойство (*)
SignificantDigitsCount = Object.GetSignificantDigitsCount()	Получить свойство (**)
Object.SetSignificantDigitsCount(SignificantDigitsCount)	Установить свойство (**)

Синтаксис COM:

Object.get_SignificantDigitsCount(&SignificantDigitsCount)	Получить свойство
Object.put_SignificantDigitsCount(SignificantDigitsCount)	Установить свойство

Свойство позволяет устанавливать и получать количество значащих цифр после запятой.

Примечание:

Установить/получить свойство можно только для ksPropertyDataTypeDouble.

SourceName – Имя источника свойства

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

SourceName = Object.SourceName	Получить свойство (*)
SourceName = Object.GetSourceName()	Получить свойство (**)

Синтаксис COM:

Object.get_SourceName(&SourceName)	Получить свойство

Свойство позволяет получить имя источника свойства.

Примечание:

Свойство только для чтения.

UnitId – Идентификатор единицы измерения свойства

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

UnitId = Object.UnitId	Получить свойство (*)
Object.UnitId = UnitId	Установить свойство (*)
UnitId = Object.GetUnitId()	Получить свойство (**)
Object.SetUnitId(UnitId)	Установить свойство (**)

Синтаксис COM:

Object.get_UnitId(&UnitId)	Получить свойство
Object.put_UnitId(UnitId)	Установить свойство

Свойство позволяет устанавливать и получать идентификатор единицы измерения свойства.

UseListVal – Использовать список значений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseListVal = Object.UseListVal	Получить свойство (*)
Object.UseListVal = UseListVal	Установить свойство (*)
UseListVal = Object.GetUseListVal()	Получить свойство (**)
Object.SetUseListVal(UseListVal)	Установить свойство (**)

Синтаксис COM:

Object.get_UseListVal(&UseListVal)	Получить свойство

Object.put_UseListVal(UseListVal) — Установить свойство

Свойство позволяет устанавливать и получать признак использования списка значений.

IProperty – методы

Delete – Удалить свойство

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL* PRes);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetValueRange – Установить новые ограничения на значение

Интерфейс...

Синтаксис Automation:

BOOL SetValueRange(VARIANT MinVal, VARIANT MaxVal);

Синтаксис COM:

HRESULT SetValueRange(VARIANT MinVal, VARIANT MaxVal, BOOL * PVal);

Входные параметры:

MinVal	- минимальное значение,
MaxVal	- максимальное значение.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Для свойства с типом данных:

- ksPropertyDataTypeDouble значение должно иметь тип VT_R8,
- ksPropertyDataTypeLong значение должно иметь тип VT_I4,
- ksPropertyDataTypeString значение должно иметь тип VT_BSTR.

ViewEdit – Выводится диалог для просмотра и редактирования свойства

Интерфейс...

Синтаксис Automation:

BOOL ViewEdit(OLE_HANDLE Parent, BOOL readOnly);

Синтаксис COM:

HRESULT ViewEdit(OLE_HANDLE Parent, BOOL ReadOnly, BOOL * Res);

Синтаксис Automation:

Parent	- дескриптор окна (может быть задан ноль - признак главного окна),
ReadOnly	- диалог только на чтение свойства.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL* PRes);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IPropertyKeeper

Интерфейс получения/редактирования значения свойств

Иерархия:

IDispatch

IPropertyKeeper

Описание:

Позволяет получать и редактировать значения свойств объектов.

Примечание:

Дополнительный интерфейс. Данный интерфейс можно получить у IPart7, IBody7, IEmbodiment, IMacroObject, IInsertionObject или IKompasDocument2D посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IPropertyKeeper – свойства

Properties – Метаданные объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Properties = Object.Properties	Получить свойство (*)
Object.Properties = Properties	Установить свойство (*)
Properties = Object.GetProperties()	Получить свойство (**)
Object.SetProperties(Properties)	Установить свойство (**)

Синтаксис COM:

Object.get_Properties(&Properties)	Получить свойство
Object.put_Properties(Properties)	Установить свойство

UniqueMetaObjectKey – Уникальный идентификатор объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

UniqueMetaObjectKey = Object.UniqueMetaObjectKey	Получить свойство(*)
UniqueMetaObjectKey = Object.GetUniqueMetaObjectKey()	Получить свойство (**)

Синтаксис COM:

Object.get_UniqueMetaObjectKey(&UniqueMetaObjectKey)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPropertyKeeper – методы

IsComplexPropertyValue – Признак комплексного значения свойства

Интерфейс...

Синтаксис Automation:

BOOL IsComplexPropertyValue(IProperty * Property);

Синтаксис COM:

HRESULT IsComplexPropertyValue(IProperty * Property, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Property	- интерфейс свойства.

GetComplexPropertyValue – Получить значение составного свойства

Интерфейс...

Синтаксис Automation:

BSTR GetComplexPropertyValue(IProperty * Property, BOOL * FromSource);

Синтаксис COM:

HRESULT GetComplexPropertyValue(IProperty * Property, BOOL * FromSource, BSTR * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Property	- интерфейс свойства.

Выходные параметры:

FromSource	- признак получения значения свойства из источника.

SetComplexPropertyValue – Установить значение составного свойства

Интерфейс...

Синтаксис Automation:

BOOL SetComplexPropertyValue(IProperty * Property, BSTR Value);

Синтаксис COM:

HRESULT SetComplexPropertyValue(IProperty * Property, BSTR Value, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Property	- интерфейс свойства,
Value	- xml-строка с описанием структуры значения составного свойства.

GetPropertyAdditionalStorage – Получить дополнительные данные, связанные со свойством

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPropertyAdditionalStorage(IProperty * Property,
BOOL Create,
BOOL * FromSource);

Синтаксис COM:

HRESULT GetPropertyAdditionalStorage(IProperty * Property,
BOOL Create,
BOOL * FromSource,
IUserDataStorage ** Result);

Возвращаемое значение:

указатель на интерфейс пользовательского хранилища с дополнительными данными для свойства.

Входные параметры:

Property	- указатель на интерфейс свойства,
Create	- создавать хранилище, если оно отсутствует.

Выходные параметры:

FromSource	- хранилище было получено из источника.

Примечание:

При первом изменении значения свойства дополнительные данные очищаются автоматически.

GetPropertyValue – Получить значение свойства

Интерфейс...

Синтаксис Automation:

BOOL GetPropertyValue(LPDISPATCH Property, VARIANT * Value, BOOL BaseUnit, BOOL * FromSource);

Синтаксис COM:

HRESULT GetPropertyValue(IProperty* Property, VARIANT * Value, BOOL BaseUnit, BOOL * FromSource, BOOL * Result);

Входные параметры:

Property	- интерфейс свойства,
Value	- значение свойства,
BaseUnit	- TRUE - значение в базовых единицах измерения (СИ), - FALSE - в настроенных единицах измерения,
FromSource	- значения было получено из источника.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetUserDataStoragesManager – Менеджер для работы с пользовательским хранилищем данных

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetUserDataStoragesManager(BOOL FromSource);

Синтаксис COM:

HRESULT GetUserDataStoragesManager(BOOL FromSource, IUserDataStoragesMng** PVal);

Входные параметры:

FromSource	- TRUE вернуть менеджер из файла источника объекта, - FALSE - вернуть менеджер из файла владельца объекта.

Возвращаемое значение:

- Указатель на интерфейс менеджера пользовательских данных IUserDataStoragesMng,

InsertHypertextReference – Добавить ссылку в свойство

Интерфейс...

Синтаксис Automation:

```
BOOL InsertHypertextReference( IProperty * Property,
                               IKompasAPIObject * Object,
                               ksHypertextTypeEnum Type,
                               BOOL Brackets,
                               long TextLineIndex,
                               long Precision,
                               double PropertyId ;
```

Синтаксис COM:

```
HRESULT InsertHypertextReference( IProperty * Property,
                                  IKompasAPIObject * Object,
                                  ksHypertextTypeEnum Type,
                                  BOOL Brackets,
                                  long TextLineIndex,
                                  long Precision,
                                  double PropertyId,
                                  BOOL * Result );
```

Входные параметры:

Property	- свойство, в которое добавляется ссылка,
Object	- объект, на свойство или текст которого ссылаемся,
Type	-тип ссылки,
Brackets	- добавлять скобки,
TextLineIndex	- индекс строки в тексте объекта,
Precision	- количество знаков после запятой,
PropertyId	- идентификатор свойства, на которое ссылаемся.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetPropertyValue – Установить значение свойства

Интерфейс...

Синтаксис Automation:

BOOL SetPropertyValue(LPDISPATCH Property, VARIANT Value, BOOL BaseUnit);

Синтаксис COM:

HRESULT SetPropertyValue(IProperty* Property, VARIANT Value, BOOL BaseUnit, BOOL * Result);

Входные параметры:

Property	- интерфейс свойства,
Value	- значение свойства,
BaseUnit	-TRUE - значение в базовых единицах измерения (СИ), - FALSE в настроенных единицах измерения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IPropertyMng

Интерфейс Менеджера свойств.

Иерархия

IDispatch

IPropertyMng

Описание:

Позволяет управлять свойствами.

Примечание:

Дополнительный интерфейс. Данный интерфейс можно получить у IApplication посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IPropertyMng – свойства

PropertyCount – Количество свойств

Интерфейс...

Тип данных: long

PropertyCount = Object.PropertyCount(Libname)	Получить свойство (*)

PropertyCount = Object.GetPropertyCount(Libname) — Получить свойство (**)

Синтаксис COM:

Object.get_PropertyCount(Libname, &PropertyCount) — Получить свойство

Входные параметры:

Libname (VARIANT) - полный путь к библиотеке на диске VT_BSTR либо указатель на документ VT_DISPATCH. Текущий документ - VT_EMPTY.

Примечание:

Свойство только для чтения.

IPropertyMng – методы

AddProperty – Добавить свойство

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddProperty(VARIANT libname, VARIANT val);

Синтаксис COM:

HRESULT AddProperty(VARIANT Libname, VARIANT Val, IProperty ** ColVal);

Входные параметры:

Libname (VARIANT) - полный путь к библиотеке на диске VT_BSTR, либо указатель на документ VT_DISPATCH. Текущий документ - VT_EMPTY,

val(VARIANT) - новое свойство VT_EMPY, создать по прототипу другого свойства - VT_DISPATCH.

Возвращаемое значение:

- Указатель на интерфейс свойства IProperty.

GetProperty – Получить свойство

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetProperty (VARIANT libname, VARIANT index);

Синтаксис COM:

HRESULT GetProperty(VARIANT Libname, VARIANT Index, IProperty ** Res);

Входные параметры:

Libname (VARIANT)	- полный путь к библиотеке на диске VT_BSTR, либо указатель на документ VT_DISPATCH. Текущий документ - VT_EMPTY,
index(VARIANT)	- индекс в хранилище VT_I4, либо по идентификатору самого свойства - VT_R8.

Возвращаемое значение:

- Указатель на интерфейс свойства IProperty.

Номера системных свойств...

GetProperties – Получить массив свойств

Интерфейс...

Синтаксис Automation:

VARIANT GetProperties (VARIANT libname);

Синтаксис COM:

HRESULT GetProperties(VARIANT Libname, VARIANT * Res);

Входные параметры:

Libname (VARIANT)	- полный путь к библиотеке на диске VT_BSTR, либо указатель на документ VT_DISPATCH. Текущий документ - VT_EMPTY.

Возвращаемое значение:

- Массив библиотек VARIANT типа VT_ARRAY | VT_DISPATCH.

Примечание:

Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

GetReport – Получить интерфейс таблицы отчета по имени файла или указателю на документ

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetReport(VARIANT Document, ksReportTypeEnum Type);

Синтаксис COM:

HRESULT GetReport(VARIANT Document, ksReportTypeEnum Type, IReport ** Result);

Возвращаемое значение:

- Указатель на интерфейс отчета IReport.

Входные параметры:

Document - имя файла или указатель на документ.

Выходные параметры:

Type - тип отчета.

RemoveProperty – Удалить свойство

Интерфейс...

Синтаксис Automation:

BOOL RemoveProperty (VARIANT libname, VARIANT val);

Синтаксис COM:

HRESULT RemoveProperty(VARIANT Libname, VARIANT Val, BOOL * ColVal);

Входные параметры:

Libname (VARIANT) - полный путь к библиотеке на диске VT_BSTR, либо указатель на документ VT_DISPATCH. Текущий документ - VT_EMPTY,

val(VARIANT) - по интерфейсу свойства - VT_DISPATCH, по индексу в массиве - VT_I4, по идентификатору свойства VT_R8.

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Хранилища

Интерфейс IUserDataStorage

Интерфейс пользовательского хранилища.

Иерархия:

IKompasAPIObject

IUserDataStorage

Описание:

Хранилище представляет собой именованную «закладку» в коллекции. Для хранилища могут задаваться версии. Хранилище может быть защищено от изменений паролем. Парольная защита не препятствует просмотру содержания хранилища. Данные в хранилище представлены в виде таблицы, состоящей из двух столбцов – данных различных типов и комментария.

Примечание:

Интерфейс можно получить у коллекции хранилищ, используя свойство IUserDataStorages::Item.

IUserDataStorage – свойства

Name – Имя хранилища

Интерфейс...

Синтаксис Automation:

Тип данных: LPSTR

Name = StorageObj.Name(Pass)	Получить свойство (*)
StorageObj.Name(Pass) = Name	Установить свойство (*)
Name = StorageObj.GetName(Pass)	Получить свойство (**)
StorageObj.SetName (Name, Pass)	Установить свойство (**)

Входные параметры:

Pass (LPSTR) - пароль хранилища.

Синтаксис COM:

Тип данных:BSTR.

StorageObj->get_Name (Pass, &Name)	Получить свойство
StorageObj->put_Name (Pass, Name)	Установить свойство

Входные параметры:

Pass (BSTR) - пароль хранилища.

Примечание:

Если хранилище защищено паролем, то изменить свойство можно, только указав правильный пароль.

Version – Версия хранилища

Интерфейс...

Синтаксис Automation:

Тип данных: long

```
Version = StorageObj.Version(Pass)          Получить свойство (* )
StorageObj.Version(Pass) = Version          Установить свойство (* )
Version = StorageObj.GetVersion(Pass)       Получить свойство (**)
StorageObj.SetVersion (Version, Pass)       Установить свойство (**)
```

Входные параметры:

Pass (LPSTR) - пароль хранилища.

Синтаксис COM:

Тип данных: long

```
StorageObj->get_Version (Pass, &Version)    Получить свойство
StorageObj->put_Version (Pass, Version)     Установить свойство
```

Входные параметры:

Pass (BSTR) - пароль хранилища.

Примечание:

Если хранилище защищено паролем, то изменить свойство можно, только указав пароль.

IUserDataStorage – методы

AddObject – Добавить объект и комментарий в пользовательское хранилище

Интерфейс...

Синтаксис Automation:

```
long AddObject( LPCTSTR pass,
const VARIANT & object,
LPCTSTR comment );
```

Синтаксис COM:

HRESULT AddObject([in] BSTR Pass,
[in] VARIANT Object,
[in] BSTR Comment,
[out, retval] long* pVal);

Входные параметры:

Pass - пароль,
Object - указатель на объект,
comment - комментарий.

Возвращаемое значение:

- индекс объекта,

Примечание:

1. Объект может быть variant'ом с типом VT_INT, VT_I2, VT_I4, VT_R4, VT_R8, VT_DATE, VT_UI1, VT_UINT, VT_BOOL, VT_BSTR, VT_DISPATCH, а также VT_ARRAY с этими типами данных. Индекс массива начинается с нуля.
2. Если задан интерфейс VT_DISPATCH и массив объектов VT_ARRAY | VT_DISPATCH, то под объектом подразумевается указатель на IDispatch от объектов API. Они также будут сохранены вместе с документом и при следующем запросе найдены.
3. Индексация массивов должна начинаться с нуля.
4. Если хранилище защищено паролем от изменений, то метод будет выполнен только после указания правильного пароля.
5. Под объектом API следует понимать объекты:

   для 3D хранилища - наследник от IModelObject,

   для 2D хранилища - наследник от IDrawingObject (кроме ILayer).

Clear – Очистить хранилище, удалив все объекты

Интерфейс...

Синтаксис Automation:

BOOL Clear (LPCTSTR Pass);

Синтаксис COM:

HRESULT Clear([in] BSTR Pass,
[out, retval] VARIANT_BOOL* Result);

Входные параметры:

Pass - пароль.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Если хранилище защищено паролем от изменений, то метод будет выполнен только после указания правильного пароля.

Delete – Удалить объект, заданный по индексу или по комментарию

Интерфейс...

Синтаксис Automation:

BOOL Delete(LPCTSTR pass,
const VARIANT & index);

Синтаксис COM:

HRESULT Delete([in] BSTR Pass,
[in] VARIANT Index,
[out, retval] VARIANT_BOOL* PVal);

Входные параметры:

Pass	- пароль,
Index	- индекс объекта или комментарий.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. В качестве индекса можно задать число или комментарий к объекту.
2. Если хранилище защищено паролем от изменений, то метод будет выполнен только после указания правильного пароля.

GetObject – Получить объект пользовательского хранилища и комментарий к нему по комментарию или индексу

Интерфейс...

Синтаксис Automation:

BSTR GetObject(const VARIANT & index, VARIANT * object, long * numb);

Синтаксис COM:

HRESULT GetObject([in] VARIANT Index,
[out] VARIANT * Object,
[out] long* numb,
[out, retval] BSTR * Comment);

Входные параметры:

Index - индекс объекта или комментарий.

Выходные параметры:

Object - указатель на объект хранилища,
numb - указатель на номер объекта.

Возвращаемое значение:

Comment (BSTR) - комментарий.

Примечание:

1. В качестве индекса можно задать число или комментарий к объекту.
2. Объект может быть variant'ом с типом VT_INT, VT_I2, VT_I4, VT_R4, VT_R8, VT_DATE, VT_UI1, VT_UINT, VT_BOOL, VT_BSTR, VT_DISPATCH, а также VT_ARRAY с этими типами данных.
3. Если задан интерфейс VT_DISPATCH и массив объектов VT_ARRAY | VT_DISPATCH, то под объектом подразумевается указатель на IDispatch от объектов API. Они также будут сохранены вместе с документом и при следующем запросе найдены.

SetObject – Установить объект пользовательского хранилища и комментарий к нему по комментарию или индексу

Интерфейс...

Синтаксис Automation:

```
long SetObject( LPCTSTR pass,
const VARIANT index,
const VARIANT & object,
LPCTSTR comment );
```

Синтаксис COM:

```
HRESULT SetObject( [in] BSTR Pass,
[in] VARIANT Index,
[in] VARIANT Object,
[in] BSTR Comment,
[out, retval] long* pVal);
```

Входные параметры:

Pass - пароль,
Index - индекс объекта или комментарий,
Object - указатель на объект,
comment - комментарий.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. В качестве индекса можно задать число или комментарий к объекту.
2. Объект может быть variant'ом с типом VT_INT, VT_I2, VT_I4, VT_R4, VT_R8, VT_DATE, VT_UI1, VT_UINT, VT_BOOL, VT_BSTR, VT_DISPATCH, а также VT_ARRAY с этими типами данных.
3. Если задан интерфейс VT_DISPATCH и массив объектов VT_ARRAY | VT_DISPATCH, то под объектом подразумевается указатель на IDispatch от объектов API. Они также будут сохранены вместе с документом и при следующем запросе найдены.
4. Если хранилище защищено паролем от изменений, то метод будет выполнен только после указания правильного пароля.

SetPassword – Установить пароль на изменение данных в хранилище

Интерфейс...

Синтаксис Automation:

BOOL SetPassword(LPCTSTR oldPass, LPCTSTR newPass);

Синтаксис COM:

HRESULT SetPassword([in] BSTR OldPass,
[in] BSTR NewPass,
[out, retval] VARIANT_BOOL* Result);

Входные параметры:

OldPass - старый пароль,
NewPass - новый пароль.

Примечание:

1. Если хранилище защищено паролем от изменений, то изменить пароль можно, только указав текущий пароль.
2. По умолчанию пароль представляет собой пустую строку или нулевой указатель.

Интерфейс IUserDataStorages

Интерфейс коллекции пользовательских хранилищ.

Иерархия:

IKompasAPIObject
    IKompasCollection
        IUserDataStorages

Описание:

Коллекция привязана к одному объекту в документе или непосредственно к документу (объект NULL).

Элементы коллекции – хранилища представляют собой «именованные закладки», каждая из которых содержит данные различных типов в табличной форме.

Примечание:

Интерфейс можно получить у Менеджера хранилищ, используя метод IUserDataStoragesMng::Item.

IUserDataStorages – свойства

Item – Хранилище, заданное по имени или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IUserDataStorage хранилища

Синтаксис Automation:

```
StorageObj = iObject.Item (Index)          Получить свойство (* )
StorageObj = iObject.GetItem (Index)       Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Item (Index, &StorageObj)     Получить свойство
```

Входные параметры:

Index	- индекс хранилища.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса можно задать число или имя хранилища (VT_BSTR).

IUserDataStorages – методы

Add – Создать новое хранилище и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(LPCTSTR storageName);

Синтаксис COM:

HRESULT Add([in] BSTR StorageName,
[out, retval] IUserDataStorage ** Result);

Входные параметры:

storageName - имя создаваемого хранилища.

Delete – Удалить хранилище из коллекции по имени или по индексу

Интерфейс...

Синтаксис Automation:

BOOL Delete (LPCTSTR password, const VARIANT index);

Синтаксис COM:

HRESULT Delete([in] BSTR password,
[in] VARIANT Index,
[out, retval] VARIANT_BOOL* PVal);

Входные параметры:

Index - индекс или имя хранилища,
password - пароль на хранилище.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. В качестве индекса можно задать число или имя хранилища (VT_BSTR).
2. Если хранилище защищено паролем, то метод работает только в случае указания правильного пароля.

Интерфейс IUserDataStoragesMng

Интерфейс Менеджера пользовательских хранилищ.

Иерархия:

IKompasAPIObject
    IKompasCollection
        IUserDataStoragesMng

Описание:

Объект документа или непосредственно документ (объект NULL) может быть связан с коллекцией хранилищ. Каждое хранилище представляет собой структурированный набор данных простых типов (Variant) или также указателей на объекты документа. Менеджер пользовательских хранилищ позволяет управлять пользовательскими хранилищами объектов и их коллекциями. Хранилище может быть использовано для графических документов и документов–моделей.

Примечание:

Интерфейс можно получить у документа, используя метод IKompasDocument::UserDataStoragesMng.

IUserDataStoragesMng – свойства

Item – Объект, заданный по ссылке на объект или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IUserDataStorages коллекции объектов хранилища.

Синтаксис Automation:

StorageObj = iObject.Item (Index) Получить свойство (*)
StorageObj = iObject.GetItem (Index) Получить свойство (**)

Синтаксис COM:

iObject->get_Item (Index, &StorageObj) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса можно задать число или объект, к которому привязано хранилище (тип VT_DISPATCH).

IUserDataStoragesMng – методы

Add – Создать новое хранилища по объекту и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(LPDISPATCH Object);

Синтаксис COM:

HRESULT Add([in] IKompasAPIObject* Object,
[out, retval] IUserDataStorages ** Result);

Входные параметры:

Object - Указатель на интерфейс объекта IKompasAPIObject.

Выходные параметры:

- Указатель на интерфейс коллекции объектов хранилища IUserDataStorages.

Примечание:

1. Метод позволяет добавить массив хранилищ по объекту в коллекцию. Если объект NULL, то хранилище добавляется к документу.
2. Если в коллекции было создано хранилище по выбранному объекту, то метод сработает как GetItem.

Clear– очистить менеджер от всех коллекций

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Delete – Удалить коллекцию по ссылке на объект или по индексу

Интерфейс...

Синтаксис Automation:

BOOL Delete (const VARIANT & index);

Синтаксис COM:

HRESULT Delete([in] VARIANT Index,
[out, retval] VARIANT_BOOL* pVal);

Входные параметры:

Index	- Индекс или указатель на интерфейс коллекции объектов хранилища IUserDataStorages.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

В качестве индекса можно задать число или объект, к которому привязано хранилище (тип VT_DISPATCH)

Интерфейс IUserMetadataManager

Менеджер пользовательских метаданных.

Иерархия:

IDispatch

IUserMetadataManager

Интерфейс можно получить методом QueryInterface у интерфейса IKompasDocument

IUserMetadataManager – свойства

StorageInfo – Получить данные из StorageInfo

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

StorageInfo = Object.StorageInfo(ApplicationIID, ParameterName)	Получить свойство (*)
Object.StorageInfo(ApplicationIID, ParameterName) = Visible	Установить свойство (*)
StorageInfo= Object.GetStorageInfo(ApplicationIID, ParameterName)	Получить свойство (**)
Object.SetStorageInfo(ApplicationIID, ParameterName, StorageInfo)	Установить свойство (**)

Синтаксис COM:

Object.get_StorageInfo(ApplicationIID, ParameterName, &StorageInfo)	Получить свойство
Object.put_StorageInfo(ApplicationIID, ParameterName, StorageInfo)	Установить свойство

Входные параметры:

BSTR	ApplicationIID	- GUID приложения,
BSTR	ParameterName	- имя параметр.

Примечание:

Свойство позволяет устанавливать и получать имя или версию хранилища:

- для получения имени ParameterName - "storageInfo.appName",
- для получения версии - "storageInfo.appVersion".

IUserMetadataManager – методы

AddFile – Добавить файл в хранилище

Интерфейс...

Синтаксис Automation:

BOOL AddFile(BSTR ApplicationIID, BSTR SrcFileName, BSTR DestFileName, BOOL AllowReplacement, BOOL Compress);

Синтаксис COM:

HRESULT AddFile(BSTR ApplicationIID, BSTR SrcFileName, BSTR DestFileName, BOOL AllowReplacement, BOOL Compress, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Входные параметры:

ApplicationIID	- GUID приложения,
SrcFileName	- имя файла в хранилище,
DestFileName	- имя файла на диске,
AllowReplacement	- перезаписывать,
Compress	- добавить со сжатием.

CreateStorage – Создать хранилище файлов приложения

Интерфейс...

Синтаксис Automation:

BOOL CreateStorage(BSTR ApplicationIID, BSTR ApplicationDescription, BSTR Version);

Синтаксис COM:

HRESULT CreateStorage(BSTR ApplicationIID, BSTR ApplicationDescription, BSTR Version, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Входные параметры:

ApplicationIID	- GUID приложения,
ApplicationDescription	- имя приложения,
DestFileName	- версия хранилища.

DeleteFile – Удалить файл из хранилища

Интерфейс...

Синтаксис Automation:

BOOL DeleteFile(BSTR ApplicationIID, BSTR StorageFileName);

Синтаксис COM:

HRESULT DeleteFile(BSTR ApplicationIID, BSTR StorageFileName, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Входные параметры:

ApplicationIID	- GUID приложения,
StorageFileName	- имя удаляемого файла.

DeleteStorage – Удалить хранилище файлов приложения

Интерфейс...

Синтаксис Automation:

BOOL DeleteStorage(BSTR ApplicationIID);

Синтаксис COM:

HRESULT DeleteStorage(BSTR ApplicationIID, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Входные параметры:

ApplicationIID	- GUID приложения.

ExistStorage – Проверить наличие хранилища данных приложения

Интерфейс...

Синтаксис Automation:

BOOL ExistStorage(BSTR ApplicationIID);

Синтаксис COM:

HRESULT ExistStorage(BSTR ApplicationIID, BOOL * Result);

Возвращаемое значение:

TRUE	-хранилище существует,
FALSE	- хранилище не существует.

Входные параметры:

ApplicationIID - GUID приложения.

ExtractFile – Извлечь файл из хранилища

Интерфейс...

Синтаксис Automation:

BOOL ExtractFile(BSTR ApplicationIID, BSTR SrcFileName, BSTR DestFileName, BOOL AllowReplacement);

Синтаксис COM:

HRESULT ExtractFile(BSTR ApplicationIID, BSTR SrcFileName, BSTR DestFileName, BOOL AllowReplacement, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Входные параметры:

ApplicationIID - GUID приложения,
SrcFileName - имя файла в хранилище,
DestFileName - имя файла на диске,
AllowReplacement - перезаписывать.

GetAllFilenames – Получить список файлов приложения

Интерфейс...

Синтаксис Automation:

VARIANT GetAllFilenames(BSTR ApplicationIID);

Синтаксис COM:

HRESULT GetAllFilenames(BSTR ApplicationIID, VARIANT * Result);

Возвращаемое значение:

-массив VT_ARRAY | VT_BSTR имен файлов в хранилище.

Входные параметры:

ApplicationIID - GUID приложения.

IsExistFile – Проверить наличие файла в хранилище

Интерфейс...

Синтаксис Automation:

BOOL IsExistFile(BSTR ApplicationIID, BSTR StorageFileName);

Синтаксис COM:

HRESULT IsExistFile(BSTR ApplicationIID, BSTR StorageFileName, BOOL * Result);

Возвращаемое значение:

TRUE	- файл существует,
FALSE	- файл не существует.

Входные параметры:

ApplicationIID	- GUID приложения,
StorageFileName	- имя удаляемого файла.

Документ 2D

Виды и слои

Интерфейс IViewsAndLayersManager

События...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Менеджер слоев и видов графического документа.

Иерархия:

IKompasAPIObject

IViewsAndLayersManager

Примечание:

1. Данный интерфейс предназначен для работы с видами, слоями, группами слоев.
2. Интерфейс можно получить при помощи свойства IKompasDocument2D::ViewsAndLayersManager от интерфейса 2D-документа IKompasDocument2D.

IViewsAndLayersManager – свойства

LayerGroups – Возвращает коллекцию групп слоев

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/396_46_2_Menedzher_dokumenta.htm

Тип данных: ILayerGroups

Синтаксис Automation:

LayerGroups = iObject.LayerGroups	Получить свойство (*)
LayerGroups = iObject.GetLayerGroups()	Получить свойство (**)

Синтаксис COM:

iObject->get_LayerGroups (LayerGroups)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Views – Возвращает коллекцию видов

Интерфейс...

Тип данных: IViews

Синтаксис Automation:

Views = iObject.Views	Получить свойство (*)
Views = iObject.GetViews()	Получить свойство (**)

Синтаксис COM:

iObject->get_Views (Views)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ILayerGroup

Справка системы КОМПАС...

`kompas.chm::/405_46_9_1_Nabory_sloev.htm`

Интерфейс группы слоев.

Иерархия:

IKompasAPIObject

ILayerGroup

Примечание:

Данный интерфейс можно получить у коллекции групп слоев ILayerGroups с помощью свойства ILayerGroups::Item, метода ILayerGroups::Add либо от интерфейса другой группы слоев с помощью свойства ILayerGroup::OwnerGroup.

Группы являются именованными объектами.

По умолчанию при создании группы ей присваивается имя, состоящее из слова "Группа" (для статической группы и группы свойств) или "Фильтр" (для динамической группы) и порядкового номера. Это имя может быть изменено.

Группы слоев сохраняются в документе.

Группа слоев может принадлежать:

1. Виду - в группе объединяются слои, принадлежащие виду.
2. Документу - в группе объединяются слои с разных видов.

Группа может быть:

- статической - Группа слоев или Группа свойств слоев;
- динамической - Фильтр.

Группа слоев группа объединяет слои. Для создания статической группы используется метод ILayerGroups::Add, в который передается параметр dynamic равный FALSE. Работа

с группой слоев доступна, если в документе установлен режим **Группировать слои**, т.е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayers.

Фильтр включает в себя слои, отфильтрованные согласно заданным условиям. См. ILayerFilterConditions и ILayerFilterCondition.

Признаком того, что группа является динамической, является возможность получения у нее коллекции условий ILayerFilterConditions.

Для создания динамической группы используется метод ILayerGroups::Add, в который передается параметр dynamic равный TRUE. Работа с группой слоев доступна, если в документе установлен режим **Группировать слои**, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayers.

Группа свойств слоев, кроме списка слоев, содержит массив свойств, устанавливаемых данным слоям. Для создания группы используется метод ILayerGroups::Add, в котором параметр dynamic не учитывается. Для передачи параметров слоям нужно установить свойство ILayerGroup::Current равным TRUE

Работа с **Группой свойств слоев** доступна, если в документе установлен режим **Группировать свойства слоев**, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayersCharacteristics.

Добавление слоя в документ не приводит к автоматическому добавлению этого слоя в статическую группу или в группу свойств.

При удалении слоя из группы он не удаляется из документа.

При удалении слоя из документа он автоматически удаляется из группы.

Изменение способа группировки в документе приводит к необратимым изменениям созданных групп.

Если устанавливается режим группировки слоев, то Группы свойств слоев превращаются в Группы слоев.

Если устанавливается режим группировки свойств слоев, то Группы слоев и Фильтры превращаются в Группы свойств.

ILayerGroup – свойства

Current – Группа является текущей (состояния слоев переданы в модель)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Current = iObject.Current                Получить свойство (* )
iObject.Current = Current                Установить свойство (* )
Current = iObject.GetCurrent()           Получить свойство (**)
iObject.SetCurrent (Current)             Установить свойство (**)
```

Синтаксис COM:

iObject->get_Current (&Current)	Получить свойство
iObject->put_Current (Current)	Установить свойство

Примечание:

Свойство используется, если в документе установлен режим **Группировать свойства слоев**, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayersCharacteristics.

Layers – Коллекция слоев группы

Интерфейс...

Тип данных: указатель на интерфейс ILayers

Синтаксис Automation:

Layers = iObject.Layers	Получить свойство (*)
Layers = iObject.GetLayers()	Получить свойство (**)

Синтаксис COM:

iObject->get_Layers (&Layers)	Получить свойство

LayerFilterConditions – Коллекция условий фильтрации слоев

Интерфейс...

Тип данных: указатель на интерфейс коллекции условий фильтрации ILayerFilterConditions

Синтаксис Automation:

LayerFilterConditions = iObject.LayerFilterConditions	Получить свойство (*)
LayerFilterConditions = iObject.GetLayerFilterConditions()	Получить свойство (**)

Синтаксис COM:

iObject->get_LayerFilterConditions (&LayerFilterConditions)	Получить свойство

Примечание:

Коллекцию условий можно получить только для динамической группы слоев.

LayerGroups – Коллекция групп слоев

Интерфейс...

Тип данных: указатель на интерфейс ILayerGroups

Синтаксис Automation:

```
LayerGroups = iObject.LayerGroups          Получить свойство (* )
LayerGroups = iObject.GetLayerGroups()     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_LayerGroups                   Получить свойство
(&LayerGroups)
```

Name – Имя группы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = iObject.Name                        Получить свойство (* )
iObject.Name = Name                        Установить свойство (* )
Name = iObject.GetName()                   Получить свойство (**)
iObject.SetName (Name)                     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name (&Name)                  Получить свойство
iObject->put_Name (Name)                   Установить свойство
```

OwnerGroup – Владелец группы – другая группа

Интерфейс...

Тип данных: указатель на интерфейс группы ILayerGroup

Синтаксис Automation:

```
OwnerGroup = iObject.OwnerGroup            Получить свойство (* )
OwnerGroup = iObject.GetOwnerGroup()       Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_OwnerGroup                    Получить свойство,
(&OwnerGroup)
```

Примечание:

1. Позволяет получить владельца группы.
2. Свойство доступно только для чтения.

OwnerView – Владелец группы – вид

Интерфейс...

Тип данных: указатель на интерфейс вида IView

Синтаксис Automation:

```
OwnerView = iObject.OwnerView                Получить свойство (* )
iObject.OwnerView = OwnerView                Установить свойство (* )
OwnerView = iObject.GetOwnerView()           Получить свойство (**)
iObject.SetOwnerView (OwnerView)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_OwnerView (&OwnerView)          Получить свойство
iObject->put_OwnerView (OwnerView)           Установить свойство
```

Примечание:

1. Свойство позволяет отсоединить группу от вида.
2. Чтобы установить свойство отсоединения группы от вида, необходимо передать параметр с нулевым значением.

UniqueId – Уникальный идентификатор группы слоев

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
UniqueId = iObject.UniqueId                  Получить свойство (* )
UniqueId = iObject.GetUniqueId()             Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_UniqueId (&UniqueId)            Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. После создания группы свойство не изменяется и хранится в чертеже.

ILayerGroup – методы

Delete – Удалить группу слоев

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL * pVal);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

При удалении группы слоев слои, входящие в нее, из документа не удаляются. Для удаления слоя используется метод IDrawingObject::Delete.

GetLayerStates – Получить состояния слоя группы по индексу, имени или ссылке

Интерфейс...

Синтаксис Automation:

BOOL GetLayerStates ([in] VARIANT LayerIndex,
[out] BOOL* visible,
[out] BOOL* background,
[out] BOOL* curent,
[out] long* color);

Синтаксис COM:

HRESULT GetLayerStates ([in] VARIANT LayerIndex,
[out] VARIANT_BOOL* visible,
[out] VARIANT_BOOL* background,
[out] VARIANT_BOOL* curent,
[out] long* color,
[out, retval] VARIANT_BOOL* pRes);

Входные параметры:

LayerIndex	- индекс слоя.

Выходные параметры:

visible	- свойство - видимый слой,
background	- свойство - фоновый слой,

current	- свойство - текущий слой,
color	- цвет слоя.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Если в документе установлен режим **Группировать свойства слоев**, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayersCharacteristics, то метод позволяет получить свойства слоя в группе. При этом значения свойств могут отличаться от установленных в текущий момент у слоя в документе. Для передачи свойств слою из группы нужно сделать группу текущей, то есть установить значение ILayerGroup::Current равным TRUE.
2. Если в документе установлен режим Группировать слои, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayers, то метод используется для получения свойств конкретного слоя (LayerIndex - индекс слоя).

SetLayerStates – Изменить состояния слоя группы по индексу или имени

Интерфейс...

Синтаксис Automation:

BOOL SetLayerStates ([in] VARIANT LayerIndex,
[in] BOOL visible,
[in]BOOL background,
[in] BOOL* curent,
[in] long color);

Синтаксис COM:

HRESULT SetLayerStates ([in] VARIANT LayerIndex,
[in] VARIANT_BOOL visible,
[in] VARIANT_BOOL background,
[in] VARIANT_BOOL curent,
[in] long color,
[out, retval] VARIANT_BOOL* pRes);

Входные параметры:

LayerIndex	- индекс слоя,
visible	- свойство - видимый слой,
background	- свойство - фоновый слой,
current	- свойство - текущий слой,
color	- цвет слоя.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Если в документе установлен режим **Группировать свойства слоев**, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayersCharacteristics, то метод позволяет установить свойства слоя в группе. При этом параметры слоя в документе не меняются. При этом значения свойств могут отличаться от установленных в текущий момент у слоя в документе. Для передачи свойств слою из группы нужно сделать группу текущей, то есть установить значение ILayerGroup::Current равным TRUE.
2. Если в документе установлен режим Группировать слои, т. е. свойство IDocument2DSettings::LayersGroupWay имеет значение wgLayers, то метод используется для установки свойств конкретного слоя (LayerIndex - индекс слоя). Для установки одинаковых состояний и цвета всем слоям, входящим в группу, нужно передать LayerIndex = -1.

Интерфейс ILayerFilterCondition

Справка системы КОМПАС...

`kompas.chm::/405_46_9_1_Nabory_sloev.htm`

Интерфейс условия фильтрации слоев.

Иерархия:

IKompasAPIObject

ILayerFilterCondition

Описание:

Условия фильтрации используются для формирования состава динамической группы слоев.

Примечание:

Данный интерфейс можно получить через коллекцию условий фильтрации слоев ILayerFilterConditions с помощью свойства ILayerFilterConditions::Item.

ILayerFilterCondition – свойства

Background – Состояние слоя для фильтрации – фоновый

Интерфейс...

Тип данных: из перечисления FilterConditionStateEnum.

Синтаксис Automation:

Background = iObject.Background	Получить свойство (*)
iObject.Background = Background	Установить свойство (*)
Background = iObject.GetBackground()	Получить свойство (**)
iObject.SetBackground (Background)	Установить свойство (**)

Синтаксис COM:

iObject->get_Background (&Background)	Получить свойство
iObject->put_Background (Background)	Установить свойство

Примечание:

После установки нового значения данного свойства состав группы будет автоматически изменен.

Color – Цвет слоя для фильтрации

Интерфейс...

Тип данных: long

Синтаксис Automation:

Color = iObject.Color	Получить свойство (*)
iObject.Color = Color	Установить свойство (*)
Color = iObject.GetColor()	Получить свойство (**)
iObject.SetColor (Color)	Установить свойство (**)

Синтаксис COM:

iObject->get_Color (&Color)	Получить свойство
iObject->put_Color (Color)	Установить свойство

Примечание:

После установки нового значения данного свойства состав группы будет автоматически изменен.

Comment – Содержание комментария для фильтрации

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Comment = iObject.Comment	Получить свойство (*)
iObject.Comment = Comment	Установить свойство (*)
Comment = iObject.GetComment()	Получить свойство (**)
iObject.SetComment (Comment)	Установить свойство (**)

Синтаксис COM:

iObject->get_Comment (&Comment)	Получить свойство

iObject->put_Comment (Comment)	Установить свойство

Примечание:

1. При задании условия фильтрации по номеру можно использовать маски. В масках знак «*» заменяет любое количество любых символов. Знак «?» заменяет один любой символ.
2. После установки нового значения данного свойства состав группы будет автоматически изменен.

HaveObjects – Состояние слоя для фильтрации – с объектами или пустой

Интерфейс...

Тип данных: из перечисления FilterConditionStateEnum

Синтаксис Automation:

HaveObjects = iObject.HaveObjects	Получить свойство (*)
iObject.HaveObjects = HaveObjects	Установить свойство (*)
HaveObjects = iObject.GetHaveObjects()	Получить свойство (**)
iObject.SetHaveObjects (HaveObjects)	Установить свойство (**)

Синтаксис COM:

iObject->get_HaveObjects (&HaveObjects)	Получить свойство
iObject->put_HaveObjects (HaveObjects)	Установить свойство

Примечание:

После установки нового значения данного свойства состав группы будет автоматически изменен.

Name – Имя слоя для фильтрации

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName()	Получить свойство (**)
iObject.SetName (Name)	Установить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство
iObject->put_Name (Name)	Установить свойство

Примечание:

1. При задании условия фильтрации по номеру можно использовать маски. В масках знак «*» заменяет любое количество любых символов. Знак «?» заменяет один любой символ.
2. После установки нового значения данного свойства состав группы будет автоматически изменен.

Number – Номер слоя для фильтрации

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
iObject.Number = Number	Установить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)
iObject.SetNumber (Number)	Установить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство
iObject->put_Number (Number)	Установить свойство

Примечание:

1. При задании условия фильтрации по номеру можно использовать маски. В масках знак «*» заменяет любое количество любых символов. Знак «?» заменяет один любой символ.
2. После установки нового значения данного свойства состав группы будет автоматически изменен.

Projected – Признак проецирования для слоя 3D

Интерфейс...

Тип данных: из перечисления FilterConditionStateEnum

Синтаксис Automation:

Projected = Object.Projected	Получить свойство (*)
Object.Projected = Projected	Установить свойство (*)
Projected = Object.GetProjected()	Получить свойство (**)
Object.SetProjected(Projected)	Установить свойство (**)

Синтаксис COM:

Object.get_Projected(&Projected)	Получить свойство
Object.put_Projected(Projected)	Установить свойство

Visible – Состояние слоя для фильтрации – видимый или погашенный

Интерфейс...

Тип данных: из перечисления FilterConditionStateEnum

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство (**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

После установки нового значения данного свойства состав группы будет автоматически изменен.

Интерфейс ILayerGroups

Справка системы КОМПАС...

`KOMPAS.chm::/405_46_9_1_Nabory_sloev.htm`

Интерфейс коллекции групп слоев.

Иерархия:

IKompasAPIObject

IKompasCollection

ILayerGroups

Описание:

Коллекция объединяет группы слоев ILayerGroup.

Примечание:

Данный интерфейс может быть получен следующими способами:

- от интерфейса менеджера видов и слоёв IViewsAndLayersManager с помощью свойства IViewsAndLayersManager::LayerGroups,
- от интерфейса группы слоев ILayerGroup с помощью свойства ILayerGroup::LayerGroups.

ILayerGroups – свойства

Item – Группа, заданная по индексу, имени или уникальному идентификатору

Интерфейс...

Тип данных: указатель на интерфейс ILayerGroup

Синтаксис Automation:

```
Item = iObject.Item                    Получить свойство (* )
Item = iObject.GetItem()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Item (&Item)              Получить свойство
```

Входные параметры:

```
index          - индекс объекта в коллекции (тип VARIANT).
```

Примечание:

Свойство доступно только для чтения.

В качестве индекса может использоваться индекс элемента в массиве, имя или уникальный идентификатор ILayerGroup::UniqueId. Для получения группы по идентификатору нужно передать вещественное число, установив тип значения VT_R8.

ILayerGroups – методы

Add – Создать группу

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add([in] LPDISPATCH owner, [in] VARIANT_BOOL dinamic);

Синтаксис Automation:

HRESULT Add([in] IView* owner, [in] VARIANT_BOOL dinamic, [out, retval] ILayerGroup** Result);

Входные параметры:

```
owner      - вид - владелец группы,
dinamic    - признак динамической группы.
```

Возвращаемое значение:

Указатель на интерфейс группы слоев ILayerGroup

Примечание:

Группа добавляется в коллекцию.

Attach – Добавить существующую группу

Интерфейс...

Синтаксис Automation:

BOOL Attach (LPDISPATCH pVal);

Синтаксис Automation:

HRESULT Attach ([in] ILayerGroup * pVal, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

pVal - Указатель на интерфейс группы слоев ILayerGroup.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Интерфейс группы добавляется в коллекцию.

Detach – Отсоединить группу слоев

Интерфейс...

Синтаксис Automation:

BOOL Detach (LPDISPATCH pVal);

Синтаксис COM:

HRESULT Detach ([in] ILayerGroup * pVal, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

pVal - Указатель на интерфейс группы слоев ILayerGroup.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Интерфейс группы слоев отсоединяется от коллекции, группа при этом из документа не удаляется.

Интерфейс ILayerFilterConditions

Справка системы КОМПАС...

KOMPAS.chm::/408_46_9_3_Operacii_s_naborami_.htm

Интерфейс коллекции условий фильтрации слоев.

Иерархия:

IKompasAPIObject

IKompasCollection

ILayerFilterConditions

Описание:

1. Условия фильтрации используются для формирования состава динамической группы слоев.
2. Слой автоматически будет включен в состав группы если его свойства удовлетворяют хотя бы одному из условий.

Примечание:

Данный интерфейс можно получить через группу слоев ILayerGroup с помощью свойства ILayerGroup::LayerFilterConditions.

ILayerFilterConditions – свойства

Item – Условие фильтрации слоев, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс ILayerFilterCondition

Синтаксис Automation:

LayerFilterCondition = iObject.Item (Index)	Получить свойство (*)
LayerFilterCondition = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &LayerFilterCondition)	(Index,	Получить свойство

Входные параметры:

index	- индекс в коллекции; тип VARIANT.

Примечание:

Свойство доступно только для чтения.

ILayerFilterConditions – методы

Add – Создать условие фильтрации слоев

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add ([out, retval] ILayerFilterCondition** Result);

Возвращаемое значение:

Указатель на интерфейс ILayerFilterCondition

Примечание:

После вызова этого метода условие фильтрации слоев будет добавлено в коллекцию.

Состав группы изменится в соответствии с новыми условиями.

IDrawingObjects

Базовый интерфейс для коллекций графических объектов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

Примечание:

1. Данный интерфейс является базовым для коллекций графических объектов.
2. Интерфейс может быть получен с помощью метода IUnknown:: QueryInterface от интерфейса коллекции конкретных объектов, например, от интерфейса видов IViews, слоев ILayers.

IDrawingObjects – свойства

Item – Объект, заданный по ссылке или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

Item = iObject.Item	Получить свойство (*)
Item = iObject.GetItem()	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (&Item) Получить свойство

Входные параметры:

Index - индекс объекта в коллекции, тип VARIANT.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс элемента в массиве, reference объекта.

Интерфейс ILayers

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Интерфейс коллекции слоев графического документа.

Иерархия:

IKompasAPIObject
IKompasCollection
IDrawingObjects
ILayers

Примечание:

Интерфейс коллекции слоев можно получить от интерфейса вида с помощью свойства IView::Layers или у группы слоев с помощью свойства ILayerGroup::Layers.

ILayers – свойства

Layer – Слой, заданный по индексу

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Тип данных: указатель на интерфейс слоя ILayer

Синтаксис Automation:

Layer = iObject.Layer (Index) Получить свойство (*)
Layer = iObject.GetLayer (Index) Получить свойство (**)

Синтаксис COM:

iObject->get_Layer (Index, &Layer) Получить свойство

Входные параметры:

index	- индекс слоя в коллекции; тип VARIANT.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс элемента в массиве, ссылка на слой, имя слоя.

LayerByNumber – Слой, заданный по номеру

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Тип данных: указатель на интерфейс слоя ILayer

Синтаксис Automation:

Layer = iObject.LayerByNumber (Number) Layer = iObject.GetLayerByNumber (Number)	Получить свойство (*) Получить свойство (**)

Синтаксис COM:

iObject->get_LayerByNumber (Number, &Layer)	Получить свойство

Входные параметры:

number	- номер слоя; тип long.

Примечание:

Свойство доступно только для чтения.

ILayers – методы

Add – Создать слой (добавляет слой в коллекцию)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add ([out, retval] ILayer** Layer);

Возвращаемое значение:

- Указатель на интерфейс слоя ILayer.

Примечание:

1. Интерфейс слоя добавляется в коллекцию.
2. Слой в документе будет создан после вызова метода IDrawingObject::Update.

Attach – Добавить существующий слой

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Синтаксис Automation:

BOOL Attach (LPDISPATCH pVal);

Синтаксис COM:

HRESULT Attach ([in] ILayer * pVal, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

pVal - Указатель на интерфейс слоя ILayer.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Интерфейс слоя добавляется в коллекцию.
2. Метод работает только для коллекции слоев группы ILayerGroup, если она не является динамической.

Detach – Отсоединить слой

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Тип данных : указатель на интерфейс слоя ILayer.

Синтаксис Automation:

BOOL Detach (LPDISPATCH pVal);

Синтаксис COM:

HRESULT Detach ([in] ILayer * pVal, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

pVal - Указатель на интерфейс слоя ILayer.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Интерфейс слоя отсоединяется от коллекции, слой при этом из документа не удаляется.
2. Метод работает только для коллекции слоев группы ILayerGroup, если она не является динамической.

Интерфейс IViews

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Интерфейс коллекции видов графического документа.

Иерархия:

IKompasAPIObject
IKompasCollection
IDrawingObjects
IViews

Описание:

Типы видов описаны в перечислении LtViewType.

В документе типа Фрагмент есть только один системный вид.

В документе типа Чертеж обязательно есть системный вид (только один) и могут быть до 254 видов любого типа из перечисления LtViewType, кроме системного.

Примечание:

Данный интерфейс можно получить от интерфейса менеджера видов и слоев IViewsAndLayersManager с помощью свойства IViewsAndLayersManager::Views.

IViews – свойства

ActiveView – Активный вид

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Тип данных: указатель на интерфейс вида IView

Синтаксис Automation:

ActiveView = iObject.ActiveView	Получить свойство (*)
ActiveView = iObject.GetActiveView	Получить свойство (**)

Синтаксис COM:

iObject->get_ActiveView (&ActiveView)	Получить свойство

Примечание:

Свойство доступно только для чтения.

View – Вид, заданный по имени, ссылке или по индексу

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Тип данных: указатель на интерфейс вида IView

Синтаксис Automation:

View = iObject.View (Index)	Получить свойство (*)
View = iObject.GetView (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_View (Index, &View)		Получить свойство

Входные параметры:

index	- индекс вида в коллекции, ссылка на вид или имя вида; тип VARIANT.

Примечание:

Свойство доступно только для чтения.

ViewByNumber – Вид, заданный по номеру

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Тип данных: указатель на интерфейс вида IView

Синтаксис Automation:

View = iObject.ViewByNumber (number)	Получить свойство (*)
View = iObject.GetViewByNumber (number)	Получить свойство (**)

Синтаксис COM:

iObject->get_ViewByNumber (number, &View)	Получить свойство

Входные параметры:

number	- номер вида; тип long.

Примечание:

Свойство доступно только для чтения.

IViews – методы

Add – Создать вид (добавляет вид в коллекцию)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Синтаксис Automation:

LPDISPATCH Add (long ViewType);

Синтаксис COM:

HRESULT Add ([in] LtViewType ViewType, [out, retval] IView** Result);

Входные параметры:

ViewType	- тип создаваемого вида из перечисления LtViewType.

Возвращаемое значение:

	- Указатель на интерфейс вида IView.

Примечание:

1. Системный вид создается автоматически при создании документа. Создавать второй системный вид (тип vt_System) нельзя.
2. После вызова этого метода вид будет добавлен в коллекцию, но в документе не появится.
3. Для отображения вида в документе нужно задать его параметры и вызвать метод IDrawingObject::Update.

AddStandartViews – Создать группу стандартных ассоциативных видов (добавить виды в коллекцию)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Синтаксис Automation:

BOOL AddStandartViews (BSTR FileName,
BSTR ProjectionName,
VARIANT ProjectionsTypes,
double X,
double Y,
double Scale,
double DX,
double DY);

Синтаксис COM:

HRESULT AddStandartViews (BSTR FileName,
BSTR ProjectionName,
VARIANT ProjectionsTypes,
double X,
double Y,
double Scale,
double DX,
double DY,
BOOL * Result);

Входные параметры:

FileName	- имя файла модели-источника для проекции,
ProjectionName	- ориентация модели,
ProjectionsTypes	- массив SAFEARRAY VT I VT_I4 типов проекций,
X, Y	- точка привязки группы видов,
Scale	- масштаб,
DX, DY	- расстояние между видами.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс IDrawingObject

Базовый интерфейс для всех графических объектов.

Иерархия:

IKompasAPIObject

IDrawingObject

IDrawingObject1

Примечание:

1. Данный интерфейс является базовым для графических объектов.
2. Интерфейс можно получить из коллекции графических объектов IDrawingObjects с помощью свойства IDrawingObjects::Item.
3. Имея интерфейс IDrawingObject, можно подписаться на события ksDrawingObjectNotify.
4. Данный интерфейс может быть получен с помощью метода IUnknown QueryInterface от интерфейса конкретного объекта, например, от интерфейса вида IView, слоя ILayer.
5. Посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif) у данного интерфейса можно получить дополнительный интерфейс для графических объектов IDrawingObject1.

Интерфейс событий для объектов графического документа...

IDrawingObject – свойства

DrawingObjectType – Тип графического объекта

Интерфейс...

Тип данных: DrawingObjectTypeEnum.

Синтаксис Automation:

DrawingObjectType iObject.DrawingObjectType	=	Получить свойство(*)
DrawingObjectType iObject.GetDrawingObjectType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingObjectType (&DrawingObjectType)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

DrawingObjectParamType – Тип для получения и изменения параметов объекта

Интерфейс...

Тип данных: ksDrawingObjectParamTypeEnum.

Синтаксис Automation:

type = iObject.DrawingObjectParamType		Получить свойство(*)
iObject.DrawingObjectParamType = type		Установить свойство (*)
type = iObject.GetDrawingObjectParamType()		Получить свойство (**)
iObject.SetDrawingObjectParamType (type)		Установить свойство (**)

Синтаксис COM:

iObject->get_DrawingObjectParamType (&type)	Получить свойство
iObject->put_DrawingObjectParamType (type)	Установить свойство

Примечание:

Свойство позволяет установить и получить тип используемой матрицы для пересчета параметров объектов. От матрицы зависят возвращаемые значения координат, углов и скалярных величин размеров, например, радиус, высота, длина.

LayerNumber – Номер слоя, на котором расположен объект. Для вида – номер активного слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

LayerNumber = iObject.LayerNumber		Получить свойство(*)
iObject.LayerNumber = LayerNumber		Установить свойство (*)
LayerNumber = iObject.GetLayerNumber()		Получить свойство (**)
iObject.SetLayerNumber (LayerNumber)		Установить свойство (**)

Синтаксис COM:

iObject->get_LayerNumber (&LayerNumber)	Получить свойство
iObject->put_LayerNumber (LayerNumber)	Установить свойство

Примечание:

Позволяет получить и установить номер слоя.

Temp – Признак временности объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Temp = iObject.Temp	Получить свойство(*)
Temp = iObject.GetTemp()	Получить свойство (**)

Синтаксис COM:

iObject->get_Temp (&Temp)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Valid – Признак невырожденности объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Valid = iObject.Valid	Получить свойство(*)
Valid = iObject.GetValid()	Получить свойство (**)

Синтаксис COM:

iObject->get_Valid (&Valid)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IDrawingObject – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL* pRes);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После успешного выполнения метода объект будет удален из модели. Свойство Valid для объекта будет возвращать FALSE. Все методы интерфейса возвращают ошибку (FALSE).

Update – Обновить данные объекта

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update ([out, retval] VARIANT_BOOL* pRes);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод Update необходимо вызвать для объекта, свойства которого были изменены (после вызова Set- свойств или методов интерфейса), чтобы эти изменения вступили в силу.
2. Метод Update вернёт TRUE в случае успеха и FALSE в случае неудачи.
3. Вызов Update сработает как при изменении одного свойства, так и при изменении группы свойств объекта.

Интерфейс ILayer

Справка системы КОМПАС...

KOMPAS.chm::/395_Glava46_Sloi.htm

Интерфейс слоя графического документа.

Иерархия:

IKompasAPIObject

IDrawingObject

ILayer

Описание:

В графическом документе в каждом виде всегда присутствует хотя бы один слой, который создается автоматически при создании документа и нового вида.

Примечание:

Данный интерфейс можно получить через коллекцию слоев ILayers с помощью свойства ILayers::Layer, ILayers::LayerByNumber, метода ILayers::Add, либо с помощью свойства IDrawingObjects::Item с последующим приведением к данному интерфейсу с помощью метода IUnknown QueryInterface.

ILayer – свойства

Background – Состояние слоя – фоновый

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Background = iObject.Background	Получить свойство (*)
iObject.Background = Background	Установить свойство (*)
Background = iObject.GetBackground()	Получить свойство (**)
iObject.SetBackground (Background)	Установить свойство (**)

Синтаксис COM:

iObject->get_Background (&Background)	Получить свойство
iObject->put_Background (Background)	Установить свойство

Примечание:

1. Позволяет получить и установить свойство слоя - фоновый.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Color – Цвет слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

Color = iObject.Color	Получить свойство (*)
iObject.Color = Color	Установить свойство (*)
Color = iObject.GetColor()	Получить свойство (**)
iObject.SetColor (Color)	Установить свойство (**)

Синтаксис COM:

iObject->get_Color (&Color)	Получить свойство
iObject->put_Color (Color)	Установить свойство

Примечание:

1. Позволяет получить и установить цвет слоя.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Comment – Примечание

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Comment = iObject.Comment	Получить свойство (*)
iObject.Comment = Comment	Установить свойство (*)

```
Comment =                    Получить свойство (**)
iObject.GetComment()
iObject.SetComment           Установить свойство (**)
(Comment)
```

Синтаксис COM:

```
iObject->get_Comment         Получить свойство
(&Comment)
iObject->put_Comment         Установить свойство
(Comment)
```

Примечание:

1. Позволяет получить и установить примечание для слоя.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Current – Состояние слоя – текущий

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Current = iObject.Current    Получить свойство (* )
iObject.Current = Current    Установить свойство (* )
Current =                    Получить свойство (**)
iObject.GetCurrent()
iObject.SetCurrent           Установить свойство (**)
(Current)
```

Синтаксис COM:

```
iObject->get_Current         Получить свойство
(&Current)
iObject->put_Current         Установить свойство
(Current)
```

Примечание:

1. Позволяет получить и установить состояние слоя - текущий.
2. Установить свойство текущий можно только равным TRUE. Новый слой всегда создается текущим.
3. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Name – Имя слоя

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName()	Получить свойство (**)
iObject.SetName (Name)	Установить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство
iObject->put_Name (Name)	Установить свойство

Примечание:

1. Позволяет получить и установить имя слоя.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

ObjectCount – Количество объектов в слое

Интерфейс...

Тип данных: long

Синтаксис Automation:

ObjectCount = iObject.ObjectCount	Получить свойство (*)
ObjectCount = iObject.GetObjectCount()	Получить свойство (**)

Синтаксис COM:

iObject->get_ObjectCount (&ObjectCount)	Получить свойство

Примечание:

1. Позволяет получить количество объектов в слое.
2. Свойство доступно только для чтения.

Printable – Признак разрешения/запрещения печати слоя

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Printable = Object.Printable	Получить свойство (*)
Object.Printable = Printable	Установить свойство (*)
Printable = Object.GetPrintable()	Получить свойство (**)
Object.SetPrintable (Printable)	Установить свойство (**)

Синтаксис COM:

Object.get_Printable (&Printable)	Получить свойство
Object.put_Printable (Printable)	Установить свойство

Visible – Состояние слоя – видимый или погашенный

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство (**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

1. Позволяет получить и установить свойство слоя - видимый или погашенный.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Интерфейс IView

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Интерфейс вида графического документа.

Иерархия:

IKompasAPIObject

IDrawingObject

IView

IViewDesignation

IDrawingContainer

IBuildingContainer

ISymbols2DContainer

IBreakViewParam

ICutViewParam

IView1

Описание:

Типы видов описаны в перечислении LtViewType.

В графическом документе всегда присутствует системный вид, который формируется автоматически при создании документа, и может быть только один. В документе типа Фрагмент может быть только один вид - системный.

Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы вида:

- Интерфейс обозначения вида IViewDesignation.
- Интерфейс контейнера объектов вида графического документа IDrawingContainer.
- Интерфейс контейнера объектов СПДС IBuildingContainer.
- Интерфейс контейнера условных обозначений ISymbols2DContainer.
- Интерфейс параметров разрыва вида IBreakViewParam. Этот интерфейс может быть получен для видов всех типов, кроме системного.
- Интерфейс параметров разреза вида ICutViewParam. Этот интерфейс может быть получен для видов всех типов, кроме системного.
- Интерфейс для получения коллекции ЛСК и признака редактирования макроэлемента IView1.

Примечание:

Данный интерфейс можно получить следующими способами:

- через коллекцию видов IViews,
- с помощью свойства IViews::View, IViews::ViewByNumber или метода IViews::Add, либо с помощью свойства IDrawingObjects::Item с последующим приведением к данному интерфейсу с помощью метода IUnknown QueryInterface,
- от интерфейса группы слоев ILayerGroup с помощью свойства ILayerGroup::OwnerView,

IView – свойства

Angle – Угол поворота вида

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle	Получить свойство (*)
iObject.Angle = Angle	Установить свойство (*)
iObject.Angle = Angle	Получить свойство (**)
iObject.SetAngle (Angle)	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle (&Angle)	Получить свойство
iObject->put_Angle (Angle)	Установить свойство

Примечание:

1. Позволяет получить и установить угол поворота вида.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Background – Состояние вида – фоновый

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Background = iObject.Background	Получить свойство (*)
iObject.Background = Background	Установить свойство (*)
Background = iObject.GetBackground()	Получить свойство (**)
iObject.SetBackground (Background)	Установить свойство (**)

Синтаксис COM:

iObject->get_Background (&Background)	Получить свойство
iObject->put_Background (Background)	Установить свойство

Примечание:

1. Позволяет получить и установить свойство слоя - фоновый.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Color – Цвет вида

Интерфейс...

Тип данных: long

Синтаксис Automation:

Color = iObject.Color	Получить свойство (*)

```
iObject.Color = Color                    Установить свойство (* )
Color = iObject.GetColor()               Получить свойство (**)
iObject.SetColor( Color )                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Color( &Color )             Получить свойство
iObject->put_Color( Color )              Установить свойство
```

Примечание:

1. Позволяет получить и установить цвет вида.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Comment – Примечание

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Comment = iObject.Comment                Получить свойство (* )
iObject.Comment = Comment                Установить свойство (* )
Comment = iObject.GetComment()           Получить свойство (**)
iObject.SetComment (Comment)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Comment (&Comment)          Получить свойство
iObject->put_Comment (Comment)           Установить свойство
```

Примечание:

1. Позволяет получить и установить примечание.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Current – Состояние вида – текущий

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Current = iObject.Current                Получить свойство (* )
iObject.Current = Current                Установить свойство (* )
Current = iObject.GetCurrent()           Получить свойство (**)
iObject.SetCurrent (Current)             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Current (&Current)          Получить свойство
```

iObject->put_Current (Current)	Установить свойство

Примечание:

1. Позволяет узнать, является ли вид текущим или сделать его текущим.
2. Установить свойство "текущий" можно только равным TRUE.
3. Новый вид всегда создается текущим.
4. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Layers – Коллекция слоев

Интерфейс...

Тип данных: указатель на интерфейс ILayers коллекции слоев.

Синтаксис Automation:

Layers = iObject.Layers	Получить свойство (*)
Layers = iObject.GetLayers()	Получить свойство (**)

Синтаксис COM:

iObject->get_Layers(&Layers)	Получить свойство

Примечание:

1. Позволяет получить коллекцию слоев вида.
2. Свойство доступно только для чтения.

Name – Имя вида

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName()	Получить свойство (**)
iObject.SetName (Name)	Установить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство
iObject->put_Name (Name)	Установить свойство

Примечание:

1. Для нового вида, полученного методом IViews::Add, имя должно быть обязательно установлено до вызова метода IDrawingObject::Update.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Number – Номер вида

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
iObject.Number = Number	Установить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)
iObject.SetNumber (Number)	Установить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство
iObject->put_Number (Number)	Установить свойство

Примечание:

Номер вида "0" зарезервирован для системного вида. Номер должен быть уникальным в документе. Для нового вида, полученного методом IViews::Add, можно задать номер "-1" или не задавать номер. В этом случае при вызове метода IDrawingObject::Update для данного вида номер будет задан автоматически.

ObjectCount – Количество объектов в виде

Интерфейс...

Тип данных: long

Синтаксис Automation:

ObjectCount = iObject.ObjectCount	Получить свойство (*)
ObjectCount = iObject.GetObjectCount()	Получить свойство (**)

Синтаксис COM:

iObject->get_ObjectCount (&ObjectCount)	Получить свойство

Примечание:

1. Позволяет получить количество объектов в виде.
2. Свойство доступно только для чтения.

Scale – Масштаб вида

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = iObject.Scale	Получить свойство (*)
iObject.Scale = Scale	Установить свойство (*)
Scale = iObject.GetScale()	Получить свойство (**)
iObject.SetScale (Scale)	Установить свойство (**)

Синтаксис COM:

iObject->get_Scale (&Scale)	Получить свойство
iObject->put_Scale (Scale)	Установить свойство

Примечание:

1. Позволяет получить и установить масштаб вида.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Variable – Получить параметрическую переменную по имени, индексу или указателю на размер

Интерфейс...

Тип данных: указатель на интерфейс параметрической переменной IVariable7

Синтаксис Automation:

Variable = Object.Variable (VARIANT Index)	Получить свойство (*)
Variable = Object.GetVariable (VARIANT Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Variable(VARIANT Index, &Variable)	Получить свойство

Входные параметры:

Index	- индекс, имя или указатель на размер параметрической переменной.

Свойство позволяет получать интерфейс параметрической переменной.

Примечание:

Свойство доступно только для чтения.

Variables – Получить массив параметрических переменных

Интерфейс...

Тип данных: массив SAFEARRAY VT_DISPATCH

Синтаксис Automation:

Variables = Object.Variables	Получить свойство (*)
Variables = Object.GetVariables()	Получить свойство (**)

Синтаксис COM:

Object.get_Variables (&Variables)	Получить свойство

Свойство позволяет получать массив параметрических переменных.

Примечание:

Свойство доступно только для чтения.

VariablesCount – Получить количество параметрических переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

VariablesCount = Object.VariablesCount	Получить свойство (*)
VariablesCount = Object.GetVariablesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_VariablesCount (&VariablesCount)	Получить свойство

Свойство позволяет получать количество параметрических переменных.

Примечание:

Свойство доступно только для чтения.

ViewType – Тип вида

Интерфейс...

Тип данных: Тип вида из перечисления LtViewType

Синтаксис Automation:

ViewType = iObject.ViewType	Получить свойство (*)
ViewType = iObject.GetViewType()	Получить свойство (**)

Синтаксис COM:

iObject->get_ViewType (&ViewType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Visible – Состояние вида – видимый или погашенный

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство (**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

1. Позволяет получить и установить свойство вида - видимый или погашенный.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

X – Координата привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = iObject.X	Получить свойство (*)
iObject.X = X	Установить свойство (*)
X = iObject.GetX()	Получить свойство (**)
iObject.SetX (X)	Установить свойство (**)

Синтаксис COM:

iObject->get_X (&X)	Получить свойство
iObject->put_X (X)	Установить свойство

Примечание:

1. Позволяет получить и установить координату привязки вида по оси X.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Y – Координата привязки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = iObject.Y	Получить свойство (*)
iObject.Y = Y	Установить свойство (*)
Y = iObject.GetY()	Получить свойство (**)
iObject.SetY (Y)	Установить свойство (**)

Синтаксис COM:

iObject->get_Y (&Y)	Получить свойство
iObject->put_Y (Y)	Установить свойство

Примечание:

1. Позволяет получить и установить координату привязки вида по оси Y.
2. Свойство вступает в силу после вызова метода IDrawingObject::Update.

Интерфейс IAssociationView

Справка системы КОМПАС...

`kompas.chm::/422_Glava49_Obshchie_svedenija_.htm`

Интерфейс ассоциативного вида графического документа.

Иерархия:

IKompasAPIObject

IDrawingObject

IView

IAssociationView

IHatchParam

IAssociationViewElements

Описание:

Типы видов описаны в перечислении LtViewType.

Примечание:

1. Данный интерфейс можно получить следующими способами:

- через коллекцию видов IViews,
- с помощью метода IUnknown QueryInterface от интерфейса IView.

2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить дополнительные интерфейсы:

- IHatchParam
- IAssociationViewElements.

IAssociationView – свойства

BaseObject – Опорный объект

Интерфейс...

Тип данных:указатель на интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство,
Object.put_BaseObject(BaseObject)	Установить свойство.

Примечание:

В зависимости от типа ассоциативного вида нужно указать вспомогательный опорный объект.

Опорный объект требуется при создании следующих видов:

- Вид по стрелке- требуется объект Стрелка взгляда,
- Выносной вид- требуется объект Обозначение выносного элемента,
- Вид разрез/сечение- требуется объект Линия разреза.

BaseView – Опорный вид

Интерфейс...

Тип данных: указатель на интерфейс вида IView

Синтаксис Automation:

BaseView = Object.BaseView	Получить свойство (*)
Object.BaseView = BaseView	Установить свойство (*)
BaseView = Object.GetBaseView()	Получить свойство (**)
Object.SetBaseView(BaseView)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseView(&BaseView)	Получить свойство,

Object.put_BaseView(BaseView)	Установить свойство.

Примечание:

Свойство используется при создании проекционного вида.

Позволяет установить вид, относительно которого выполняется производная проекция.

Базовый вид должен быть ассоциативным.

BendLinesStyle – Стиль линий сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BendLinesStyle = Object.BendLinesStyle	Получить свойство (*)
Object.BendLinesStyle = BendLinesStyle	Установить свойство (*)
BendLinesStyle = Object.GetBendLinesStyle()	Получить свойство (**)
Object.SetBendLinesStyle(BendLinesStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_BendLinesStyle(&BendLinesStyle)	Получить свойство,
Object.put_BendLinesStyle(BendLinesStyle)	Установить свойство.

Примечание:

Позволяет установить системный (ksCurveStyleEnum) или пользовательский стиль линий для отображения линий сгиба.

BendLinesVisible – Показывать линии сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BendLinesVisible = Object.BendLinesVisible	=	Получить свойство (*)
Object.BendLinesVisible = BendLinesVisible	=	Установить свойство (*)
BendLinesVisible = Object.GetBendLinesVisible()	=	Получить свойство (**)
Object.SetBendLinesVisible(BendLinesVisible)		Установить свойство (**)

Синтаксис COM:

Object.get_BendLinesVisible(&BendLinesVisible)	Получить свойство,

Object.put_BendLinesVisible(BendLinesVisible)		Установить свойство.

Свойство позволяет получить и установить признак отображения линий сгиба.

BreakLinesStyle – Стиль линий перехода

Интерфейс...

Тип данных: long

Синтаксис Automation:

BreakLinesStyle = Object.BreakLinesStyle	=	Получить свойство (*)
Object.BreakLinesStyle = BreakLinesStyle	=	Установить свойство (*)
BreakLinesStyle = Object.GetBreakLinesStyle()	=	Получить свойство (**)
Object.SetBreakLinesStyle(BreakLinesStyle)		Установить свойство (**)

Синтаксис COM:

Object.get_BreakLinesStyle(&BreakLinesStyle)	Получить свойство,
Object.put_BreakLinesStyle(BreakLinesStyle)	Установить свойство.

Примечание:

Позволяет установить системный (ksCurveStyleEnum) или пользовательский стиль линий для отображения линий перехода.

BreakLinesVisible – Показывать линии перехода

Интерфейс...

Тип данных: long

Синтаксис Automation:

BreakLinesVisible = Object.BreakLinesVisible	=	Получить свойство (*)
Object.BreakLinesVisible = BreakLinesVisible	=	Установить свойство (*)
BreakLinesVisible = Object.GetHiddenLinesVisible()	=	Получить свойство (**)
Object.SetBreakLinesVisible(BreakLinesVisible)		Установить свойство (**)

Синтаксис COM:

Object.get_BreakLinesVisible(&BreakLinesVisible)		Получить свойство,
Object.put_BreakLinesVisible(BreakLinesVisible)		Установить свойство.

CenterLinesVisible – Показывать осевые линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CenterLinesVisible = Object.CenterLinesVisible	=	Получить свойство (*)
Object.CenterLinesVisible = CenterLinesVisible	=	Установить свойство (*)
CenterLinesVisible = Object.GetCenterLinesVisible()	=	Получить свойство (**)
Object.SetCenterLinesVisible(CenterLinesVisible)		Установить свойство (**)

Синтаксис COM:

Object.get_CenterLinesVisible(&CenterLinesVisible)	Получить свойство,
Object.put_CenterLinesVisible(CenterLinesVisible)	Установить свойство.

DimensionLayoutScaling – Масштабировать аннотационные объекты вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Object.DimensionLayoutScaling = DimensionLayoutScaling	=	Установить свойство (*)
Object.SetDimensionLayoutScaling(DimensionLayoutScaling)		Установить свойство (**)

Синтаксис COM:

Object.put_DimensionLayoutScaling(DimensionLayoutScaling)	Установить свойство.

Примечание:

1. Свойство доступно только для изменения.
2. Значение свойства не сохраняется и используется при создании и изменении параметров вида.

ExplodedView – Сборка в разнесенном виде

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ExplodedView = Object.ExplodedView                Получить свойство (* )
Object.ExplodedView = ExplodedView                Установить свойство (* )
ExplodedView = Object.GetExplodedView()           Получить свойство (**)
Object.SetExplodedView( ExplodedView )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ExplodedView( &ExplodedView )          Получить свойство,
Object.put_ExplodedView(  ExplodedView )          Установить свойство.
```

HiddenLines – Передавать невидимые линии в вид

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
HiddenLines = Object.HiddenLines                  Получить свойство (* )
Object.HiddenLines = HiddenLines                  Установить свойство (* )
HiddenLines = Object.GetHiddenLines()             Получить свойство (**)
Object.SetHiddenLines( HiddenLines )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HiddenLines( &HiddenLines )            Получить свойство,
Object.put_HiddenLines( HiddenLines )             Установить свойство.
```

Позволяет получить и установить признак отображения невидимых линий при проецировании.

Примечание:

Невидимые линии хранятся в виде вне зависимости от того, включена их отрисовка или нет (IAssociationView::HiddenLinesVisible).

Если отрисовка этих линий не требуется, отключите опцию *Невидимые линии* — это уменьшит размер файла чертежа.

Рекомендуется отключать опцию *Невидимые линии* при построении ассоциативных видов сборок, содержащих более 1000 компонентов.

HiddenLinesStyle – Стиль невидимых линий

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
HiddenLinesStyle = Object.HiddenLinesStyle          Получить свойство (* )
Object.HiddenLinesStyle = HiddenLinesStyle          Установить свойство (* )
HiddenLinesStyle = Object.GetHiddenLinesStyle()     Получить свойство (**)
Object.SetHiddenLinesStyle( HiddenLinesStyle )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HiddenLinesStyle( &HiddenLinesStyle )    Получить свойство,
Object.put_HiddenLinesStyle( HiddenLinesStyle )     Установить свойство.
```

Примечание:

Позволяет установить системный (ksCurveStyleEnum) или пользовательский стиль линий для отображения невидимых линий

Для использования данного свойства требуется включение передачи невидимых линий (IAssociationView::HiddenLines) и включение отображения невидимых линий (IAssociationView::HiddenLinesVisible).

HiddenLinesVisible – Показывать невидимые линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
HiddenLinesVisible = Object.HiddenLinesVisible          Получить свойство (* )
Object.HiddenLinesVisible = HiddenLinesVisible          Установить свойство (* )
HiddenLinesVisible = Object.GetHiddenLinesVisible()     Получить свойство (**)
Object.SetHiddenLinesVisible( HiddenLinesVisible )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HiddenLinesVisible(                  Получить свойство,
&HiddenLinesVisible )
Object.put_HiddenLinesVisible(                  Установить свойство.
HiddenLinesVisible )
```

Свойство позволяет получить и установить признак отображения невидимых линий.

Примечание:

Для использования данного свойства требуется включение передачи невидимых линий (IAssociationView::HiddenLines).

Local – Местный вид

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Local = Object.Local	Получить свойство (*)
Object.Local = Local	Установить свойство (*)
Local = Object.GetLocal()	Получить свойство (**)
Object.SetLocal(Local)	Установить свойство (**)

Синтаксис COM:

Object.get_Local(&Local)	Получить свойство,
Object.put_Local(Local)	Установить свойство.

Примечание:

Включает полное изображение вида или усеченное (местный вид).

Требует предварительного создания местного вида с помощью команды IAssociationView::CreateLocalView

ProjectionMatrix – Матрица ассоциативного вида

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ProjectionMatrix = Object.ProjectionMatrix	=	Получить свойство (*)
Object.ProjectionMatrix = ProjectionMatrix	=	Установить свойство (*)
ProjectionMatrix = Object.GetProjectionMatrix()	=	Получить свойство (**)
Object.SetProjectionMatrix(ProjectionMatrix)		Установить свойство (**)

Синтаксис COM:

Object.get_ProjectionMatrix(&ProjectionMatrix)	Получить свойство,

Object.put_ProjectionMatrix(ProjectionMatrix)	Установить свойство.

Примечание

Матрица представляет собой массив SAFEARRAY double (VT_ARRAY | VT_R8)

В массиве будут лежать 16 элементов, которые представляют собой матрицу размера 4x4.

ProjectionName – Имя проекции

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ProjectionName = Object.ProjectionName	Получить свойство (*)
Object.ProjectionName = ProjectionName	Установить свойство (*)
ProjectionName = Object.GetProjectionName()	Получить свойство (**)
Object.SetProjectionName (ProjectionName)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectionName (&ProjectionName)	Получить свойство,
Object.put_ProjectionName (ProjectionName)	Установить свойство.

Свойство позволяет получить и установить имя проекции (из списка проекций в документе-источнике).

ProjectionLink – Проекционная связь

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectionLink = Object.ProjectionLink	Получить свойство (*)
Object.ProjectionLink = ProjectionLink	Установить свойство (*)
ProjectionLink = Object.GetProjectionLink()	Получить свойство (**)
Object.SetProjectionLink(ProjectionLink)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectionLink(&ProjectionLink)	Получить свойство,
Object.put_ProjectionLink(ProjectionLink)	Установить свойство.

Свойство позволяет получить и установить состояние проекционной связи между ассоциативными видами.

SameHatch – Признак одинаковой штриховки всех деталей сборки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
SameHatch = Object.SameHatch                Получить свойство (* )
Object.SameHatch = SameHatch                Установить свойство (* )
SameHatch = Object.GetSameHatch()           Получить свойство (**)
Object.SetSameHatch( SameHatch )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SameHatch( &SameHatch )          Получить свойство,
Object.put_SameHatch( SameHatch )           Установить свойство.
```

Примечание:

Свойство используется только в виде разрез/сечение.

Section – Признак «разрез/сечение»

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Section = Object.Section                    Получить свойство (* )
Object.Section = Section                    Установить свойство (* )
Section = Object.GetSection()               Получить свойство (**)
Object.SetSection( Section )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Section( &Section )              Получить свойство,
Object.put_Section( Section )               Установить свойство.
```

Свойство позволяет получить и установить признак «разрез/сечение».

SourceFileName – Полное имя файла-источника

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

SourceFileName = Object.SourceFileName	=	Получить свойство (*)
Object.SourceFileName = SourceFileName	=	Установить свойство (*)
SourceFileName = Object.GetSourceFileName()	=	Получить свойство (**)
Object.SetSourceFileName (SourceFileName)		Установить свойство (**)

Синтаксис COM:

Object.get_SourceFileName (&SourceFileName)	Получить свойство,
Object.put_SourceFileName (SourceFileName)	Установить свойство.

Свойство позволяет получить и установить полное имя файла модели или сборки, для которой создан ассоциативный вид.

Unfold – Листовая деталь в разогнутом виде

Интерфейс...

Тип данных:BOOL

Синтаксис Automation:

Unfold = Object.Unfold	Получить свойство (*)
Object.Unfold = Unfold	Установить свойство (*)
Unfold = Object.GetUnfold()	Получить свойство (**)
Object.SetUnfold(Unfold)	Установить свойство (**)

Синтаксис COM:

Object.get_Unfold(&Unfold)	Получить свойство,
Object.put_Unfold(Unfold)	Установить свойство.

UseOcclusion – Черновое проецирование

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseOcclusion = Object.UseOcclusion	Получить свойство (*)

Object.UseOcclusion = UseOcclusion — Установить свойство (*)
UseOcclusion = Object.GetUseOcclusion() — Получить свойство (**)
Object.SetUseOcclusion(UseOcclusion) — Установить свойство (**)

Синтаксис COM:

Object.get_UseOcclusion(&UseOcclusion) — Получить свойство,
Object.put_UseOcclusion(UseOcclusion) — Установить свойство.

Версия Компас v18.1

VisibleLinesStyle – Стиль видимых линий

Интерфейс...

Тип данных: long

Синтаксис Automation:

VisibleLinesStyle = Object.VisibleLinesStyle — Получить свойство (*)
Object.VisibleLinesStyle = VisibleLinesStyle — Установить свойство (*)
VisibleLinesStyle = Object.GetVisibleLinesStyle() — Получить свойство (**)
Object.SetVisibleLinesStyle(VisibleLinesStyle) — Установить свойство (**)

Синтаксис COM:

Object.get_VisibleLinesStyle(&VisibleLinesStyle) — Получить свойство,
Object.put_VisibleLinesStyle(VisibleLinesStyle) — Установить свойство.

Позволяет установить системный (ksCurveStyleEnum) или пользовательский стиль линий для отображения видимых линий.

IAssociationView – методы

CreateLocalView – Местный вид

Интерфейс...

Синтаксис Automation:

BOOL CreateLocalView(LPDISPATCH Contour);

Синтаксис COM:

HRESULT CreateLocalView(IDrawingObject * Contour, BOOL * Result);

Входные параметры:

Contour - указатель на замкнутую кривую.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Границей местного вида может являться любой замкнутый контур (окружность, эллипс, замкнутая кривая Безье и т.п.).

Если в опорном виде нет замкнутого контура, создайте его

Необходимым условием корректного построения местного вида является расположение ограничивающего его контура именно в опорном виде.

Переключение между полным и усеченным отображением вида осуществляется с помощью свойства IAssociationView::Local

Интерфейс IAssociationViewElements

Справка системы КОМПАС...

KOMPAS.chm::/429_49_3_3_Obwekty.htm

Интерфейс – Объекты и элементы для передачи в ассоциативный вид.

Иерархия:

IDispatch

IAssociationViewElements

Описание:

Интерфейс позволяет задать, какие объекты и элементы надо передавать в ассоциативный вид.

Примечание:

Интерфейс является дополнительным к интерфейсу ассоциативного вида IAssociationView.

IAssociationViewElements – свойства

CreateAxis – Создавать оси

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateAxis = Object.CreateAxis	Получить свойство (*)
Object.CreateAxis = CreateAxis	Установить свойство (*)
CreateAxis = Object.GetCreateAxis()	Получить свойство (**)
Object.SetCreateAxis(CreateAxis)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateAxis(&CreateAxis)	Получить свойство,

Object.put_CreateAxis(CreateAxis)		Установить свойство.

CreateCentresMarkers – Создавать обозначения центров

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateCentresMarkers = Object.CreateCentresMarkers	=	Получить свойство (*)
Object.CreateCentresMarkers = CreateCentresMarkers	=	Установить свойство (*)
CreateCentresMarkers = Object.GetCreateCentresMarkers()	=	Получить свойство (**)
Object.SetCreateCentresMarkers(CreateCentresMarkers)		Установить свойство (**)

Синтаксис COM:

Object.get_CreateCentresMarkers(&CreateCentresMarkers)	Получить свойство,
Object.put_CreateCentresMarkers(CreateCentresMarkers)	Установить свойство.

CreateCircularCentres – Создавать круговые сетки центров

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateCircularCentres = Object.CreateCircularCentres	Получить свойство (*)
Object.CreateCircularCentres = CreateCircularCentres	Установить свойство (*)
CreateCircularCentres = Object.GetCreateCircularCentres()	Получить свойство (**)
Object.SetCreateCircularCentres(CreateCircularCentres)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateCircularCentres(&CreateCircularCentres)	Получить свойство,

Object.put_CreateCircularCentres(CreateCircularCentres)		Установить свойство.

CreateLinearCentres – Создавать линейные сетки центров

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateLinearCentres = Object.CreateLinearCentres	=	Получить свойство (*)
Object.CreateLinearCentres = CreateLinearCentres	=	Установить свойство (*)
CreateLinearCentres = Object.GetCreateLinearCentres()	=	Получить свойство (**)
Object.SetCreateLinearCentres(CreateLinearCentres)		Установить свойство (**)

Синтаксис COM:

Object.get_CreateLinearCentres(&CreateLinearCentres)	Получить свойство,
Object.put_CreateLinearCentres(CreateLinearCentres)	Установить свойство.

HiddenObjectsVisible – Отображать скрытые объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HiddenObjectsVisible = Object.HiddenObjectsVisible	Получить свойство (*)
Object.HiddenObjectsVisible = HiddenObjectsVisible	Установить свойство (*)
HiddenObjectsVisible = Object.GetHiddenObjectsVisible()	Получить свойство (**)
Object.SetHiddenObjectsVisible(HiddenObjectsVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_HiddenObjectsVisible (&HiddenObjectsVisible)	Получить свойство
Object.put_HiddenObjectsVisible (HiddenObjectsVisible)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак отображения скрытых объектов.

ProjectAllDesignations – Передавать в вид все обозначения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectAllDesignations = Object.ProjectAllDesignations	Получить свойство (*)
Object.ProjectAllDesignations = ProjectAllDesignations	Установить свойство (*)
ProjectAllDesignations = Object.GetProjectAllDesignations()	Получить свойство (**)
Object.SetProjectAllDesignations(ProjectAllDesignations)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectAllDesignations(&ProjectAllDesignations)	Получить свойство
Object.put_ProjectAllDesignations(ProjectAllDesignations)	Установить свойство

Свойство позволяет одновременно включить проецирование всех типов размеров и обозначений, включая обозначение резьбы.

ProjectAllObjects – Передавать в вид все объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectAllObjects = Object.ProjectAllObjects	Получить свойство (*)
Object.ProjectAllObjects = ProjectAllObjects	Установить свойство (*)
ProjectAllObjects = Object.GetProjectAllObjects()	Получить свойство (**)
Object.SetProjectAllObjects(ProjectAllObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectAllObjects(&ProjectAllObjects)	Получить свойство
Object.put_ProjectAllObjects(ProjectAllObjects)	Установить свойство

Примечание:

Свойство позволяет одновременно включить проецирование тел ProjectBodies, поверхностей ProjectSurfaces, кривых ProjectCurves и точек ProjectPoints.

См. также IAssociationViewElements::ProjectAllDesignations

ProjectAxis – Передавать в вид оси

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectAxis = Object.ProjectAxis	Получить свойство (*)
Object.ProjectAxis = ProjectAxis	Установить свойство (*)
ProjectAxis = Object.GetProjectAxis()	Получить свойство (**)
Object.SetProjectAxis(ProjectAxis)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectAxis(&ProjectAxis)	Получить свойство,
Object.put_ProjectAxis(ProjectAxis)	Установить свойство.

ProjectBases – Передавать в вид обозначения баз

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectBases = Object.ProjectBases	Получить свойство (*)
Object.ProjectBases = ProjectBases	Установить свойство (*)
ProjectBases = Object.GetProjectBases()	Получить свойство (**)
Object.SetProjectBases(ProjectBases)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectBases(&ProjectBases)	Получить свойство
Object.put_ProjectBases(ProjectBases)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений баз.

ProjectBodies – Передавать в вид тела

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectBodies = Object.ProjectBodies	Получить свойство (*)
Object.ProjectBodies = ProjectBodies	Установить свойство (*)
ProjectBodies = Object.GetProjectBodies()	Получить свойство (**)
Object.SetProjectBodies(ProjectBodies)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectBodies(&ProjectBodies)	Получить свойство
Object.put_ProjectBodies(ProjectBodies)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид тел.

ProjectBrandLeaders – Передавать в вид обозначения клеймений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectBrandLeaders = Object.ProjectBrandLeaders	Получить свойство (*)
Object.ProjectBrandLeaders = ProjectBrandLeaders	Установить свойство (*)
ProjectBrandLeaders = Object.GetProjectBrandLeaders()	Получить свойство (**)
Object.SetProjectBrandLeaders(ProjectBrandLeaders)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectBrandLeaders (&ProjectBrandLeaders)	Получить свойство
Object.put_ProjectBrandLeaders (ProjectBrandLeaders)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений клеймений.

ProjectCurves – Передавать в вид кривые

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectCurves = Object.ProjectCurves	Получить свойство (*)
Object.ProjectCurves = ProjectCurves	Установить свойство (*)
ProjectCurves = Object.GetProjectCurves()	Получить свойство (**)
Object.SetProjectCurves(ProjectCurves)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectCurves(&ProjectCurves)	Получить свойство
Object.put_ProjectCurves(ProjectCurves)	Установить свойство

Свойство позволяет включать и отключать проецирование в вид кривых.

ProjectDimensions – Передавать в вид обозначения размеров

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectDimensions = Object.ProjectDimensions	Получить свойство (*)
Object.ProjectDimensions = ProjectDimensions	Установить свойство (*)
ProjectDimensions = Object.GetProjectDimensions()	Получить свойство (**)
Object.SetProjectDimensions(ProjectDimensions)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectDimensions(&ProjectDimensions)	Получить свойство
Object.put_ProjectDimensions(ProjectDimensions)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений размеров.

ProjectHiddenComponents – Передавать в вид скрытые компоненты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectHiddenComponents = Object.ProjectHiddenComponents	Получить свойство (*)
Object.ProjectHiddenComponents = ProjectHiddenComponents	Установить свойство (*)
ProjectHiddenComponents = Object.GetProjectHiddenComponents()	Получить свойство (**)
Object.SetProjectHiddenComponents(ProjectHiddenComponents)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectHiddenComponents(&ProjectHiddenComponents)	Получить свойство
Object.put_ProjectHiddenComponents(ProjectHiddenComponents)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид скрытых компонентов.

ProjectLeaders – Передавать в вид обозначения линий-выносок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectLeaders = Object.ProjectLeaders	Получить свойство (*)
Object.ProjectLeaders = ProjectLeaders	Установить свойство (*)
ProjectLeaders = Object.GetProjectLeaders()	Получить свойство (**)
Object.SetProjectLeaders(ProjectLeaders)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectLeaders(&ProjectLeaders)	Получить свойство
Object.put_ProjectLeaders(ProjectLeaders)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений линий-выносок.

ProjectMarkLeaders – Передавать в вид обозначения маркировок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectMarkLeaders = Object.ProjectMarkLeaders	Получить свойство (*)
Object.ProjectMarkLeaders = ProjectMarkLeaders	Установить свойство (*)
ProjectMarkLeaders = Object.GetProjectMarkLeaders()	Получить свойство (**)
Object.SetProjectMarkLeaders(ProjectMarkLeaders)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectMarkLeaders(&ProjectMarkLeaders)	Получить свойство
Object.put_ProjectMarkLeaders(ProjectMarkLeaders)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений маркировок.

ProjectPoints – Передавать в вид точки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectPoints = Object.ProjectPoints	Получить свойство (*)
Object.ProjectPoints = ProjectPoints	Установить свойство (*)
ProjectPoints = Object.GetProjectPoints()	Получить свойство (**)
Object.SetProjectPoints(ProjectPoints)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectPoints(&ProjectPoints)	Получить свойство
Object.put_ProjectPoints(ProjectPoints)	Установить свойство

Свойство позволяет включать и отключать проецирование в вид точек.

ProjectPositions – Передавать в вид обозначения позиций

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectPositions = Object.ProjectPositions	Получить свойство (*)
Object.ProjectPositions = ProjectPositions	Установить свойство (*)
ProjectPositions = Object.GetProjectPositions()	Получить свойство (**)
Object.SetProjectPositions(ProjectPositions)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectPositions(&ProjectPositions)	Получить свойство
Object.put_ProjectPositions(ProjectPositions)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений позиций.

ProjectRoughs – Передавать в вид обозначения шероховатостей

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectRoughs = Object.ProjectRoughs	Получить свойство (*)
Object.ProjectRoughs = ProjectRoughs	Установить свойство (*)
ProjectRoughs = Object.GetProjectRoughs()	Получить свойство (**)
Object.SetProjectRoughs(ProjectRoughs)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectRoughs(&ProjectRoughs)	Получить свойство
Object.put_ProjectRoughs(ProjectRoughs)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений шероховатостей.

ProjectSketches – Передавать в вид эскизы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectSketches = Object.ProjectSketches	Получить свойство (*)
Object.ProjectSketches = ProjectSketches	Установить свойство (*)
ProjectSketches = Object.GetProjectSketches()	Получить свойство (**)
Object.SetProjectSketches(ProjectSketches)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectSketches(&ProjectSketches)	Получить свойство
Object.put_ProjectSketches(ProjectSketches)	Установить свойство

Версия Компас v18.1

ProjectSpecRough – Передавать в вид обозначения неуказанной шероховатости

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectSpecRough = Object.ProjectSpecRough	Получить свойство (*)
Object.ProjectSpecRough = ProjectSpecRough	Установить свойство (*)
ProjectSpecRough = Object.GetProjectSpecRough()	Получить свойство (**)
Object.SetProjectSpecRough(ProjectSpecRough)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectSpecRough(&ProjectSpecRough)	Получить свойство
Object.put_ProjectSpecRough(ProjectSpecRough)	Установить свойство

ProjectLayers – Учитывать при проецировании слои

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectLayers = Object.ProjectLayers	Получить свойство (*)
Object.ProjectLayers = ProjectLayers	Установить свойство (*)
ProjectLayers = Object.GetProjectLayers()	Получить свойство (**)
Object.SetProjectLayers(ProjectLayers)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectLayers(&ProjectLayers)	Получить свойство
Object.put_ProjectLayers(ProjectLayers)	Установить свойство

ProjectStandartElements – Передавать в вид библиотечные элементы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectStandartElements = Object.ProjectStandartElements	Получить свойство (*)
Object.ProjectStandartElements = ProjectStandartElements	Установить свойство (*)
ProjectStandartElements = Object.GetProjectStandartElements()	Получить свойство (**)
Object.SetProjectStandartElements(ProjectStandartElements)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectStandartElements(&ProjectStandartElements)	Получить свойство
Object.put_ProjectStandartElements(ProjectStandartElements)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид библиотечных элементов.

ProjectSurfaces – Передавать в вид поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectSurfaces = Object.ProjectSurfaces	Получить свойство (*)
Object.ProjectSurfaces = ProjectSurfaces	Установить свойство (*)
ProjectSurfaces = Object.GetProjectSurfaces()	Получить свойство (**)
Object.SetProjectSurfaces(ProjectSurfaces)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectSurfaces(&ProjectSurfaces)	Получить свойство
Object.put_ProjectSurfaces(ProjectSurfaces)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид поверхностей.

ProjectThreads – Передавать в вид обозначения резьб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectThreads = Object.ProjectThreads	Получить свойство (*)
Object.ProjectThreads = ProjectThreads	Установить свойство (*)
ProjectThreads = Object.GetProjectThreads()	Получить свойство (**)
Object.SetProjectThreads(ProjectThreads)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectThreads(&ProjectThreads)	Получить свойство
Object.put_ProjectThreads(ProjectThreads)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений резьб.

ProjectTolerances – Передавать в вид обозначения допусков форм

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectTolerances = Object.ProjectTolerances	Получить свойство (*)
Object.ProjectTolerances = ProjectTolerances	Установить свойство (*)
ProjectTolerances = Object.GetProjectTolerances()	Получить свойство (**)
Object.SetProjectTolerances(ProjectTolerances)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectTolerances(&ProjectTolerances)	Получить свойство
Object.put_ProjectTolerances(ProjectTolerances)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак передачи в вид обозначений допусков форм.

Интерфейс IView1

Дополнительный интерфейс для вида.

Иерархия:

IDispatch

IView1

Примечание:

Дополнительный интерфейс для графических объектов. Данный интерфейс можно получить у интерфейса вида IView посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IView1 – свойства

BaseObject – Опорный объект

Интерфейс...

Тип данных: Указатель на интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Примечание:

В зависимости от типа ассоциативного вида нужно указать вспомогательный опорный объект.

Опорный объект требуется при создании следующих видов:

- Вид по стрелке - требуется объект стрелка взгляда.
- Выносной вид - требуется объект обозначение выносного элемента.
- Вид разреза/сечения - требуется объект линия разреза.

Crossed – Признак необходимости перестроения вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Crossed = Object.Crossed	Установить свойство (*)
Crossed = Object.GetCrossed()	Установить свойство (**)

Синтаксис COM:

Object.get_Crossed(&Crossed)	Установить свойство.

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

CrossedTitle – Признак необходимости перестроения заголовка вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CrossedTitle = Object.CrossedTitle	Установить свойство (*)
CrossedTitle = Object.GetCrossedTitle()	Установить свойство (**)

Синтаксис COM:

Object.get_CrossedTitle(&CrossedTitle)	Установить свойство.

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

EditMacroVisibleRegime – Находится ли вид в режиме редактирования макроэлемента

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EditMacroVisibleRegime = Object.EditMacroVisibleRegime	Получить свойство (*)
EditMacroVisibleRegime = Object.GetEditMacroVisibleRegime()	Получить свойство (**)

Синтаксис COM:

Object.get_EditMacroVisibleRegime(&EditMacroVisibleRegime)	Получить свойство

Примечание:

Свойство используется при редактировании макроэлементов.

LocalCoordinateSystems2D – Получить коллекцию ЛСК вида

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystems2D

Синтаксис Automation:

LocalCoordinateSystems2D = Object.LocalCoordinateSystems2D	Получить свойство (*)
LocalCoordinateSystems2D = Object.GetLocalCoordinateSystems2D()	Получить свойство (**)

Синтаксис COM:

Object.get_LocalCoordinateSystems2D (&LocalCoordinateSystems2D)	Получить свойство

Свойство позволяет получить коллекцию ЛСК вида.

Numerator – Числитель масштаба

Интерфейс...

Тип данных: double

Синтаксис Automation:

Numerator = Object.Numerator	Получить свойство (*)
Object.Numerator = Numerator	Установить свойство (*)
Numerator = Object.GetNumerator()	Получить свойство (**)
Object.SetNumerator(Numerator)	Установить свойство (**)

Синтаксис COM:

Object.get_Numerator(&Numerator)	Получить свойство
Object.put_Numerator(Numerator)	Установить свойство

Printable – Признак разрешения/запрещения печати вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Printable = Object.Printable	Получить свойство (*)
Object.Printable = Printable	Установить свойство (*)
Printable = Object.GetPrintable()	Получить свойство (**)
Object.SetPrintable(Printable)	Установить свойство (**)

Синтаксис COM:

Object.get_Printable(&Printable)	Получить свойство
Object.put_Printable(Printable)	Установить свойство

IView1 – методы

FindObject – Найти объект, ближайший к заданной точке, удовлетворяющий параметрам поиска

Интерфейс...

Синтаксис Automation:

LPDISPATCH FindObject(double X, double Y, double Limit, IFindObjectParameters * Param);

Синтаксис COM:

HRESULT FindObject(double X, double Y, double Limit, IFindObjectParameters * Param, IDrawingObject * * Result);

Возвращаемое значение:

указатель на интерфейс IDrawingObject.

Входные параметры:

X	- первая координата точки,
Y	- вторая координата точки,
Limit	- предел поиска и найденное расстояние,
Param	- параметры поиска объектов IFindObjectParameters.

Примечание:

Для поиска объектов в точке могут быть заданы расширенные параметры поиска.

Интерфейс расширенных параметров поиска IFindObjectParameters можно получить с помощью метода IKompasDocument1::GetInterface;

в 2D документах при использовании константы ksObjectFindObjectParameters.

Если расширенные параметры поиска не требуются, необходимо передать NULL в функцию.

FindObjects – Найти объекты в заданной точке, удовлетворяющие параметрам поиска

Интерфейс...

Синтаксис Automation:

VARIANT FindObjects(double X, double Y, double Limit, IFindObjectParameters * Param);

Синтаксис COM :

HRESULT FindObjects(double X, double Y, double Limit, IFindObjectParameters * Param, VARIANT * Result);

Возвращаемое значение:

VARIANT	- объект типа VARIANT на безопасный массив SAFEARRAY: VT_ARRAY|VT_DISPATCH.

Входные параметры:

X	- первая координата точки,
Y	- вторая координата точки,
Limit	- предел поиска и найденное расстояние,
Param	- параметры поиска объектов IFindObjectParameters.

Примечание:

Для поиска объектов в точке могут быть заданы расширенные параметры поиска.

Интерфейс расширенных параметров поиска IFindObjectParameters можно получить с помощью метода IKompasDocument1::GetInterface;

в 2D документах при использовании константы ksObjectFindObjectParameters.

Если расширенные параметры поиска не требуются, необходимо передать NULL в функцию.

SelectObjects – Отобрать объекты рамкой

Интерфейс...

Синтаксис Automation:

VARIANT SelectObjects(ksRegionTypeEnum RegionType, double XMin, double YMin, double XMax, double YMax);

Синтаксис COM :

HRESULT SelectObjects(ksRegionTypeEnum RegionType, double XMin, double YMin, double XMax, double YMax, VARIANT * Result);

Возвращаемое значение:

VARIANT	- объект типа VARIANT на безопасный массив SAFEARRAY: VT_ARRAYIVT_DISPATCH.
RegionType	- тип региона,
XMin	- минимальная граница поиска по x,
YMin	- минимальная граница поиска по y,
XMax	- максимальная граница поиска по x,
YMax	- максимальная граница поиска по y.

Интерфейс IViewDesignation

Справка системы КОМПАС...

KOMPAS.chm::/381_44_5_Oboznachenie_vida.htm

Интерфейс обозначения вида.

Иерархия:

IDispatch

IViewDesignation

Описание:

Позволяет получить и изменить обозначение вида.

Примечание:

Дополнительный интерфейс вида. Данный интерфейс можно получить у вида IView посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif); в визуальном режиме этому интерфейсу соответствует закладка в параметрах вида "Обозначение вида".

Если простой вид, интерфейс позволяет получить или установить ссылку на объект "Стрелка взгляда", "Линия разреза", "Выносной элемент".

С помощью флагов можно регулировать надпись обозначения вида, включать или выключать соответствующие компоненты обозначения.

Для ассоциативных видов изменить ссылку на объект нельзя. Можно только регулировать надпись обозначения вида при помощи флагов.

Чтобы изменения, установленные в интерфейсе, передались в модель, нужно вызвать у вида метод Update();

IViewDesignation – свойства

Designation – Обозначение вида

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Designation = iObject.Designation	Получить свойство (*)
iObject.GetDesignation()	Получить свойство (**)

Синтаксис COM:

iObject->get_Designation (&Designation)	Получить свойство

Примечание:

1. Свойство позволяет получить обозначение вида в виде строки текста.
2. Свойство доступно только для чтения.

DrawingText – Надпись вида

Интерфейс...

Тип данных: Указатель на интерфейс IDrawingObject

Синтаксис Automation:

DrawingText = Object.DrawingText	Получить свойство (*)
DrawingText = Object.GetDrawingText()	Получить свойство (**)

Синтаксис COM:

Object.get_DrawingText(&DrawingText)	Получить свойство

Примечание:

Свойство доступно только для чтения.

RefObject – Ссылка на объект "Стрелка вида", "Линия разреза" или "Выносной элемент"

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

RefObject = iObject.RefObject	Получить свойство (*)
iObject.RefObject = RefObject	Установить свойство (*)
RefObject = iObject.GetRefObject()	Получить свойство (**)
iObject.SetRefObject (RefObject)	Установить свойство (**)

Синтаксис COM:

iObject->get_RefObject (&RefObject)	Получить свойство
iObject->put_RefObject (RefObject)	Установить свойство

Примечание:

Свойство позволяет получить или установить ссылку на объект "Стрелка вида", "Линия разреза", "Выносной элемент".

Наименование из этого объекта используется в формировании обозначения вида. Если вид ассоциативный, свойство работает как readOnly (только для чтения). В настоящее время свойство реализовано через reference на объект.

```
VARIANTpVal;
    pVal.lVal = p;   // reference объекта
    pVal.vt   = VT_I4; // тип long
```

ShowAngle – Показывать "Повернуто на угол"

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowAngle = iObject.ShowAngle	Получить свойство (*)
iObject.ShowAngle = ShowAngle	Установить свойство (*)
ShowAngle = iObject.GetShowAngle()	Получить свойство (**)
iObject.SetShowAngle (ShowAngle)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowAngle (&ShowAngle)	Получить свойство
iObject->put_ShowAngle (ShowAngle)	Установить свойство

Примечание:

Если свойство возвращает TRUE, в обозначении вида включается угол поворота вида.

Если FALSE, угол поворота вида из обозначения выключается.

ShowName – Показывать наименование

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowName = iObject.ShowName	Получить свойство (*)
iObject.ShowName = ShowName	Установить свойство (*)
ShowName = iObject.GetShowName()	Получить свойство (**)
iObject.SetShowName (ShowName)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowName(&ShowName)	Получить свойство
iObject->put_ShowName(ShowName)	Установить свойство

Примечание:

Если свойство возвращает TRUE, в обозначении вида включается наименование из ссылки на объект например (А-А).

Если FALSE, наименование из обозначения выключается.

ShowPage – Показывать лист

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowPage = iObject.ShowPage	Получить свойство (*)
iObject.ShowPage = ShowPage	Установить свойство (*)
ShowPage = iObject.GetShowPage()	Получить свойство (**)
iObject.SetShowPage (ShowPage)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowPage (&ShowPage)	Получить свойство
iObject->put_ShowPage (ShowPage)	Установить свойство

Примечание:

Если свойство возвращает TRUE, в обозначении вида включается номер листа.

Если FALSE, номер листа из обозначения выключается.

ShowScale – Показывать масштаб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowScale = iObject.ShowScale	Получить свойство (*)
iObject.ShowScale = ShowScale	Установить свойство (*)
ShowScale = iObject.GetShowScale()	Получить свойство (**)
iObject.SetShowScale (ShowScale)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowScale(&ShowScale)	Получить свойство
iObject->put_ShowScale(ShowScale)	Установить свойство

Примечание:

Если свойство возвращает TRUE, в обозначении вида включается масштаб.

Если FALSE, масштаб из обозначения выключается.

ShowTurn – Показывать "Повернуто"

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowTurn = iObject.ShowTurn	Получить свойство (*)
iObject.ShowTurn = ShowTurn	Установить свойство (*)
ShowTurn = iObject.GetShowTurn()	Получить свойство (**)
iObject.SetShowTurn (ShowTurn)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowTurn (&ShowTurn)	Получить свойство
iObject->put_ShowTurn (ShowTurn)	Установить свойство

Примечание:

Если свойство возвращает TRUE, в Обозначении вида включается спецзнак "Повернуто".

Если FALSE, спецзнак "Повернуто" из обозначения выключается.

ShowUnfold – Показывать "Развернуто"

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ShowUnfold = iObject.ShowUnfold          Получить свойство (* )
iObject.ShowUnfold = ShowUnfold          Установить свойство (* )
ShowUnfold = iObject.GetShowUnfold()     Получить свойство (**)
iObject.SetShowUnfold (ShowUnfold)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ShowUnfold (&ShowUnfold)    Получить свойство
iObject->put_ShowUnfold (ShowUnfold)     Установить свойство
```

Примечание:

Если свойство возвращает TRUE, в Обозначении вида включается спецзнак "Развернуто".

Если FALSE, спецзнак "Развернуто" из обозначения выключается.

ShowZone – Показывать зону

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ShowZone = iObject.ShowZone              Получить свойство (* )
iObject.ShowZone = ShowZone              Установить свойство (* )
ShowZone                          =      Получить свойство (**)
iObject.GetShowZone()
iObject.SetShowZone (ShowZone)           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ShowZone                    Получить свойство
(&ShowZone)
iObject->put_ShowZone                    Установить свойство
(ShowZone)
```

Примечание:

Если свойство возвращает TRUE, в обозначении вида включается зона.

Если FALSE, зона из обозначения выключается.

Вспомогательные объекты

Интерфейс ILocalCoordinateSystems2D

Справка системы КОМПАС...

kompas.chm::/842_91_2_lok_sist_koord.htm

Коллекция локальных систем координат.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

ILocalCoordinateSystems2D

Интерфейс коллекции локальных систем координат, позволяющий получить доступ к локальным системам координат вида.

Примечание:

Данный интерфейс можно получить с помощью свойства LocalCoordinateSystems2D интерфейса IView1.

ILocalCoordinateSystems2D – свойства

Current – Получить/установить текущую ЛСК

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem2D

Синтаксис Automation:

Current = Object.Current	Получить свойство (*)
Object.Current = Current	Установить свойство (*)
Current = Object.GetCurrent()	Получить свойство (**)
Object.SetCurrent(Current)	Установить свойство (**)

Синтаксис COM:

Object.get_Current(&Current)	Получить свойство
Object.put_Current(Current)	Установить свойство

Свойство позволяет получать и устанавливать текущую ЛСК вида.

Item – Возвращает ЛСК, заданную по индексу или имени

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem2D

Синтаксис Automation:

Item = Object.Item(Index)	Получить свойство (*)
Item = Object.GetItem(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Item(Index, &Item)	Получить свойство

Возвращает ЛСК из коллекции по заданному индексу или имени.

Примечание:

Свойство доступно только для чтения.

ILocalCoordinateSystems2D – методы

Add – Создает новую ЛСК и добавляет её в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILocalCoordinateSystem2D * * Result);

Создает и добавляет новую ЛСК в коллекцию.

Возвращаемое значение:

Указатель на интерфейс добавленной ЛСК.

Интерфейс ILocalCoordinateSystem2D

Справка системы КОМПАС...

`kompas.chm::/842_91_2_lok_sist_koord.htm`

Интерфейс локальной системы координат.

Иерархия:

IDispatch

IKompasAPIObject

ILocalCoordinateSystem2D

Интерфейс локальных систем координат, позволяющий получить доступ к ее свойствам, таким как имя, положение в системе координат вида, обозначения осей координат.

Примечание:

Данный интерфейс можно получить с помощью свойств ItemILocal и CurrentILocal, а также метода AddILocal интерфейса ILocalCoordinateSystems2D.

ILocalCoordinateSystem2D – свойства

Angle – Угол наклона оси X ЛСК

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                    Получить свойство (* )
Object.Angle = Angle                    Установить свойство (* )
Angle = Object.GetAngle()               Получить свойство (**)
Object.SetAngle( Angle )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )              Получить свойство
Object.put_Angle(  Angle )              Установить свойство
```

Свойство позволяет получать и устанавливать угол наклона оси X в системе координат вида.

AxisXLabel – Обозначение оси X ЛСК

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
AxisXLabel = Object.AxisXLabel          Получить свойство (* )
Object.AxisXLabel = AxisXLabel          Установить свойство (* )
AxisXLabel = Object.GetAxisXLabel()     Получить свойство (**)
Object.SetAxisXLabel( AxisXLabel )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AxisXLabel( &AxisXLabel )    Получить свойство
Object.put_AxisXLabel(  AxisXLabel )    Установить свойство
```

Свойство позволяет получать и устанавливать обозначение оси X ЛСК.

AxisYLabel – Обозначение оси Y ЛСК

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
AxisYLabel = Object.AxisYLabel          Получить свойство (* )
Object.AxisYLabel = AxisYLabel          Установить свойство (* )
```

AxisYLabel = Object.GetAxisYLabel() Получить свойство (**)
Object.SetAxisYLabel(AxisYLabel) Установить свойство (**)

Синтаксис COM:

Object.get_AxisYLabel(&AxisYLabel) Получить свойство
Object.put_AxisYLabel(AxisYLabel) Установить свойство

Свойство позволяет получать и устанавливать обозначение оси Y ЛСК.

Name – Название ЛСК

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name Получить свойство (*)
Object.Name = Name Установить свойство (*)
Name = Object.GetName() Получить свойство (**)
Object.SetName(Name) Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name) Получить свойство
Object.put_Name(Name) Установить свойство

Свойство позволяет получать и устанавливать название ЛСК.

X – X-координата начала ЛСК

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X Получить свойство (*)
Object.X = X Установить свойство (*)
X = Object.GetX() Получить свойство (**)
Object.SetX(X) Установить свойство (**)

Синтаксис COM:

Object.get_X(&X) Получить свойство
Object.put_X(X) Установить свойство

Свойство позволяет получать и устанавливать значение x-координаты начала ЛСК в системе координат вида.

Y – Y-координата начала ЛСК

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y = Object.Y                    Получить свойство (* )
Object.Y = Y                    Установить свойство (* )
Y = Object.GetY()               Получить свойство (**)
Object.SetY( Y )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y( &Y )              Получить свойство
Object.put_Y(  Y )              Установить свойство
```

Свойство позволяет получать и устанавливать значение y-координаты начала ЛСК в системе координат вида.

ILocalCoordinateSystem2D – методы

Delete – Удаление ЛСК

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Удаляет ЛСК из коллекции.

Возвращаемое значение:

Возвращает TRUE в случае успешного удаления, FALSE - в случае неудачи.

Вставки видов, фрагментов, OLE объектов

Интерфейс IInsertionsManager

Интерфейс менеджера вставок фрагментов и видов.

Иерархия:

IDispatch

    IInsertionsManager

Описание:

Интерфейс позволяет создавать и получать вставки фрагментов и видов.

Примечание:

Интерфейс является дополнительным к интерфейсу документа IKompasDocument. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IInsertionsManager – свойства

DefinitionsCount – Количество описаний

Интерфейс...

Тип данных: long

Синтаксис Automation:

DefinitionsCount = Object.DefinitionsCount(long type)	Получить свойство (*)
DefinitionsCount = Object.GetDefinitionsCount(long type)	Получить свойство (**)

Синтаксис COM:

Object.get_DefinitionsCount(ksInsertionTypeEnum type, &DefinitionsCount)	Получить свойство

Входные параметры:

type	- тип вставки фрагмента или вида.

Примечание:

1. Свойство позволяет получать количество описаний вставок фрагмента или вида заданного типа.
2. Свойство доступно только для чтения.

InsertionDefinition – Получить описание по имени или индексу

Интерфейс...

Тип данных: указатель на интерфейс IInsertionDefinition

Синтаксис Automation:

InsertionDefinition = Object.InsertionDefinition(long type, VARIANT Index)	Получить свойство (*)

InsertionDefinition = Object.GetInsertionDefinition(long type, VARIANT Index)	Получить свойство (**)

Синтаксис COM:

Object.get_InsertionDefinition(ksInsertionTypeEnum type, VARIANT Index, &InsertionDefinition)	Получить свойство

Входные параметры:

type	- тип вставки фрагмента или вида,
Index	- индекс или имя вставки фрагмента или вида.

Примечание:

1. Свойство позволяет получать описание вставки фрагмента или вида по имени или индексу.
2. Свойство доступно только для чтения.

InsertionDefinitions – Описания фрагментов и видов

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

InsertionDefinitions = Object.InsertionDefinitions	Получить свойство (*)
InsertionDefinitions = Object.GetInsertionDefinitions()	Получить свойство (**)

Синтаксис COM:

Object.get_InsertionDefinitions(&InsertionDefinitions)	Получить свойство

Примечание:

1. Свойство позволяет получать массив описаний фрагментов и видов. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

IInsertionsManager – методы

AddDefinition – Создать новое описание для фрагмента или вида

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddDefinition(long type,
BSTR Name,
BSTR FileName);

Синтаксис COM:

HRESULT AddDefinition(ksInsertionTypeEnum type,
BSTR Name,
BSTR FileName,
IInsertionDefinition ** Result);

Входные параметры:

type	- тип вставки фрагмента или вида,
Name	- имя вставки,
FileName	- имя файла, из которого производится вставка.

Возвращаемое значение:

- Указатель на интерфейс вставки фрагмента или вида IInsertionDefinition.

Примечание:

Метод позволяет создать новое описание вставки фрагмента или вида.

Интерфейс IInsertionObjects

Справка системы КОМПАС...

KOMPAS.chm::/498_Glava_52_Obshchie_svedenija.htm

Коллекция вставок фрагментов и видов других чертежей.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IDrawingObjects
IInsertionObjects

Описание:

Позволяет создавать и получать вставки фрагментов и видов.

Примечание:

Данный интерфейс можно получить у контейнера графических объектов IDrawingContainer::InsertionObjects.

IInsertionObjects – свойства

InsertionObject – Вставка фрагмента, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс вставки фрагмента IInsertionObject

Синтаксис Automation:

BrokenLine = iObject.BrokenLine (Index);	Получить свойство (*)
BrokenLine = iObject.GetBrokenLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_BrokenLine(Index, &BrokenLine)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IInsertionObjects – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IInsertionObject ** Result);

Возвращаемое значение:

- Указатель на интерфейс вставки фрагмента IInsertionObject.

Интерфейс IOleDrawingObjects

Справка системы КОМПАС...

KOMPAS.chm::/OLE_Overview.htm

Интерфейс коллекции OLE-объектов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IOleDrawingObjects

Описание:

Позволяет создавать и получать OLE-объекты.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::OleDrawingObjects.

IOleDrawingObjects – свойства

OleDrawingObject – OLE-объект, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс OLE-объекта IOleDrawingObject

Синтаксис Automation:

OleDrawingObject = iObject.OleDrawingObject (Index);	Получить свойство (*)
OleDrawingObject = iObject.GetOleDrawingObject(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_OleDrawingObject(Index, &OleDrawingObject)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IOleDrawingObjects – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IOleDrawingObject ** Result);

Возвращаемое значение:

- Указатель на интерфейс OLE-объекта IOleDrawingObject.

Интерфейс IInsertionObject

Справка системы КОМПАС...

KOMPAS.chm::/498_Glava_52_Obshchie_svedenija.htm

Базовый интерфейс для вставки фрагментов и видов другого чертежа.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IInsertionObject

IInsertionFragment

IInsertionView

IPropertyKeeper

Описание:

Интерфейс позволяет задавать и получать параметры вставки фрагментов и видов другого чертежа.

Примечание:

1. Интерфейс можно получить у коллекции вставок фрагментов и видов, используя свойство IInsertionObjects::InsertionObject или метод IInsertionObjects::Add.
2. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы:

- IInsertionFragment,
- IInsertionView,
- IPropertyKeeper.

IInsertionObject – свойства

DimensionLineScale – Масштабирование выносных линий

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- выносные линии масштабировать,
FALSE	- не масштабировать.

Синтаксис Automation:

DimensionLineScale = Object.DimensionLineScale	Получить свойство (*)
Object.DimensionLineScale = DimensionLineScale	Установить свойство (*)
DimensionLineScale = Object.GetDimensionLineScale()	Получить свойство (**)
Object.SetDimensionLineScale(DimensionLineScale)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionLineScale(&DimensionLineScale)	Получить свойство
Object.put_DimensionLineScale(DimensionLineScale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак масштабирования выносных линий.

FileName – Имя файла источника

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство

Примечание:

1. Свойство позволяет получать имя файла источника.
2. Свойство доступно только для чтения.

InsertionDefinition – описание вставки фрагмента или вида

Интерфейс...

Тип данных: указатель на интерфейс описания вставки IInsertionDefinition

Синтаксис Automation:

InsertionDefinition = Object.InsertionDefinition	Получить свойство (*)
Object.InsertionDefinition = InsertionDefinition	Установить свойство (*)
InsertionDefinition = Object.GetInsertionDefinition()	Получить свойство (**)
Object.SetInsertionDefinition(InsertionDefinition)	Установить свойство (**)

Синтаксис COM:

Object.get_InsertionDefinition(&InsertionDefinition)	Получить свойство
Object.put_InsertionDefinition(InsertionDefinition)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать интерфейс описания вставки фрагмента или вида.

InsertionType – Способ вставки

Интерфейс...

Тип данных: из перечисления ksInsertionTypeEnum

Синтаксис Automation:

InsertionType = Object.InsertionType	Получить свойство (*)
InsertionType = Object.GetInsertionType()	Получить свойство (**)

Синтаксис COM:

Object.get_InsertionType(&InsertionType)	Получить свойство

Примечание:

1. Свойство позволяет получать способ вставки.
2. Свойство доступно только для чтения.

Name – Имя вставки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя вставки фрагмента.

IInsertionObject – методы

GetPlacement – Получить местоположение объекта

Интерфейс...

Синтаксис Automation:

```
BOOL GetPlacement( double * X,
double * Y,
double * Angle,
BOOL  * MirrorSymmetry )
```

Синтаксис COM:

```
HRESULT GetPlacement( double * X,
double * Y,
double * Angle,
BOOL  * MirrorSymmetry,
BOOL  * Result );
```

Выходные параметры:

X, Y	- координаты объекта,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить местоположение объекта.

SetPlacement – Установить местоположение объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement(double X,
double Y,
double Angle,
BOOL MirrorSymmetry)

Синтаксис COM:

HRESULT SetPlacement(double X,
double Y,
double Angle,
BOOL MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты объекта,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить местоположение объекта.

Интерфейс IInsertionFragment

Справка системы КОМПАС...

KOMPAS.chm::/498_Glava_52_Obshchie_svedenija.htm

Интерфейс вставки фрагментов (локальных и внешних).

Иерархия:

IDispatch
IKompasAPIObject
IDrawingObject
IInsertionObject
IInsertionFragment

Описание:

Интерфейс позволяет задавать и получать параметры вставки локальных и внешних фрагментов.

Примечание:

Дополнительный интерфейс базового интерфейса вставки фрагмента IInsertionObject. Может быть получен с помощью метода IUnknown::QueryInterface.

IInsertionFragment – свойства

BreakObjectsEnabled – Учитывать разрыв вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BreakObjectsEnabled = Object.BreakObjectsEnabled	Получить свойство (*)
Object.BreakObjectsEnabled = BreakObjectsEnabled	Установить свойство (*)
BreakObjectsEnabled = Object.GetBreakObjectsEnabled()	Получить свойство (**)
Object.SetBreakObjectsEnabled(BreakObjectsEnabled)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakObjectsEnabled (&BreakObjectsEnabled)	Получить свойство
Object.put_BreakObjectsEnabled (BreakObjectsEnabled)	Установить свойство

Версия Компас v18.1

CreateSpcObjects – Создавать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateSpcObjects = Object.CreateSpcObjects	Получить свойство (*)
Object.CreateSpcObjects = CreateSpcObjects	Установить свойство (*)
CreateSpcObjects = Object.GetCreateSpcObjects()	Получить свойство (**)

Object.SetCreateSpcObjects(CreateSpcObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateSpcObjects(&CreateSpcObjects)	Получить свойство
Object.put_CreateSpcObjects(CreateSpcObjects)	Установить свойство

Версия Компас v19

MultiLayer – Многослойность

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE	- многослойный объект,
FALSE	- однослойный объект.

Синтаксис Automation:

MultiLayer = Object.MultiLayer	Получить свойство (*)
Object.MultiLayer = MultiLayer	Установить свойство (*)
MultiLayer = Object.GetMultiLayer()	Получить свойство (**)
Object.SetMultiLayer(MultiLayer)	Установить свойство (**)

Синтаксис COM:

Object.get_MultiLayer(&MultiLayer)	Получить свойство
Object.put_MultiLayer(MultiLayer)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак многослойности объекта.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство (*)

Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать масштаб вставляемого фрагмента.

Variable – Получить параметрическую переменную по имени, индексу или указателю на размер

Интерфейс...

Тип данных: указатель на интерфейс параметрической переменной IVariable7

Синтаксис Automation:

Variable = Object.Variable (VARIANT Index)	Получить свойство (*)
Variable = Object.GetVariable(VARIANT Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Variable(Index, &Variable)	VARIANT	Получить свойство

Входные параметры:

Index - индекс, имя или указатель на размер параметрической переменной.

Примечание:

1. Свойство позволяет получать интерфейс параметрической переменной.
2. Свойство доступно только для чтения.

Variables – Получить массив параметрических переменных

Интерфейс...

Тип данных: массив SAFEARRAY VT_DISPATCH

Синтаксис Automation:

Variables = Object.Variables() Variables = Object.GetVariables()	Получить свойство (*) Получить свойство (**)

Синтаксис COM:

Object.get_Variables(&Variables)	Получить свойство

Примечание:

1. Свойство позволяет получать массив параметрических переменных.
2. Свойство доступно только для чтения.

VariablesCount – Получить количество параметрических переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

VariablesCount = Object.VariablesCount() VariablesCount = Object.GetVariablesCount()	Получить свойство (*) Получить свойство (**)

Синтаксис COM:

Object.get_VariablesCount(&VariablesCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество параметрических переменных.
2. Свойство доступно только для чтения.

VariableTable – Таблица переменных

Интерфейс...

Тип данных: указатель на интерфейс таблицы переменных IVariableTable

Синтаксис Automation:

VariableTable = Object.VariableTable() VariableTable = Object.GetVariableTable()	Получить свойство (*) Получить свойство (**)

Синтаксис COM:

Object.get_VariableTable(&VariableTable)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс таблицы переменных.
2. Свойство доступно только для чтения.

Интерфейс IInsertionView

Справка системы КОМПАС...

KOMPAS.chm::/498_Glava_52_Obshchie_svedenija.htm

Интерфейс вставки вида из другого чертежа.

Иерархия:

IDispatch
IKompasAPIObject
IDrawingObject
IInsertionObject
IInsertionView

Описание:

Интерфейс позволяет задавать и получать параметры вставки вида из другого чертежа.

Примечание:

Дополнительный интерфейс базового интерфейса вставки фрагмента IInsertionObject. Может быть получен с помощью метода IUnknown::QueryInterface.

IInsertionView – свойства

CreateSpcObjects – Создавать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateSpcObjects = Object.CreateSpcObjects	Получить свойство (*)
Object.CreateSpcObjects = CreateSpcObjects	Установить свойство (*)
CreateSpcObjects = Object.GetCreateSpcObjects()	Получить свойство (**)
Object.SetCreateSpcObjects(CreateSpcObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateSpcObjects(&CreateSpcObjects)	Получить свойство

Object.put_CreateSpcObjects(CreateSpcObjects)	Установить свойство

OwnerDocumentParams – Параметры документа

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- отображать с параметрами документа владельца,
FALSE	- отображать с параметрами документа источника.

Синтаксис Automation:

OwnerDocumentParams = Object.OwnerDocumentParams	Получить свойство (*)
Object.OwnerDocumentParams = OwnerDocumentParams	Установить свойство (*)
OwnerDocumentParams = Object.GetOwnerDocumentParams()	Получить свойство (**)
Object.SetOwnerDocumentParams(OwnerDocumentParams)	Установить свойство (**)

Синтаксис COM:

Object.get_OwnerDocumentParams(&OwnerDocumentParams)	Получить свойство
Object.put_OwnerDocumentParams(OwnerDocumentParams)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ отображения параметров.

ThinLines – В тонких линиях

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ThinLines = Object.ThinLines	Получить свойство (*)
Object.ThinLines = ThinLines	Установить свойство (*)
ThinLines = Object.GetThinLines()	Получить свойство (**)
Object.SetThinLines(ThinLines)	Установить свойство (**)

Синтаксис COM:

Object.get_ThinLines(&ThinLines)	Получить свойство
Object.put_ThinLines(ThinLines)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак отрисовки в тонких линиях.

ViewName – Имя вида

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ViewName = Object.ViewName	Получить свойство (*)
ViewName = Object.GetViewName()	Получить свойство (**)

Синтаксис COM:

Object.get_ViewName(&ViewName)	Получить свойство

Примечание:

1. Свойство позволяет получать имя вида.
2. Свойство доступно только для чтения.

ViewNumber – Номер вида

Интерфейс...

Тип данных: long

Синтаксис Automation:

ViewNumber = Object.ViewNumber	Получить свойство (*)
Object.ViewNumber = ViewNumber	Установить свойство (*)
ViewNumber = Object.GetViewNumber()	Получить свойство (**)
Object.SetViewNumber(ViewNumber)	Установить свойство (**)

Синтаксис COM:

Object.get_ViewNumber(&ViewNumber)	Получить свойство
Object.put_ViewNumber(ViewNumber)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать номер вида.

Интерфейс IOleDrawingObject

Справка системы КОМПАС...

KOMPAS.chm::/353_Glava40_Ispolqzovanie_rastr.htm

Интерфейс OLE-объекта.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IOleDrawingObject

Описание:

Интерфейс позволяет задавать параметры OLE-объекта.

Примечание:

1. Интерфейс можно получить у коллекции OLE-объектов, используя свойство IOleDrawingObjects::OleDrawingObject или метод IOleDrawingObjects::Add.
2. После задания параметров OLE-объекта требуется вызвать метод IDrawingObject::Update.
3. OLE-объект можно создать двумя способами:

- внедрить телом в документ (для этого требуется задать свойство IOleDrawingObject::ClassId),
- вставить по ссылке из файла (для этого требуется задать свойство IOleDrawingObject::FileName).

IOleDrawingObject – свойства

ClassId – Идентификатор объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ClassId = Object.ClassId	Получить свойство (*)
Object.ClassId = ClassId	Установить свойство (*)
ClassId = Object.GetClassId()	Получить свойство (**)
Object.SetClassId(ClassId)	Установить свойство (**)

Синтаксис COM:

Object.get_ClassId(&ClassId)	Получить свойство
Object.put_ClassId(ClassId)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать идентификатор OLE-объекта.
2. Значение свойства нельзя изменить после создания OLE-объекта.
3. Если задано ClassId, объект вставляется телом, тип объекта определяется по ClassId.

FileName – Имя файла объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя файла объекта.

InsertionType – Способ вставки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- в виде значка,
FALSE	- в виде содержимого.

Синтаксис Automation:

InsertionType = Object.InsertionType	Получить свойство (*)
Object.InsertionType = InsertionType	Установить свойство (*)
InsertionType = Object.GetInsertionType()	Получить свойство (**)
Object.SetInsertionType(InsertionType)	Установить свойство (**)

Синтаксис COM:

Object.get_InsertionType(&InsertionType)	Получить свойство

Object.put_InsertionType(InsertionType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ вставки OLE-объекта.

Link – Связать файл и объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Link = Object.Link	Получить свойство (*)
Object.Link = Link	Установить свойство (*)
Link = Object.GetLink()	Получить свойство (**)
Object.SetLink(Link)	Установить свойство (**)

Синтаксис COM:

Object.get_Link(&Link)	Получить свойство
Object.put_Link(Link)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак связи файла и OLE-объекта.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство (*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать масштаб OLE-объекта.

IOleDrawingObject – методы

Close – Завершить редактирование

Интерфейс...

Синтаксис Automation:

BOOL Close(BOOL Save);

Синтаксис COM:

HRESULT Close(BOOL Save, BOOL * Result);

Входные параметры:

Save	- сохранить изменения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DoVerb – Запустить редактирование вставки OLE-объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH DoVerb(long iVerb);

Синтаксис COM:

HRESULT DoVerb(long iVerb, IDispatch ** Result);

Входные параметры:

iVerb	- действие, которое должно быть выполнено над OLE-объектом.

Возвращаемое значение:

- Указатель на интерфейс OLE-объекта.

GetPlacement – Получить местоположение объекта

Интерфейс...

Синтаксис Automation:

BOOL GetPlacement(double * X,
double * Y,

double * Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

HRESULT GetPlacement(double * X,
double * Y,
double * Angle,
BOOL * MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить точку привязки и угол поворота растрового объекта.

SetPlacement – Установить местоположение объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement(double X,
double Y,
double Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

HRESULT SetPlacement(double X,
double Y,
double Angle,
BOOL * MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить точку привязки и угол поворота растрового объекта.

Интерфейс IInsertionDefinition

Справка системы КОМПАС...

KOMPAS.chm::/498_Glava_52_Obshchie_svedenija.htm

Описание вставки фрагмента или вида.

Иерархия:

IDispatch

IKompasAPIObject

IInsertionDefinition

Описание:

Интерфейс позволяет задавать и получать параметры описания вставки фрагмента или вида.

Примечание:

Данный интерфейс можно получить следующими способами:

- с помощью метода IInsertionObject::InsertionDefinition.
- с помощью метода IInsertionsManager::InsertionDefinition.

IInsertionDefinition – свойства

FileName – Имя файла источника

Интерфейс..

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя файла источника.

InsertionObjectsCount – Количество вставок

Интерфейс...

Тип данных: long

Синтаксис Automation:

InsertionObjectsCount = Object.InsertionObjectsCount()	Получить свойство (*)
InsertionObjectsCount = Object.GetInsertionObjectsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_InsertionObjectsCount(&InsertionObjectsCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество вставок.
2. Свойство доступно только для чтения.

InsertionType – Способ вставки

Интерфейс..

Тип данных: из перечисления ksInsertionTypeEnum

Синтаксис Automation:

InsertionType = Object.InsertionType()	Получить свойство (*)
InsertionType = Object.GetInsertionType()	Получить свойство (**)

Синтаксис COM:

Object.get_InsertionType(&InsertionType)	Получить свойство

Примечание:

1. Свойство позволяет получать способ вставки фрагмента.
2. Свойство доступно только для чтения.

Name – Имя описания

Интерфейс..

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя описания вставки.

IInsertionDefinition - методы

Open - Открыть документ-источник на редактирование

Интерфейс..

Синтаксис Automation:

LPDISPATCH Open(BOOL Visible,
BOOL ReadOnly);

Синтаксис COM:

HRESULT Open(BOOL Visible,
BOOL ReadOnly,
IKompasDocument2D ** Result);

Выходные параметры:

Visible	- открыть в видимом режиме (по умолчанию - TRUE),
ReadOnly	- открыть только для чтения (по умолчанию - FALSE).

Возвращаемое значение:

- указатель на интерфейс документа IKompasDocument2D.

Примечание:

Метод позволяет открыть документ-источник на редактирование.

Интерфейс IInsertionParameters

Справка системы КОМПАС...

kompas.chm::/540_65_9_8_Illjustracii.htm

Параметры вставки фрагмента и растрового объекта в ячейку таблицы.

Иерархия:

IKompasAPIObject

IInsertionParameters

Описание:

Интерфейс позволяет получить и изменить параметры вставки фрагмента и растрового объекта в текст ячейки таблицы.

Интерфейс можно получить с помощью свойства ITextLine::TextLineData.

Примечание:

Чтобы вставить фрагмент, свойство ITextLine::TextLineType должно иметь значение ksTLFragment, чтобы вставить растровый объект - ksTLRaster.

IInsertionParameters – свойства

Angle – Угол

Интерфейс...

Тип данных: ksAngleEnum

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Свойство позволяет устанавливать и получать угол наклона вставки к оси OX.

FileName – Имя файла-источника

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство

Свойство позволяет получить имя файла-источника.

Примечание:

Свойство доступно только для чтения.

Height – Высота вставки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство (*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight()	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height)	Получить свойство
Object.put_Height(Height)	Установить свойство

Свойство позволяет устанавливать и получать высоту вставки.

ImageResolution – Разрешение изображения для вставки растра

Интерфейс...

Тип данных: double

Синтаксис Automation:

ImageResolution = Object.ImageResolution	Получить свойство (*)
Object.ImageResolution = ImageResolution	Установить свойство (*)
ImageResolution = Object.GetImageResolution()	Получить свойство (**)
Object.SetImageResolution(ImageResolution)	Установить свойство (**)

Синтаксис COM:

Object.get_ImageResolution(&ImageResolution)	Получить свойство
Object.put_ImageResolution(ImageResolution)	Установить свойство

Свойство позволяет устанавливать и получать разрешение изображения для вставки растра.

InsertionDefinition – описание вставки фрагмента

Интерфейс...

Тип данных: указатель на интерфейс IInsertionDefinition

Синтаксис Automation:

InsertionDefinition = Object.InsertionDefinition	Получить свойство (*)
Object.InsertionDefinition = InsertionDefinition	Установить свойство (*)
InsertionDefinition = Object.GetInsertionDefinition()	Получить свойство (**)
Object.SetInsertionDefinition(InsertionDefinition)	Установить свойство (**)

Синтаксис COM:

Object.get_InsertionDefinition(&InsertionDefinition)	Получить свойство
Object.put_InsertionDefinition(InsertionDefinition)	Установить свойство

Свойство позволяет устанавливать и получать интерфейс описания вставки фрагмента.

Palette – Цветовая палитра (количество разрядов)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Palette = Object.Palette	Получить свойство (*)
Object.Palette = Palette	Установить свойство (*)
Palette = Object.GetPalette()	Получить свойство (**)
Object.SetPalette(Palette)	Установить свойство (**)

Синтаксис COM:

Object.get_Palette(&Palette)	Получить свойство
Object.put_Palette(Palette)	Установить свойство

Свойство позволяет устанавливать и получать количество разрядов цветовой палитры.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство (*)
Object.Scale = Scale	Установить свойство (*)

```
Scale = Object.GetScale()          Получить свойство (**)
Object.SetScale( Scale )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Scale( &Scale )         Получить свойство
Object.put_Scale(  Scale )         Установить свойство
```

Свойство позволяет устанавливать и получать масштаб вставки.

SourceHeight – Высота источника

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
SourceHeight = Object.SourceHeight          Получить свойство (* )
SourceHeight = Object.GetSourceHeight()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SourceHeight( &SourceHeight )    Получить свойство
```

Свойство позволяет получить высоту источника.

Примечание:

Свойство доступно только для чтения.

SourceWidth – Ширина источника

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
SourceWidth = Object.SourceWidth            Получить свойство (* )
SourceWidth = Object.GetSourceWidth()       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SourceWidth( &SourceWidth )      Получить свойство
```

Свойство позволяет получить ширину источника.

Примечание:

Свойство доступно только для чтения.

Width – Ширина вставки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Width = Object.Width	Получить свойство (*)
Object.Width = Width	Установить свойство (*)
Width = Object.GetWidth()	Получить свойство (**)
Object.SetWidth(Width)	Установить свойство (**)

Синтаксис COM:

Object.get_Width(&Width)	Получить свойство
Object.put_Width(Width)	Установить свойство

Свойство позволяет устанавливать и получать ширину вставки.

IInsertionParameters – методы

AutoScale – Подобрать масштаб

Интерфейс...

Синтаксис Automation:

BOOL AutoScale(BOOL ByHeight);

Синтаксис COM:

HRESULT AutoScale(BOOL ByHeight, BOOL * Res)

Входные параметры:

ByHeight	- TRUE - по высоте,
	- FALSE по ширине.

Возвращаемое значение:

TRUE	- в случае удачи.

Геометрия

Интерфейс IDrawingContainer

Интерфейс контейнера объектов вида графического документа.

Иерархия:

IDispatch

IDrawingContainer

Описание:

Позволяет получить коллекции объектов, которые входят в состав вида графического документа или его объекта.

Примечание:

Дополнительный интерфейс вида. Данный интерфейс можно получить у вида IView посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IDrawingContainer – свойства

Arcs – Коллекция дуг

Интерфейс...

Тип данных: указатель на интерфейс коллекции дуг IArcs

Синтаксис Automation:

```
IArcs = iObject.Arcs;          Получить свойство(* )
IArcs = iObject.GetArcs();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Arcs(&Arcs);     Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Beziers – Коллекция Bezier-сплайнов

Интерфейс...

Тип данных: указатель на интерфейс IBeziers

Синтаксис Automation:

```
Beziers = Object.Beziers      Получить свойство(* )
Beziers =                     Получить свойство (**)
Object.GetBeziers()
```

Синтаксис COM:

```
Object.get_Beziers(           Получить свойство
&Beziers )
```

Свойство позволяет получать интерфейс коллекции Bezier-сплайнов.

Примечание:

Свойство доступно только для чтения.

Circles – Коллекция окружностей

Интерфейс...

Тип данных: указатель на интерфейс ICircles

Синтаксис Automation:

Circles = Object.Circles	Получить свойство(*)
Circles = Object.GetCircles()	Получить свойство (**)

Синтаксис COM:

Object.get_Circles(&Circles)	Получить свойство

Свойство позволяет получать интерфейс коллекции окружностей.

Примечание:

Свойство доступно только для чтения.

Colourings – Коллекция заливок

Интерфейс...

Тип данных: указатель на интерфейс IColourings

Синтаксис Automation:

Colourings = Object.Colourings	Получить свойство(*)
Colourings = Object.GetColourings()	Получить свойство (**)

Синтаксис COM:

Object.get_Colourings(&Colourings)	Получить свойство

Свойство позволяет получать интерфейс коллекции заливок.

Примечание:

Свойство доступно только для чтения.

ConicCurves – Коллекция конических кривых

Интерфейс...

Тип данных: Указатель на интерфейс IConicCurves.

Синтаксис Automation:

ConicCurves = Object.ConicCurves	Получить свойство(*)

ConicCurves = Object.GetConicCurves()	Получить свойство (**)

Синтаксис COM:

Object.get_ConicCurves(&ConicCurves)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

DrawingContours – Коллекция контуров

Интерфейс...

Тип данных: указатель на интерфейс коллекции контуров IDrawingContours

Синтаксис Automation:

IDrawingContours = iObject.DrawingContours;	Получить свойство(*)
IDrawingContours = iObject.GetDrawingContours();	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingContours(&DrawingContours);	Получить свойство

Примечание:

Свойство доступно только для чтения.

DrawingTexts – Коллекция текстов на чертеже

Интерфейс...

Тип данных: указатель на интерфейс коллекции текстов на чертеже IDrawingTexts.

Синтаксис Automation:

DrawingTexts = iObject.DrawingTexts	Получить свойство(*)
DrawingTexts = iObject.GetDrawingTexts()	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingTexts (&DrawingTexts)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Ellipses – Коллекция эллипсов

Интерфейс...

Тип данных: указатель на интерфейс IEllipses

Синтаксис Automation:

Ellipses = Object.Ellipses	Получить свойство(*)
Ellipses = Object.GetEllipses()	Получить свойство (**)

Синтаксис COM:

Object.get_Ellipses(&Ellipses)	Получить свойство

Свойство позволяет получать интерфейс коллекции эллипсов.

Примечание:

Свойство доступно только для чтения.

EllipseArcs – Коллекция дуг эллипсов

Интерфейс...

Тип данных: указатель на интерфейс IEllipseArcs

Синтаксис Automation:

EllipseArcs = Object.EllipseArcs	Получить свойство(*)
EllipseArcs = Object.GetEllipseArcs()	Получить свойство (**)

Синтаксис COM:

Object.get_EllipseArcs(&EllipseArcs)	Получить свойство

Свойство позволяет получать интерфейс коллекции дуг эллипсов.

Примечание:

Свойство доступно только для чтения.

Equidistants – Коллекция эквидистант

Интерфейс...

Тип данных: указатель на интерфейс IEquidistants

Синтаксис Automation:

Equidistants = Object.Equidistants	Получить свойство(*)
Equidistants = Object.GetEquidistants()	Получить свойство (**)

Синтаксис COM:

Object.get_Equidistants(&Equidistants)	Получить свойство

Свойство позволяет получать интерфейс коллекции эквидистант.

Примечание:

Свойство доступно только для чтения.

Hatches – Коллекция штриховок

Интерфейс...

Тип данных: указатель на интерфейс IHatches

Синтаксис Automation:

Hatches = Object.Hatches	Получить свойство(*)
Hatches = Object.GetHatches()	Получить свойство (**)

Синтаксис COM:

Object.get_Hatches(&Hatches)	Получить свойство

Свойство позволяет получать интерфейс коллекции штриховок.

Примечание:

Свойство доступно только для чтения.

InsertionObjects – Коллекция вставок фрагментов и видов другого чертежа

Интерфейс...

Тип данных: указатель на интерфейс IInsertionObjects

Синтаксис Automation:

```
InsertionObjects =                Получить свойство(* )
Object.InsertionObjects
InsertionObjects =                Получить свойство (**)
Object.GetInsertionObjects
()
```

Синтаксис COM:

```
Object.get_InsertionObject    Получить свойство
s( &InsertionObjects )
```

Свойство позволяет получать интерфейс коллекции вставок фрагментов и видов другого чертежа.

Примечание:

Свойство доступно только для чтения.

Lines – Коллекция линий

Интерфейс...

Тип данных: указатель на интерфейс ILines

Синтаксис Automation:

```
Lines = Object.Lines          Получить свойство(* )
Lines = Object.GetLines()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Lines( &Lines )    Получить свойство
```

Примечание:

1. Свойство позволяет получать интерфейс коллекции коллекции линий.
2. Свойство доступно только для чтения.

LineSegments – Коллекция отрезков

Интерфейс...

Тип данных: указатель на интерфейс коллекции отрезков в документе ILineSegments

Синтаксис Automation:

```
LineSegments =                Получить свойство(* )
iObject.LineSegments
LineSegments =                Получить свойство (**)
iObject.GetLineSegments()
```

Синтаксис COM:

iObject->get_LineSegments(&LineSegments)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MacroObjects – Коллекция макроэлементов

Интерфейс...

Тип данных: указатель на интерфейс IMacroObjects

Синтаксис Automation:

MacroObjects = Object.MacroObjects	Получить свойство(*)
MacroObjects = Object.GetMacroObjects()	Получить свойство (**)

Синтаксис COM:

Object.get_MacroObjects(&MacroObjects)	Получить свойство

Свойство позволяет получать интерфейс коллекции макроэлементов.

Примечание:

Свойство доступно только для чтения.

Multilines – Коллекция мультилиний

Интерфейс...

Тип данных: указатель на интерфейс коллекции дуг IMultilines

Синтаксис Automation:

IMultilines = iObject.Multilines;	Получить свойство(*)
IMultilines = iObject.GetMultilines();	Получить свойство (**)

Синтаксис COM:

iObject->get_Multilines(&Multilines);	Получить свойство

Примечание:

Свойство доступно только для чтения.

Nurbses – Коллекция Nurbs-сплайнов

Интерфейс...

Тип данных: указатель на интерфейс INurbses

Синтаксис Automation:

Nurbses = Object.Nurbses	Получить свойство(*)
Nurbses = Object.GetNurbses	Получить свойство (**)

Синтаксис COM:

Object.get_Nurbses(&Nurbses)	Получить свойство

Свойство позволяет получать интерфейс коллекции Nurbs-сплайнов.

Примечание:

Свойство доступно только для чтения.

NurbsesByPoints – Получить коллекцию NURBS-кривых по точкам

Интерфейс...

Тип данных: указатель на интерфейс INurbses

Синтаксис Automation:

NurbsesByPoints = Object.NurbsesByPoints	Получить свойство(*)
NurbsesByPoints = Object.GetNurbsesByPoints	Получить свойство (**)

Синтаксис COM:

Object.get_NurbsesByPoints(&NurbsesByPoints)	Получить свойство

Свойство позволяет получать интерфейс коллекции Nurbs-сплайнов.

Примечание:

Свойство доступно только для чтения.

Objects – Массив SAFEARRAY объектов, входящих в состав данного объекта вида

Интерфейс...

Массив SAFEARRAY объектов, входящих в состав данного объекта вида.

Тип данных: VARIANT

Синтаксис Automation:

Objects = iObject.Objects(ObjType)	Получить свойство(*)
Objects = iObject.GetObjects(ObjType)	Получить свойство (**)

Синтаксис COM:

iObject->get_Objects(&Objects(ObjType))	Получить свойство

Входные параметры:

ObjType	Тип объекта VARIANT.

Примечание:

Типы объектов могут быть из перечисления DrawingObjectTypeEnum. Если Null – все объекты.

OleDrawingObjects – Коллекция OLE-объектов

Интерфейс...

Тип данных: указатель на интерфейс IOleDrawingObjects

Синтаксис Automation:

OleDrawingObjects = Object.OleDrawingObjects	Получить свойство(*)
OleDrawingObjects = Object.GetOleDrawingObjects()	Получить свойство (**)

Синтаксис COM:

Object.get_OleDrawingObjects(&OleDrawingObjects)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс коллекции OLE-объектов.
2. Свойство доступно только для чтения.

Points – Коллекция точек

Интерфейс...

Тип данных: указатель на интерфейс IPoints

Синтаксис Automation:

Points = Object.Points	Получить свойство(*)
Points = Object.GetPoints()	Получить свойство (**)

Синтаксис COM:

Object.get_Points(&Points)	Получить свойство

Свойство позволяет получать интерфейс коллекции точек.

Примечание:

Свойство доступно только для чтения.

PolyLines2D – Коллекция 2D-ломаных

Интерфейс...

Тип данных: указатель на интерфейс IPolyLines2D

Синтаксис Automation:

PolyLines2D = Object.PolyLines2D	Получить свойство(*)
PolyLines2D = Object.GetPolyLines2D()	Получить свойство (**)

Синтаксис COM:

Object.get_PolyLines2D(&PolyLines2D)	Получить свойство

Свойство позволяет получать интерфейс коллекции 2D-ломаных.

Примечание:

Свойство доступно только для чтения.

Rasters – Коллекция растровых объектов

Интерфейс...

Тип данных: указатель на интерфейс IRasters

Синтаксис Automation:

Rasters = Object.Rasters	Получить свойство(*)
Rasters = Object.GetRasters()	Получить свойство (**)

Синтаксис COM:

Object.get_Rasters(&Rasters)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс коллекции растровых объектов.
2. Свойство доступно только для чтения.

Rectangles – Коллекция прямоугольников

Интерфейс...

Тип данных: указатель на интерфейс IRectangles

Синтаксис Automation:

Rectangles = Object.Rectangles	Получить свойство(*)
Rectangles = Object.GetRectangles()	Получить свойство (**)

Синтаксис COM:

Object.get_Rectangles(&Rectangles)	Получить свойство

Свойство позволяет получать интерфейс коллекции прямоугольников.

Примечание:

Свойство доступно только для чтения.

RegularPolygons – Коллекция многоугольников

Интерфейс...

Тип данных: указатель на интерфейс IRegularPolygons

Синтаксис Automation:

RegularPolygons = Object.RegularPolygons	Получить свойство(*)
RegularPolygons = Object.GetRegularPolygons()	Получить свойство (**)

Синтаксис COM:

Object.get_RegularPolygons(&RegularPolygons)	Получить свойство

Свойство позволяет получать интерфейс коллекции многоугольников.

Примечание:

Свойство доступно только для чтения.

Интерфейс IBoundariesObject

Интерфейс объекта с границами.

Иерархия:

IDispatch

IBoundariesObject

Описание:

Позволяет редактировать границы объекта. Дополнительный интерфейс штриховки IHatch, заливки IColouring и растрового объекта IRaster.

Примечание:

Данный интерфейс можно получить с помощью метода IUnknown::QueryInterface.

IBoundariesObject – свойства

Boundaries – Получить копию границ

Интерфейс...

Тип данных: массив SAFEARRAY | VT_DISPATCH

Синтаксис Automation:

Boundaries = Object.Boundaries	Получить свойство(*)
Boundaries = Object.GetBoundaries()	Получить свойство (**)

Синтаксис COM:

Object.get_Boundaries(&Boundaries)	Получить свойство

Примечание:

1. Свойство позволяет получать копию границ объекта.
2. Свойство доступно только для чтения.

IBoundariesObject – методы

AddBoundaries – Добавить объекты в границу

Интерфейс...

Синтаксис Automation:

BOOL AddBoundaries(VARIANT Objects,
BOOL DeleteSource);

Синтаксис COM:

HRESULT AddBoundaries(VARIANT Objects,
BOOL DeleteSource,
BOOL * Result);

Входные параметры:

Objects	- массив SAFEARRAY I VT_DISPATCH объектов, добавляемых в границу,
DeleteSource	- TRUE - удалить исходные объекты, - FALSE - копировать исходные объекты.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить объекты в границу.

Clear – Удалить все границы

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить границы.

Интерфейс IArcs

Справка системы КОМПАС...

`KOMPAS.chm::/1436_Postroenie_dug.htm`

Интерфейс коллекции дуг.

Иерархия:

IKompasAPIObject
IKompasCollection
IDrawingObjects
IArcs

Описание:

Интерфейс позволяет получить коллекции дуг.

Примечание:

Получить интерфейс коллекции дуг можно, используя метод контейнера объектов IDrawingContainer::Arcs.

IArcs – свойства

Arc – Дуга, заданная по индексу

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1436_Postroenie_dug.htm

Тип данных: указатель на интерфейс дуги IArc.

Синтаксис Automation:

Arc = iObject.Arc(Index);	Получить свойство(*)
Arc = iObject.GetArc(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Arc(Index, &Arc)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс отрезка в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса могут использоваться следующие типы:

- Индекс объекта в коллекции,
- Ссылка на объект (reference),

IArcs – методы

Add – Создать дугу (добавить дугу в коллекцию

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1436_Postroenie_dug.htm

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add([out, retval] IArc** Result);

Возвращаемое значение:

Result	- Указатель на базовый интерфейс для дуг IArc.

Примечание:

Метод позволяет создать новый интерфейс дуги. После получения нового интерфейса нужно задать параметры дуги и вызвать метод IDrawingObject::Update.

Интерфейс IBeziers

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#BEZIER

Коллекция кривых Безье.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBeziers

Описание:

Позволяет создавать и получать кривые Безье на чертеже.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Beziers.

IBeziers – свойства

Bezier – Кривая Безье, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс кривой Безье на чертеже Bezier

Синтаксис Automation:

Bezier = iObject.Bezier(Index);	Получить свойство(*)
Bezier = iObject.GetBezier(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Bezier(Index, &Bezier) — Получить свойство

Примечание:

Свойство доступно только для чтения.

IBeziers – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IBezier ** Result);

Возвращаемое значение:

- Указатель на интерфейс кривой Безье IBezier.

Интерфейс ICircles

Справка системы КОМПАС...

KOMPAS.chm::/1432_Okruznosti.htm#okr_po_centru_i_tochke

Коллекция окружностей.

Иерархия:

IKompasAPIObject
    IKompasCollection
        IDrawingObjects
            ICircles

Описание:

Интерфейс позволяет создавать и получать окружности на чертеже.

Примечание:

Получить интерфейс можно используя метод контейнера графических объектов IDrawingContainer::Circles.

ICircles – свойства

Circle – Окружность, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс окружности на чертеже ICircle

Синтаксис Automation:

Circle = iObject.Circle(Index);	Получить свойство(*)
Circle = iObject.GetCircle(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Circle(&Circle)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

ICircles – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICircle ** Result);

Возвращаемое значение:

- Указатель на интерфейс окружности на чертеже ICircle.

Интерфейс IColourings

Справка системы КОМПАС...

KOMPAS.chm::/220_Glava_26_Shtrihovka_i_zalivka.htm

Интерфейс коллекции заливок.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IColourings

Описание:

Позволяет создавать и получать заливки.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Colourings.

IColourings – свойства

Colouring – Заливка, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс заливки IColouring

Синтаксис Automation:

Colouring = iObject.Colouring (Index);	Получить свойство(*)
Colouring = iObject.GetColouring(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Hatch(&Hatch)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IColourings – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IColouring ** Result);

Возвращаемое значение:

- Указатель на интерфейс заливки IColouring.

Интерфейс IConicCurves

Интерфейс коллекции конических кривых.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IDrawingObjects

IConicCurves

Данный интерфейс можно получить, используя свойство контейнера объектов 2D IDrawingContainer::ConicCurves.

Версия Компас v18.1

IConicCurves – свойства

ConicCurve – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IConicCurve.

Синтаксис Automation:

```
ConicCurve = Object.ConicCurve( Index )        Получить свойство (* )
ConicCurve = Object.GetConicCurve( Index )     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ConicCurve( Index, &ConicCurve )    Получить свойство
```

Входные параметры:

```
VARIANT Index                                  - индекс или reference элемента.
```

Примечание:

Свойство доступно только для чтения.

IConicCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IConicCurve * * Result);

Возвращаемое значение:

- Указатель на интерфейс IConicCurve.

Примечания:

1. Метод позволяет создать новый интерфейс конической кривой.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IDrawingContours

Интерфейс коллекции контуров

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IDrawingContours

Описание:

Позволяет получать и создавать контуры.

Примечание:

Получить интерфейс можно, используя свойство контейнера графических объектов IDrawingContainer::DrawingContours.

IDrawingContours – свойства

DrawingContour – Контур, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс контура IDrawingContour

Синтаксис Automation:

DrawingContour iObject.DrawingContour(Index);	=	Получить свойство(*)
DrawingContour iObject.GetDrawingContour(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingContour(Index, &DrawingContour)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс линии разреза/сечения.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IDrawingContours – методы

Add – Создать контур (добавить в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IDrawingContour ** Result);

Возвращаемое значение:

	- Указатель на интерфейс контура IDrawingContour.

AddBooleanResultContours – Булева операция над замкнутыми и не самопересекающимися кривыми

Интерфейс...

Синтаксис Automation:

VARIANT AddBooleanResultContours(IDrawingObject * Contour1,IDrawingObject * Contour2, ksBooleanType BooleanType);

Синтаксис COM:

HRESULT AddBooleanResultContours(IDrawingObject * Contour1,IDrawingObject * Contour2, ksBooleanType BooleanType,VARIANT * Result);

Возвращаемое значение:

	- Массив контуров IDrawingContour в виде SAFEARRAY DISPATCH - VT_ARRAY I VT_DISPATCH.

Входные параметры:

Contour1	- 1-й контур или кривая,
Contour2	- 2-й контур или кривая,
BooleanType	- тип булевой операции из перечисления ksBooleanType.

Метод позволяет выполнять над контурами операции пересечения, вычитания и объединения.

Примечание:

1. Кривые должны быть замкнутыми и не иметь самопересечений.
2. Если кривые не подходят или не пересекаются, вернется пустой массив.
3. Если возвращается один объект, то тип VARIANT будет VT_DISPATCH.
4. Если возвращается несколько объектов, то тип VARIANT будет VT_ARRAY | VT_DISPATCH.

Интерфейс IEllipseArcs

Справка системы КОМПАС...

`KOMPAS.chm::/145_15_6_Dugi_ehllipsov.htm`

Интерфейс коллекции дуг эллипсов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IEllipseArcs

Описание:

Позволяет создавать и получать дуги эллипсов на чертеже.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::EllipseArcs.

IEllipseArcs – свойства

IEllipseArcs – Дуга эллипса, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс дуги эллипса на чертеже IEllipseArc

Синтаксис Automation:

EllipseArc = iObject.EllipseArc (Index);		Получить свойство(*)
EllipseArc = iObject.GetEllipseArc(Index);		Получить свойство (**)

Синтаксис COM:

iObject->get_EllipseArc(&EllipseArc)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEllipseArcs – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEllipseArc ** Result);

Возвращаемое значение:

	- Указатель на интерфейс дуги эллипса IEllipseArc.

Интерфейс IEllipses

Справка системы КОМПАС...

KOMPAS.chm::/139_Glava14_Ehllipsy.htm

Интерфейс коллекции эллипсов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IEllipses

Описание:

Позволяет создавать и получать эллипсы на чертеже.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Ellipses.

IEllipses – свойства

Ellipse – Эллипс, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс эллипса на чертеже IEllipse

Синтаксис Automation:

Ellipse = iObject.Ellipse (Index);	Получить свойство(*)
Ellipse = iObject.GetEllipse(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Ellipse(&Ellipse)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEllipses – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEllipse ** Result);

Возвращаемое значение:

- Указатель на интерфейс эллипса IEllipse.

Интерфейс IEquidistants

Справка системы КОМПАС...

KOMPAS.chm::/CM_EQUID_TO_OBJ.htm

Интерфейс коллекции эквидистант.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IEquidistants

Описание:

Позволяет создавать и получать эквидистанты кривых.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Equidistants.

IEquidistants – свойства

Equidistant – Эквидистанта, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс эквидистанты IEquidistant

Синтаксис Automation:

Equidistant = iObject.Equidistant (Index);	Получить свойство(*)
Equidistant = iObject.GetEquidistant(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Equidistant(Index, &Equidistant)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEquidistants – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEquidistant ** Result);

Возвращаемое значение:

- Указатель на интерфейс эквидистанты IEquidistant.

Интерфейс IHatches

Справка системы КОМПАС...

KOMPAS.chm::/220_Glava_26_Shtrihovka_i_zalivka.htm

Интерфейс коллекции эквидистант.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IHatches

Описание:

Позволяет создавать и получать штриховки.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Hatches.

IHatches – свойства

Hatch – Штриховка, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс штриховки IHatch

Синтаксис Automation:

Hatch = iObject.Hatch (Index);	Получить свойство(*)
Hatch = iObject.GetHatch(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Hatch(&Hatch)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IHatches – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IHatch ** Result);

Возвращаемое значение:

- Указатель на интерфейс штриховки IHatch.

Интерфейс ILines

Справка системы КОМПАС...

KOMPAS.chm::/1396_vspomogatelnye.htm#simple_line

Интерфейс коллекции линий.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ILines

Описание:

Позволяет создавать и получать прямые линии.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Lines.

ILines – свойства

Line – Линия, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс линии на чертеже ILine

Синтаксис Automation:

Line = iObject.Line (Index);	Получить свойство(*)
Line = iObject.GetLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Line(Index, &Line) Получить свойство

Примечание:

Свойство доступно только для чтения.

ILines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILine ** Result);

Возвращаемое значение:

- Указатель на интерфейс линии ILine.

Интерфейс ILineSegments

Справка системы КОМПАС...

KOMPAS.chm::/130_Glava12_Otrezki.htm

Интерфейс коллекции отрезков.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ILineSegments

Описание:

Интерфейс позволяет получить коллекции отрезков.

Примечание:

Данный интерфейс можно получить у интерфейса IDrawingContainer, используя свойство IDrawingContainer::LineSegments.

ILineSegments – свойства

LineSegment – Отрезок, заданный по индексу

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/130_Glava12_Otrezki.htm

Тип данных: указатель на интерфейс отрезка ILineSegment

Синтаксис Automation:

LineSegment = iObject.LineSegment(Index); Получить свойство(*)
LineSegment = iObject.GetLineSegment(Index); Получить свойство (**)

Синтаксис COM:

iObject->get_LineSegment(Index, &LineSegment) Получить свойство

Входные параметры:

Index (Variant)	- Индекс отрезка в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта.

Примечание:

Свойство доступно только для чтения.

ILineSegments – методы

Add – Создать отрезок и добавить его в коллекцию

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/130_Glava12_Otrezki.htm

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add([out, retval] ILineSegment** Result);

Возвращаемое значение:

- Указатель на интерфейс ILineSegment отрезка.

Примечание:

Метод позволяет добавить отрезок в коллекцию. После получения нового интерфейса нужно задать параметры отрезка и вызвать метод IDrawingObject::Update.

Интерфейс IMultilines

Интерфейс коллекции мультилиний.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IMultilines

Описание:

Позволяет получать и создавать объект мультилинии.

Примечание:

Получить интерфейс можно, используя свойство контейнера графических объектов IDrawingContainer::Multilines.

IMultilines – свойства

Multiline – Мультилиния, заданная по индексу.

Интерфейс...

Тип данных: указатель на интерфейс мультилинии IMultiline

Синтаксис Automation:

```
Multiline = iObject.Multiline( Index );        Получить свойство(* )
Multiline = iObject.GetMultiline( Index );     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Multiline( Index, &Multiline )    Получить свойство
```

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс линии разреза/сечения.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IMultilines – методы

Add – Создать мультилинию (добавить в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IMultiLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс мультилинии IMultiLine.

Интерфейс INurbses

Интерфейс коллекции NURBS-кривых.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

INurbses

Описание:

Позволяет создавать и получать NURBS-сплайны.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Nurbses.

INurbses – свойства

Nurbs – NURBS-кривая, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс NURBS-кривой INurbs

Синтаксис Automation:

Nurbs = iObject.Nurbs (Index);	Получить свойство(*)
Nurbs = iObject.GetNurbs(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Nurbs(Index, &Nurbs)	Получить свойство

Примечание:

Свойство доступно только для чтения.

INurbses – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(INurbs ** Result);

Возвращаемое значение:

- Указатель на интерфейс NURBS-кривой INurbs.

Convert – Преобразовать кривую в NURBS–кривую и добавить в коллекцию.

Интерфейс...

Синтаксис Automation:

LPDISPATCH Convert(LPDISPATCH * Curve);

Синтаксис COM:

HRESULT Convert(IDrawingObject * Curve,
INurbs ** Result);

Входные параметры:

Curve	- указатель на интерфейс кривой, которую надо преобразовать в NURBS.

Возвращаемое значение:

- Указатель на интерфейс NURBS-кривой INurbs.

PickObjects – Собрать объекты по лучу

Интерфейс...

Синтаксис Automation:

LPDISPATCH ConvertEx(LPDISPATCH Curve, ksNurbsByPointsAproximationTypeEnum AproximationType, double Step);

Синтаксис COM:

HRESULT ConvertEx([in]IDrawingObject * Curve, ksNurbsByPointsAproximationTypeEnum AproximationType, double Step, INurbs ** Result);

Входные параметры:

Curve	- указатель на кривую,
AproximationType	- способ аппроксимации,
Step	- шаг аппроксимации.

Возвращаемое значение:

- Указатель на интерфейс кривой в случае успешного завершения.

Интерфейс IPoints

Справка системы КОМПАС...

KOMPAS.chm::/121_Glava10_Tochki.htm

Коллекция точек.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IPoints

Описание:

Интерфейс позволяет создавать и получать окружности на чертеже.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Points.

IPoints – свойства

Point – Точка, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс точки на чертеже IPoint

Синтаксис Automation:

Point = iObject.Point(Index);	Получить свойство(*)
Point = iObject.GetPoint(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Point(&Point)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPoints – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IPoint ** Result);

Возвращаемое значение:

- Указатель на интерфейс точки IPoint.

Интерфейс IPolyLines2D

Справка системы КОМПАС...

KOMPAS.chm::/149_17_1_Lomanaja.htm

Интерфейс коллекции полилиний.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IPolyLines2D

Описание:

Позволяет создавать и получать полилинии.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::PolyLines2D.

IPolyLines2D – свойства

PolyLine2D – Полилиния, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс полилинии IPolyLine2D.

Синтаксис Automation:

PolyLine2D = iObject.PolyLine2D (Index);	Получить свойство(*)
PolyLine2D = iObject.GetPolyLine2D(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_PolyLine2D(Index, &PolyLine2D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPolyLines2D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IPolyLine2D ** Result);

Возвращаемое значение:

- Указатель на интерфейс полилинии IPolyLine2D.

Интерфейс IRasters

Справка системы КОМПАС...

KOMPAS.chm::/353_Glava40_Ispolqzovanie_rastr.htm

Интерфейс коллекции растровых объектов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRasters

Описание:

Позволяет создавать и получать растровые объекты.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Rasters.

IRasters – свойства

Raster – Растровый объект, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс растрового объекта IRaster

Синтаксис Automation:

Raster = iObject.Raster (Index);	Получить свойство(*)
Raster = iObject.GetRaster(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Raster(&Raster)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IRasters – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRaster ** Result);

Возвращаемое значение:

- Указатель на интерфейс растрового объекта IRaster.

Интерфейс IRectangles

Справка системы КОМПАС...

KOMPAS.chm::/Prjamougolqnik.htm

Интерфейс коллекции прямоугольников.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRectangles

Описание:

Позволяет создавать и получать прямоугольники.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::Rectangles.

IRectangles – свойства

Rectangle – Прямоугольник, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс прямоугольника IRectangle

Синтаксис Automation:

Rectangle = iObject.Rectangle (Index);	Получить свойство(*)
Rectangle = iObject.GetRectangle(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Rectangle(Index, &Rectangle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IRectangles – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRectangle ** Result);

Возвращаемое значение:

- Указатель на интерфейс прямоугольника IRectangle.

Интерфейс IRegularPolygons

Справка системы КОМПАС...

KOMPAS.chm::/Mnogougolqnik.htm

Интерфейс коллекции многоугольников.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRegularPolygons

Описание:

Позволяет создавать и получать многоугольники.

Примечание:

Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::RegularPolygons.

IRegularPolygons – свойства

RegularPolygon – Многоугольник, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс многоугольника IRegularPolygon

Синтаксис Automation:

RegularPolygon = iObject.RegularPolygon (Index);	=	Получить свойство(*)
RegularPolygon = iObject.GetRegularPolygon(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_RegularPolygon(Index, &RegularPolygon)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IRegularPolygons – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRegularPolygon ** Result);

Возвращаемое значение:

- Указатель на интерфейс многоугольника IRegularPolygon.

Интерфейс IArc

Справка системы КОМПАС - Дуга...

KOMPAS.chm::/1436_Postroenie_dug.htm#duga_po_centru_dvum_tochkam

Справка системы КОМПАС - Дуга по двум точкам и углу раствора...

KOMPAS.chm::/1436_Postroenie_dug.htm#duga_po_dvum_tochkam_uglu

Справка системы КОМПАС - Дуга по двум точкам...

KOMPAS.chm::/1436_Postroenie_dug.htm#duga_po_dvum_tochkam

Справка системы КОМПАС - Дуга по трем точкам...

KOMPAS.chm::/1436_Postroenie_dug.htm#duga_po_trem_tochkam

Интерфейс дуги.

Иерархия:

IKompasAPIObject

IDrawingObject

IArc

Описание:

Интерфейс позволяет получить дугу.

Примечание:

1. Интерфейс можно получить у коллекции дуг, используя свойство IArcs::Arc или метод IArcs::Add.
2. При создании дуговой оси можно использовать несколько способов построения.

- По координатам,
- По трем точкам,
- По углам,

- По точке и углу,
- По углу, точке и радиусу,
- По конечным точкам и углу,
- Плавающий центр, вариант 1,
- Плавающий центр, вариант 2.

3. После задания параметров оси требуется вызвать метод IDrawingObject::Update.
4. При изменении параметров зависимые параметры пересчитываются после вызова метода IDrawingObject::Update.

IArc – свойства

Angle1 – Угол между осью 0X и отрезком, соединяющим первую точку дуги с началом координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle1 = iObject.Angle1;                Получить свойство(* )
iObject.Angle1 = Angle1;                Установить свойство (* )
Angle1 = iObject.GetAngle1();           Получить свойство (**)
iObject.SetAngle1( Angle1 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle1( &Angle1 )          Получить свойство
iObject->put_Angle1( Angle1 )           Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол между осью 0X и отрезком, соединяющим первую точку дуги с началом координат.

Angle2 – Угол между осью 0X и отрезком, соединяющим вторую точку дуги с началом координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle2 = iObject.Angle2;                Получить свойство(* )
iObject.Angle2 = Angle2;                Установить свойство (* )
Angle2 = iObject.GetAngle2();           Получить свойство (**)
iObject.SetAngle2( Angle2 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle2( &Angle2 )          Получить свойство
iObject->put_Angle2( Angle2 )           Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол между осью 0X и отрезком, соединяющим вторую точку дуги с началом координат.

Direction – Направление построения дуги

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = iObject.Direction;          Получить свойство(* )
iObject.Direction = Direction;          Установить свойство (* )
Direction = iObject.GetDirection();     Получить свойство (**)
iObject.SetDirection( Direction );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Direction( &Direction );   Получить свойство
iObject->put_Direction( Direction );    Установить свойство
```

Значения свойства:

```
TRUE        - по часовой стрелке,
FALSE       - против часовой стрелки.
```

Примечание:

Свойство позволяет получить и установить направление построения дуги.

Radius – Радиус дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = iObject.Radius;                Получить свойство(* )
iObject.Radius = Radius;                Установить свойство (* )
Radius = iObject.GetRadius();           Получить свойство (**)
iObject.SetRadius( Radius );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Radius( &Radius );         Получить свойство
```

```
iObject->put_Radius( Radius );                    Установить свойство
```

Примечание:

Свойство позволяет получить и установить радиус дуги.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
style = iObject.style;                    Получить свойство(* )
iObject.style = style;                    Установить свойство (* )
style = iObject.GetStyle();               Получить свойство (**)
iObject.SetStyle( style );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Style( &style );             Получить свойство
iObject->put_Style( style );              Установить свойство
```

Примечание:

Свойство позволяет считывать и устанавливать стиль линии.

Xc – Координата X центра дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = iObject.Xc;                          Получить свойство(* )
iObject.Xc = Xc;                          Установить свойство (* )
Xc = iObject.GetXc();                     Получить свойство (**)
iObject.SetXc( Xc );                      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Xc( &Xc );                   Получить свойство
iObject->put_Xc( Xc );                    Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Xc.

X1 – Координата первой точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1;                    Получить свойство(* )
iObject.X1 = X1;                    Установить свойство (* )
X1 = iObject.GetX1();               Получить свойство (**)
iObject.SetX1( X1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1( &X1 );             Получить свойство
iObject->put_X1( X1 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги X1.

X2 – Координата второй точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2;                    Получить свойство(* )
iObject.X2 = X2;                    Установить свойство (* )
X2 = iObject.GetX2();               Получить свойство (**)
iObject.SetX2( X2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2( &X2 );             Получить свойство
iObject->put_X2( X2 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги X2.

X3 – Координата средней точки, лежащей на дуге по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = iObject.X3;                    Получить свойство(* )
iObject.X3 = X3;                    Установить свойство (* )
X3 = iObject.GetX3();               Получить свойство (**)
iObject.SetX3( X3 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X3( &X3 );          Получить свойство
iObject->put_X3( X3 );           Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату средней точки, лежащей на дуге по X.

Yc – Координата Y центра дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = iObject.Yc;                 Получить свойство(* )
iObject.Yc = Yc;                 Установить свойство (* )
Yc = iObject.GetYc();            Получить свойство (**)
iObject.SetYc( Yc );             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Yc( &Yc );          Получить свойство
iObject->put_Yc( Yc );           Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Yc.

Y1 – Координата первой точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1;                 Получить свойство(* )
iObject.Y1 = Y1;                 Установить свойство (* )
Y1 = iObject.GetY1();            Получить свойство (**)
iObject.SetY1( Y1 );             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1( &Y1 );          Получить свойство
iObject->put_Y1( Y1 );           Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги Y1.

Y2 – Координата второй точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2;                    Получить свойство(* )
iObject.Y2 = Y2;                    Установить свойство (* )
Y2 = iObject.GetY2();               Получить свойство (**)
iObject.SetY2( Y2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2( &Y2 );             Получить свойство
iObject->put_Y2( Y2 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки дуги Y2.

Y3 – Координата средней точки, лежащей на дуге по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y3 = iObject.Y3;                    Получить свойство(* )
iObject.Y3 = Y3;                    Установить свойство (* )
Y3 = iObject.GetY3();               Получить свойство (**)
iObject.SetY3( Y3 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y3( &Y3 );             Получить свойство
iObject->put_Y3( Y3 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату средней точки, лежащей на дуге по Y.

Интерфейс IBezier

Справка системы КОМПАС...

`KOMPAS.chm::/spline_postroenie.htm#BEZIER`

Интерфейс кривой Безье.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IBezier

Примечание:

1. Интерфейс можно получить, используя свойство коллекции кривых Безье IBeziers::Bezier или метод IBeziers::Add.
2. После задания параметров окружности требуется вызвать метод IDrawingObject::Update.

IBezier – свойства

Closed – Замкнутость

Интерфейс..

Тип данных: BOOL

Синтаксис Automation:

Closed = Object.Closed	Получить свойство(*)
Object.Closed = Closed	Установить свойство (*)
Closed = Object.GetClosed()	Получить свойство (**)
Object.SetClosed(Closed)	Установить свойство (**)

Синтаксис COM:

Object.get_Closed(&Closed)	Получить свойство
Object.put_Closed(Closed)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать замкнутость кривой.

Style – Стиль линии

Интерфейс..

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии кривой.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

PointsCount – Количество опорных точек

Интерфейс..

Тип данных: long

Синтаксис Automation:

PointsCount = Object.PointsCount	=	Получить свойство(*)
PointsCount = Object.GetPointsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointsCount(&PointsCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество опорных точек.
2. Свойство доступно только для чтения.

Points – Параметры узлов в виде SAFEARRAY | VT_R8

Интерфейс..

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

Points = Object.Points(AllPoints)	Получить свойство(*)
Object.Points(AllPoints) = Points	Установить свойство (*)
Points = Object.GetPoints(AllPoints)	Получить свойство (**)
Object.SetPoints(AllPoints, Points)	Установить свойство (**)

Синтаксис COM:

Object.get_Points(AllPoints, &Points)	Получить свойство
Object.put_Points(AllPoints, Points)	Установить свойство

Входные параметры:

BOOL AllPoints	- TRUE - узел в массиве Points представлен опорной точкой и двумя крайними, - FALSE - узел в массиве Points представлен только опорной точкой.

Примечание:

Свойство позволяет устанавливать и получать параметры узлов кривой.

IBezier – методы

AddPoint – Добавить узел

Интерфейс..

Синтаксис Automation:

BOOL AddPoint(long Index,
double XBase,
double YBase,
double * XLeft,
double * YLeft,
double * XRight,
double * YRight)

Синтаксис COM:

HRESULT AddPoint(long Index,
double XBase,
double YBase,
double * XLeft,
double * YLeft,
double * XRight,
double * YRight,
BOOL * Result)

Входные параметры:

Index	- индекс добавляемого узла,
XBase, YBase	- координаты опорной точки узла,
XLeft, YLeft	- координаты левой точки узла,
XRight, YRight	- координаты правой точки узла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить узел в кривую Безье.

Clear – Удалить все узлы

Интерфейс...

Синтаксис Automation:

BOOL Clear()

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все узлы кривой.

DeletePoint – Удалить узел

Интерфейс..

Синтаксис Automation:

BOOL DeletePoint(long Index)

Синтаксис COM:

HRESULT DeletePoint(long Index, BOOL * Result)

Входные параметры:

Index	- индекс удаляемого узла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить заданный узел кривой.

GetPoint – Получить параметры узла

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long Index,
double XBase,
double YBase,
double * XLeft,
double * YLeft,
double * XRight,
double * YRight)

Синтаксис COM:

HRESULT GetPoint(long Index,
double XBase,
double YBase,
double * XLeft,
double * YLeft,
double * XRight,
double * YRight,
BOOL * Result)

Входные параметры:

Index	- индекс узла,

Выходные параметры:

XBase, YBase	- координаты опорной точки узла,
XLeft, YLeft	- координаты левой точки узла,
XRight, YRight	- координаты правой точки узла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры узла кривой Безье.

ReadFromFile – Прочитать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL ReadToFile(LPCTSTR FileName);

Синтаксис COM:

HRESULT ReadToFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае удачи.

WriteToFile – Записать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(LPCTSTR FileName);

Синтаксис COM:

HRESULT WriteToFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс ICircle

Справка системы КОМПАС...

KOMPAS.chm::/1432_Okruznosti.htm#okr_po_centru_i_tochke

Интерфейс окружности.

Иерархия:

IDispatch
- IKompasAPIObject
  - IDrawingObject
    - ICircle

Примечание:

1. Интерфейс можно получить, используя свойство коллекции окружностей ICircles::Circle или метод ICircles::Add.
2. После задания параметров окружности требуется вызвать метод IDrawingObject::Update.

ICircle – свойства

Radius – Радиус окружности

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство(*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Radius( &Radius )        Получить свойство
Object.put_Radius( Radius )         Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать радиус окружности.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                Получить свойство(* )
Object.Style = Style                Установить свойство (* )
Style = Object.GetStyle()           Получить свойство (**)
Object.SetStyle( Style )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )          Получить свойство
Object.put_Style(  Style )          Установить свойство
```

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии окружности.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

X – Координата точки на окружности по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X = Object.X                        Получить свойство(* )
Object.X = X                        Установить свойство (* )
X = Object.GetX()                   Получить свойство (**)
Object.SetXc( X )                   Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X( &X )                  Получить свойство
Object.put_X( X )                   Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату точки на окружности по оси X.

Xc – Координата центра окружности по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = Object.Xc                  Получить свойство(* )
Object.Xc = Xc                  Установить свойство (* )
Xc = Object.GetXc()             Получить свойство (**)
Object.SetXc( Xc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )            Получить свойство
Object.put_Xc( Xc )             Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату центра окружности по оси X.

Y – Координата точки на окружности по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y = Object.Y                    Получить свойство(* )
Object.Y = Y                    Установить свойство (* )
Y = Object.GetY()               Получить свойство (**)
Object.SetY( Y )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y( &Y )              Получить свойство
Object.put_Y(  Y )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату точки на окружности по оси Y.

Yc – Координата центра окружности по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                  Получить свойство(* )
Object.Yc = Yc                  Установить свойство (* )
Yc = Object.GetYc()             Получить свойство (**)
```

Object.SetYc(Yc) Установить свойство (**)

Синтаксис COM:

Object.get_Yc(&Yc) Получить свойство
Object.put_Yc(Yc) Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату центра окружности по оси Y.

Интерфейс IColouring

Справка системы КОМПАС...

KOMPAS.chm::/220_Glava_26_Shtrihovka_i_zalivka.htm

Интерфейс заливки.

Иерархия:

IDispatch
IKompasAPIObject
IDrawingObject
IColouring
IBoundariesObject

Описание:

Интерфейс позволяет задавать параметры заливки.

Примечание:

1. Интерфейс можно получить у коллекции заливок, используя свойство IColourings::Colouring или метод IColourings::Add.
2. После задания параметров штриховки требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс IBoundariesObject.

IColouring – свойства

ColouringType – Тип заливки

Интерфейс...

Тип данных: из перечисления ksColouringTypeEnum

Синтаксис Automation:

ColouringType = Object.ColouringType Получить свойство(*)
Object.ColouringType = ColouringType Установить свойство (*)
ColouringType = Object.GetColouringType() Получить свойство (**)
Object.SetColouringType(ColouringType) Установить свойство (**)

Синтаксис COM:

```
Object.get_ColouringType( &ColouringType )      Получить свойство
Object.put_ColouringType(  ColouringType )      Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать тип заливки.

Color1 – Цвет или начальный цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Color1 = Object.Color1                  Получить свойство(* )
Object.Color1 = Color1                  Установить свойство (* )
Color1 = Object.GetColor1()             Получить свойство (**)
Object.SetColor1( Color1 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Color1( &Color1 )            Получить свойство
Object.put_Color1(  Color1 )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать начальный цвет заливки.

Color2 – Цвет или конечный цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Color2 = Object.Color2                  Получить свойство(* )
Object.Color2 = Color2                  Установить свойство (* )
Color2 = Object.GetColor2()             Получить свойство (**)
Object.SetColor2( Color2 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Color2( &Color2 )            Получить свойство
Object.put_Color2(  Color2 )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать конечный цвет заливки.

GradientAngle – Угол поворота линий градиента

Интерфейс...

Тип данных: double

Синтаксис Automation:

GradientAngle = Object.GradientAngle	Получить свойство(*)
Object.GradientAngle = GradientAngle	Установить свойство (*)
GradientAngle = Object.GetGradientAngle()	Получить свойство (**)
Object.SetGradientAngle(GradientAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_GradientAngle(&GradientAngle)	Получить свойство
Object.put_GradientAngle(GradientAngle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол поворота линий градиента.

GradientCount – Количество шагов

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- плавное изменение цвета,
FALSE	- заданное количество шагов.

Синтаксис Automation:

GradientCount = Object.GradientCount	=	Получить свойство(*)
Object.GradientCount = GradientCount	=	Установить свойство (*)
GradientCount = Object.GetGradientCount()	=	Получить свойство (**)
Object.SetGradientCount(GradientCount)		Установить свойство (**)

Синтаксис COM:

Object.get_GradientCount(&GradientCount)	Получить свойство
Object.put_GradientCount(GradientCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество шагов градиентной заливки.

GradientType – Тип градиента

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- плавное изменение цвета,
FALSE	- заданное количество шагов.

Синтаксис Automation:

GradientType = Object.GradientType	=	Получить свойство(*)
Object.GradientType = GradientType	=	Установить свойство (*)
GradientType = Object.GetGradientType()	=	Получить свойство (**)
Object.SetGradientType(GradientType)		Установить свойство (**)

Синтаксис COM:

Object.get_GradientType(&GradientType)	Получить свойство
Object.put_GradientType(GradientType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип градиентной заливки.

Transparency1 – Начальная прозрачность

Интерфейс...

Тип данных: long

Синтаксис Automation:

Transparency1 = Object.Transparency1	=	Получить свойство(*)
Object.Transparency1 = Transparency1	=	Установить свойство (*)
Transparency1 = Object.GetTransparency1()	=	Получить свойство (**)

Object.SetTransparency1(Transparency1)		Установить свойство (**)

Синтаксис COM:

Object.get_Transparency1(&Transparency1)		Получить свойство
Object.put_Transparency1(Transparency1)		Установить свойство

Примечание:

Свойство позволяет устанавливать и получать начальную прозрачность в процентах.

Transparency2 – Конечная прозрачность

Интерфейс...

Тип данных: long

Синтаксис Automation:

Transparency2 = Object.Transparency2	=	Получить свойство(*)
Object.Transparency2 = Transparency2	=	Установить свойство (*)
Transparency2 = Object.GetTransparency2()	=	Получить свойство (**)
Object.SetTransparency2(Transparency2)		Установить свойство (**)

Синтаксис COM:

Object.get_Transparency2(&Transparency2)		Получить свойство
Object.put_Transparency2(Transparency2)		Установить свойство

Примечание:

Свойство позволяет устанавливать и получать конечную прозрачность в процентах.

Xc – Координата положения центра заливки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)
Object.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Xc( &Xc )          Получить свойство
Object.put_Xc(  Xc )          Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату положения центра заливки по оси X в процентах относительно габарита области.

Yc – Координата положения центра заливки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                Получить свойство(* )
Object.Yc = Yc                Установить свойство (* )
Yc = Object.GetYc()           Получить свойство (**)
Object.SetYc( Yc )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )          Получить свойство
Object.put_Yc(  Yc )          Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату положения центра заливки по оси Y в процентах относительно габарита области.

GradationsCount – Количество переходов цвета

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
GradationsCount          =    Получить свойство(* )
Object.GradationsCount
GradationsCount          =    Получить свойство (**)
Object.GetGradationsCount()
```

Синтаксис COM:

```
Object.get_GradationsCount(   Получить свойство
&GradationsCount )
```

Примечание:

1. Свойство позволяет получать количество переходов цвета.
2. Свойство доступно только для чтения.

IColouring – методы

AddGradation – Добавить переход цвета

Интерфейс...

Синтаксис Automation:

long AddGradation(long Position,
long Color,
long Transparency,
BOOL Interpolation);

Синтаксис COM:

HRESULT AddGradation(long Position,
long Color,
long Transparency,
BOOL Interpolation,
long * Result);

Входные параметры:

Position	- положение перехода относительно начальной точки заливки в процентах,
Color	- цвет перехода,
Transparency	- прозрачность перехода,
Interpolation	- значений опции *Интерполирование* для перехода.

Возвращаемое значение:

- индекс перехода.

Примечание:

Метод позволяет установить параметры переход цвета.

ClearGradations – Удалить промежуточные переходы цвета

Интерфейс...

Синтаксис Automation:

BOOL ClearGradations();

Синтаксис COM:

HRESULT ClearGradations(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить промежуточные переходы цвета.

DeleteGradation – Удалить переход цвета по индексу

Интерфейс...

Синтаксис Automation:

BOOL DeleteGradation(long Index);

Синтаксис COM:

HRESULT DeleteGradation(long Index,
BOOL * Result);

Входные параметры:

Index	- индекс перехода.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить переход цвета по индексу.

FindPosition – Получить индекс позиции

Интерфейс...

Синтаксис Automation:

long FindPosition(long Position);

Синтаксис COM:

HRESULT FindPosition(long Position,
long * Result);

Входные параметры:

Positi on	- положение перехода относительно начальной точки заливки в процентах.

Возвращаемое значение:

- индекс перехода.

Примечание:

Метод позволяет получить индекс позиции перехода цвета.

GetGradation – Получить параметры перехода цвета по индексу

Интерфейс...

Синтаксис Automation:

BOOL GetGradation(long Index,
long * Position,
long * Color,
long * Transparency,
BOOL * Interpolation);

Синтаксис COM:

HRESULT GetGradation(long Index,
long * Position,
long * Color,
long * Transparency,
BOOL * Interpolation,
BOOL * Result);

Входные параметры:

Index	- индекс перехода.

Выходные параметры:

Position	- положение перехода относительно начальной точки заливки в процентах,
Color	- цвет перехода,
Transparency	- прозрачность перехода,
Interpolation	- значений опции *Интерполирование* для перехода.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры перехода цвета по индексу.

GetGradations – Получить параметры переходов цвета

Интерфейс...

Синтаксис Automation:

BOOL GetGradations(VARIANT * Positions,
VARIANT * Colors,
VARIANT * Transparences,
VARIANT * Interpolations);

Синтаксис COM:

HRESULT GetGradations(VARIANT * Positions,
VARIANT * Colors,
VARIANT * Transparences,
VARIANT * Interpolations,
BOOL * Result);

Выходные параметры:

Positions	- массив SAFEARRAY I VT_I4 положений переходов относительно начальной точки заливки в процентах,
Colors	- массив SAFEARRAY I VT_I4 цветов переходов,
Transparences	- массив SAFEARRAY I VT_I4 прозрачностей переходов,
Interpolations	- массив SAFEARRAY I VT_BOOL значений опции *Интерполирование* для переходов.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры переходов цветов.

SetGradation – Установить параметры перехода цвета по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetGradation(long Index,
long * Position,
long * Color,
long * Transparency,
BOOL * Interpolation);

Синтаксис COM:

HRESULT SetGradation(long Index,
long * Position,
long * Color,
long * Transparency,
BOOL * Interpolation,
BOOL * Result);

Входные параметры:

Index	- индекс перехода,
Position	- положение перехода относительно начальной точки заливки в процентах,
Color	- цвет перехода,
Transparency	- прозрачность перехода,
Interpolation	- значений опции *Интерполирование* для перехода.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры перехода цвета по индексу.

SetGradations – Установить параметры переходов цвета

Интерфейс...

Синтаксис Automation:

BOOL SetGradations(VARIANT * Positions,
VARIANT * Colors,
VARIANT * Transparences,
VARIANT * Interpolations);

Синтаксис COM:

HRESULT SetGradations(VARIANT * Positions,
VARIANT * Colors,
VARIANT * Transparences,
VARIANT * Interpolations,
BOOL* Result);

Входные параметры:

Positions	- массив SAFEARRAY | VT_I4 положений переходов относительно начальной точки заливки в процентах,
Colors	- массив SAFEARRAY | VT_I4 цветов переходов,
Transparences	- массив SAFEARRAY | VT_I4 прозрачностей переходов,
Interpolations	- массив SAFEARRAY | VT_BOOL значений опции *Интерполирование* для переходов.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры переходов цветов.

Интерфейс IConicCurve

Интерфейс параметров конической кривой.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IConicCurve

Данный интерфейс можно получить с помощью метода коллекции конических кривых IConicCurves::Add или свойства IConicCurves::ConicCurve.

Версия Компас v18.1

IConicCurve – свойства

Angle1 – Угол наклона первой кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle1 = Object.Angle1	Получить свойство(*)
Object.Angle1 = Angle1	Установить свойство (*)
Angle1 = Object.GetAngle1()	Получить свойство (**)
Object.SetAngle1(Angle1)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle1(&Angle1)	Получить свойство
Object.put_Angle1(Angle1)	Установить свойство

Angle2 – Угол наклона второй кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство(*)
Object.Angle2 = Angle2	Установить свойство (*)
Angle2 = Object.GetAngle2()	Получить свойство (**)
Object.SetAngle2(Angle2)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle2(&Angle2)	Получить свойство
Object.put_Angle2(Angle2)	Установить свойство

Coefficient – Коэффициент

Интерфейс...

Тип данных: double

Синтаксис Automation:

Coefficient = Object.Coefficient	Получить свойство(*)
Object.Coefficient = Coefficient	Установить свойство (*)
Coefficient = Object.GetCoefficient()	Получить свойство (**)
Object.SetCoefficient(Coefficient)	Установить свойство (**)

Синтаксис COM:

Object.get_Coefficient(&Coefficient)	Получить свойство
Object.put_Coefficient(Coefficient)	Установить свойство

Height – Высота

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство(*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight()	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height)	Получить свойство
Object.put_Height(Height)	Установить свойство

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

IConicCurve – методы

GetPoint – Получить точку пересечения

Интерфейс...

Синтаксис Automation :

BOOL GetPoint(ksConicCurvePontIndexEnum PointIndex, double * X, double * Y);

Синтаксис COM:

HRESULT GetPoint(ksConicCurvePontIndexEnum PointIndex, double * X, double * Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

PointIndex	- индекс точки из перечисления ksConicCurvePontIndexEnum.

Выходные параметры:

X, Y	- координаты точки.

SetPoint – Установить точку пересечения

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(ksConicCurvePontIndexEnum PointIndex, double X, double Y);

Синтаксис COM:

HRESULT SetPoint(ksConicCurvePontIndexEnum PointIndex, double X, double Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

PointIndex	- индекс точки из перечисления ksConicCurvePontIndexEnum,
X, Y	- координаты точки.

Интерфейс IDrawingContour

Интерфейс контура.

Иерархия:

IKompasAPIObject

IDrawingObject

IDrawingContour

IContour

Примечание:

1. Интерфейс позволяет получить и задать свойства объекта "контур".
2. Получить интерфейс можно, используя свойство IDrawingContours::DrawingContour или метод IDrawingContours::Add интерфейса коллекции контуров.
3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.
4. Интерфейс IContour являются дополнительным. Его можно получить с помощью метода IUnknown::QueryInterface.

IDrawingContour – свойства

Style – Стиль линии

Интерфейс...

Тип данных: long (соответствует перечислению ksCurveStyleEnum)

Синтаксис Automation:

```
Style = iObject.Style( );                Получить свойство(* )
iObject.Style( ) = Style;                Установить свойство (* )
Style = iObject.GetStyle( );             Получить свойство (**)
iObject.SetStyle( Style );               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Style( &Style )             Получить свойство
iObject->put_Style( Style )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать стиль линии.

Интерфейс IEllipse

Справка системы КОМПАС...

KOMPAS.chm::/139_Glava14_Ehllipsy.htm

Интерфейс эллипса.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IEllipse

Описание:

Интерфейс позволяет получить и задать свойства эллипса.

Примечание:

1. Интерфейс можно получить, используя свойство коллекции эллипсов IEllipses::Ellipse или метод IEllipses::Add.
2. После задания параметров эллипса требуется вызвать метод IDrawingObject::Update.

IEllipse – свойства

Angle – Угол наклона первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                     Получить свойство(* )
Object.Angle = Angle                     Установить свойство (* )
Angle = Object.GetAngle()                Получить свойство (**)
Object.SetAngle( Angle )                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )          Получить свойство
Object.put_Angle(  Angle )          Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол наклона первой полуоси эллипса.

SemiAxisA – Длина первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
SemiAxisA = Object.SemiAxisA        Получить свойство(* )
Object.SemiAxisA = SemiAxisA        Установить свойство (* )
SemiAxisA                  =        Получить свойство (**)
Object.GetSemiAxisA()
Object.SetSemiAxisA(                Установить свойство (**)
SemiAxisA )
```

Синтаксис COM:

```
Object.get_SemiAxisA(               Получить свойство
&SemiAxisA )
Object.put_SemiAxisA(               Установить свойство
SemiAxisA )
```

Примечание:

Свойство позволяет устанавливать и получать длину первой полуоси эллипса.

SemiAxisB – Длина второй полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
SemiAxisB = Object.SemiAxisB        Получить свойство(* )
Object.SemiAxisB = SemiAxisB        Установить свойство (* )
SemiAxisB                  =        Получить свойство (**)
Object.GetSemiAxisB()
Object.SetSemiAxisB(                Установить свойство (**)
SemiAxisB )
```

Синтаксис COM:

Object.get_SemiAxisB(&SemiAxisB)	Получить свойство
Object.put_SemiAxisB(SemiAxisB)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину второй полуоси эллипса.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии эллипса.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

Xc – Координата центра эллипса по оси X

Интерфейс...

Тип данных:double.

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)
Object.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

Object.get_Xc(&Xc)	Получить свойство
Object.put_Xc(Xc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату центра эллипса по оси X.

X1 – Координата конечной точки первой полуоси по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1()	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

Object.get_X1(&X1)	Получить свойство
Object.put_X1(X1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки первой полуоси по оси X.

X2 – Координата конечной точки второй полуоси по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X2 = Object.X2	Получить свойство(*)
Object.X2 = X2	Установить свойство (*)
X2 = Object.GetX2()	Получить свойство (**)
Object.SetX2(X2)	Установить свойство (**)

Синтаксис COM:

Object.get_X2(&X2)	Получить свойство
Object.put_X2(X2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки второй полуоси по оси X.

Yc – Координата центра эллипса по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Yc = Object.Yc	Получить свойство(*)
Object.Yc = Yc	Установить свойство (*)
Yc = Object.GetYc()	Получить свойство (**)
Object.SetYc(Yc)	Установить свойство (**)

Синтаксис COM:

Object.get_Yc(&Yc)	Получить свойство
Object.put_Yc(Yc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату центра эллипса по оси Y.

Y1 – Координата конечной точки первой полуоси по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1()	Получить свойство (**)
Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство
Object.put_Y1(Y1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки первой полуоси по оси Y.

Y2 – Координата конечной точки второй полуоси по оси Y

Интерфейс...

Тип данных:double.

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки второй полуоси по оси Y.

Интерфейс IEllipseArc

Справка системы КОМПАС...

KOMPAS.chm::/145_15_6_Dugi_ehllipsov.htm

Интерфейс дуги эллипса.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IEllipseArc

Описание:

Интерфейс позволяет получить и задать свойства дуги эллипса.

Примечание:

1. Интерфейс можно получить, используя свойство коллекции дуг эллипсов IEllipseArcs::EllipseArc или метод IEllipseArcs::Add.
2. После задания параметров дуги эллипса требуется вызвать метод IDrawingObject::Update.

IEllipseArc – свойства

Angle – Угол наклона первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол наклона первой полуоси дуги эллипса.

Angle1 – Начальный угол дуги к первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle1 = Object.Angle1	Получить свойство(*)
Object.Angle1 = Angle1	Установить свойство (*)
Angle1 = Object.GetAngle1()	Получить свойство (**)
Object.SetAngle1(Angle1)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle1(&Angle1)	Получить свойство
Object.put_Angle1(Angle1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать начальный угол дуги к первой полуоси эллипса.

Angle2 – Конечный угол дуги к первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство(*)
Object.Angle2 = Angle2	Установить свойство (*)
Angle2 = Object.GetAngle2()	Получить свойство (**)
Object.SetAngle2(Angle2)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle2(&Angle2)	Получить свойство
Object.put_Angle2(Angle2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать конечный угол дуги к первой полуоси эллипса.

Direction – Направление построения

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- по часовой стрелке,
FALSE	- против часовой стрелки.

Синтаксис Automation:

Direction = Object.Direction	Получить свойство(*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetAngle(Angle)Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать направление построения дуги эллипса.

SemiAxisA – Длина первой полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

SemiAxisA = Object.SemiAxisA	Получить свойство(*)
Object.SemiAxisA = SemiAxisA	Установить свойство (*)
SemiAxisA = Object.GetSemiAxisA()	Получить свойство (**)
Object.SetSemiAxisA(SemiAxisA)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisA(&SemiAxisA)	Получить свойство
Object.put_SemiAxisA(SemiAxisA)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину первой полуоси дуги эллипса.

SemiAxisB – Длина второй полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

SemiAxisB = Object.SemiAxisB	Получить свойство(*)
Object.SemiAxisB = SemiAxisB	Установить свойство (*)
SemiAxisB = Object.GetSemiAxisB()	Получить свойство (**)
Object.SetSemiAxisB(SemiAxisB)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisB(&SemiAxisB)	Получить свойство
Object.put_SemiAxisB(SemiAxisB)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину второй полуоси дуги эллипса.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии дуги эллипса.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

T1 – Начальное значение параметра

Интерфейс...

Тип данных: double

Синтаксис Automation:

T1 = Object.T1	Получить свойство(*)
Object.T1 = T1	Установить свойство (*)
T1 = Object.GetT1()	Получить свойство (**)
Object.SetT1(T1)	Установить свойство (**)

Синтаксис COM:

Object.get_T1(&T1)	Получить свойство
Object.put_T1(T1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать начальное значение параметра.

T2 – Конечное значение параметра

Интерфейс...

Тип данных: double

Синтаксис Automation:

T2 = Object.T2	Получить свойство(*)
Object.T2 = T2	Установить свойство (*)
T2 = Object.GetT2()	Получить свойство (**)
Object.SetT2(T2)	Установить свойство (**)

Синтаксис COM:

Object.get_T2(&T2)	Получить свойство
Object.put_T2(T2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать конечное значение параметра.

Xc – Координата центра дуги эллипса по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)
Object.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

Object.get_Xc(&Xc)	Получить свойство
Object.put_Xc(Xc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату центра дуги эллипса по оси X.

Yc – Координата центра дуги эллипса по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Yc = Object.Yc	Получить свойство(*)
Object.Yc = Yc	Установить свойство (*)
Yc = Object.GetYc()	Получить свойство (**)
Object.SetYc(Yc)	Установить свойство (**)

Синтаксис COM:

Object.get_Yc(&Yc)	Получить свойство
Object.put_Yc(Yc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату центра дуги эллипса по оси Y.

Интерфейс IEquidistant

Справка системы КОМПАС...

`KOMPAS.chm::/CM_EQUID_TO_OBJ.htm`

Интерфейс эквидистанты.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IEquidistant

Описание:

Интерфейс позволяет задавать параметры эквидистанты.

Примечание:

1. Интерфейс можно получить у коллекции эквидистант, используя свойство IEquidistants::Equidistant или метод IEquidistants::Add.
2. После задания параметров эквидистанты требуется вызвать метод IDrawingObject::Update.

IEquidistant – свойства

BaseObject – Базовая кривая

Интерфейс...

Тип данных: указатель на интерфейс графического объекта IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать базовую кривую эквидистанты.

CutMode – Обход угла

Интерфейс...

Тип данных – BOOL.

Значения свойства:

TRUE	- обход дугой,
FALSE	- обход срезом.

Синтаксис Automation:

CutMode = Object.CutMode	Получить свойство(*)
Object.CutMode = CutMode	Установить свойство (*)
CutMode = Object.GetCutMode()	Получить свойство (**)
Object.SetCutMode(CutMode)	Установить свойство (**)

Синтаксис COM:

Object.get_CutMode(&CutMode)	Получить свойство
Object.put_CutMode(CutMode)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ обхода угла кривой.

DegenerateSegment – Режим отрисовки вырожденных участков

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- вырожденные сегменты разрешены,
FALSE	- вырожденные сегменты запрещены.

Синтаксис Automation:

DegenerateSegment = Object.DegenerateSegment		Получить свойство(*)
Object.DegenerateSegment = DegenerateSegment		Установить свойство (*)
DegenerateSegment = Object.GetDegenerateSegment()		Получить свойство (**)
Object.SetDegenerateSegment(DegenerateSegment)		Установить свойство (**)

Синтаксис COM:

Object.get_DegenerateSegment(&DegenerateSegment)	Получить свойство
Object.put_DegenerateSegment(DegenerateSegment)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать режим отрисовки вырожденных участков.

LeftRadius – Радиус эквидистанты слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeftRadius = Object.LeftRadius		Получить свойство(*)
Object.LeftRadius = LeftRadius		Установить свойство (*)
LeftRadius = Object.GetLeftRadius()		Получить свойство (**)
Object.SetLeftRadius(LeftRadius)		Установить свойство (**)

Синтаксис COM:

Object.get_LeftRadius(&LeftRadius)	Получить свойство
Object.put_LeftRadius(LeftRadius)DegenerateSegment)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать радиус эквидистанты слева.

RightRadius – Радиус эквидистанты справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

RightRadius = Object.RightRadius	=	Получить свойство(*)
Object.RightRadius = RightRadius	=	Установить свойство (*)
RightRadius = Object.GetRightRadius()	=	Получить свойство (**)
Object.SetRightRadius(RightRadius)		Установить свойство (**)

Синтаксис COM:

Object.get_RightRadius(&RightRadius)	Получить свойство
Object.put_RightRadius(RightRadius)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать радиус эквидистанты справа.

Side – Тип построения

Интерфейс...

Тип данных – из перечисления ksEquidistantTypeEnum.

Синтаксис Automation:

Side = Object.Side	Получить свойство(*)
Object.Side = Side	Установить свойство (*)
Side = Object.GetSide()	Получить свойство (**)
Object.SetSide(Side)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Side( &Side )        Получить свойство
Object.put_Side(  Side )        Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать тип построения эквидистанты.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style            Получить свойство(* )
Object.Style = Style            Установить свойство (* )
Style = Object.GetStyle()       Получить свойство (**)
Object.SetStyle( Style )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )      Получить свойство
Object.put_Style(  Style )      Установить свойство
```

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии окружности.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

IEquidistant – методы

SetCorner – Установить параметры скругления/усечения угла

Интерфейс...

Синтаксис Automation:

```
BOOL SetCorner( long Index,
long * Type,
double * L1,
double * L2 );
```

Синтаксис COM:

```
HRESULT SetCorner( long Index,
ksCornerTypeEnum * Type,
double * L1,
double * L2,
BOOL* Result );
```

Входные параметры:

Index	- индекс угла.
Type	- тип угла из перечисления ksCornerTypeEnum,
L1	- длина фаски первого сегмента или радиус скругления,
L2	- длина фаски второго сегмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры скругления/усечения угла многоугольника.

Интерфейс IHatch

Справка системы КОМПАС...

KOMPAS.chm::/220_Glava_26_Shtrihovka_i_zalivka.htm

Интерфейс штриховки.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IHatch

IHatchParam

IBoundariesObject

Описание:

Интерфейс позволяет задавать параметры штриховки.

Примечание:

1. Интерфейс можно получить у коллекции штриховок, используя свойство IHatches::Hatch или метод IHatches::Add.
2. После задания параметров штриховки требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы:

- IHatchParam,
- IBoundariesObject.

IHatch – свойства

Side – Расположение штриховки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - сторона 1.

Синтаксис Automation:

Side = Object.Side	Получить свойство(*)
Object.Side = Side	Установить свойство (*)
Side = Object.GetSide()	Получить свойство (**)
Object.SetSide(Side)	Установить свойство (**)

Синтаксис COM:

Object.get_Side(&Side)	Получить свойство
Object.put_Side(Side)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать расположение штриховки.

X – Координата базовой точки штриховки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки штриховки по оси X.

Y – Координата базовой точки штриховки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки штриховки по оси Y.

Интерфейс ILine

Справка системы КОМПАС...

KOMPAS.chm::/1396_vspomogatelnye.htm#simple_line

Интерфейс линии.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

ILine

Описание:

Интерфейс позволяет получить и задать свойства прямой линии.

Примечание:

1. Интерфейс можно получить, используя свойство коллекции линий ILines::Line или метод ILines::Add.
2. После задания параметров линии требуется вызвать метод IDrawingObject::Update.

ILine – свойства

Angle – Угол между линией и осью OX в текущей системе координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                    Получить свойство(* )
Object.Angle = Angle                    Установить свойство (* )
Angle = Object.GetAngle()               Получить свойство (**)
Object.SetAngle( Angle )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )              Получить свойство
Object.put_Angle(  Angle )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол между линией и осью OX в текущей системе координат.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X1                          Получить свойство(* )
Object.X1 = X1                          Установить свойство (* )
X1 = Object.GetX1()                     Получить свойство (**)
Object.SetX1( X1 )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )                    Получить свойство
Object.put_X1(  X1 )                    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки, через которую проходит линия, по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                          Получить свойство(* )
Object.X2 = X2                          Установить свойство (* )
X2 = Object.GetX2()                     Получить свойство (**)
Object.SetX2( X2 )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )                Получить свойство
Object.put_X2(  X2 )                Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки, через которую проходит линия, по оси Y.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1                      Получить свойство(* )
Object.Y1 = Y1                      Установить свойство (* )
Y1 = Object.GetY1()                 Получить свойство (**)
Object.SetY1( Y1 )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )                Получить свойство
Object.put_Y1(  Y1 )                Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки, через которую проходит линия, по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = Object.Y2                      Получить свойство(* )
Object.Y2 = Y2                      Установить свойство (* )
Y2 = Object.GetY2()                 Получить свойство (**)
Object.SetY2( Y2 )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y2( &Y2 )                Получить свойство
Object.put_Y2(  Y2 )                Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату второй точки, через которую проходит линия, по оси Y.

Интерфейс ILineSegment

Справка системы КОМПАС...

KOMPAS.chm::/130_Glava12_Otrezki.htm

Интерфейс отрезка.

Иерархия:

IKompasAPIObject

IDrawingObject

ILineSegment

Описание:

Примечание:

1. Интерфейс можно получить у коллекции отрезков, используя свойство ILineSegments::LineSegment или метод ILineSegments::Add.
2. При создании отрезка можно использовать несколько типов построения:
   2.1. по координатам (требуется задать координаты точек (X1; Y1), (X2; Y2).
   2.2. По точке 1 (X1; Y1), длине Length и углу Angle,-
   По точке 2 (X2; Y2), длине Length и углу Angle относительно точки 1.
3. После задания параметров отрезка требуется вызвать метод IDrawingObject::Update.
4. При изменении параметров зависимые параметры пересчитываются после вызова метода IDrawingObject::Update.

ILineSegment – свойства

Angle – Угол наклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle;	Получить свойство(*)
iObject.Angle = Angle;	Установить свойство (*)
Angle = iObject.GetAngle();	Получить свойство (**)
iObject.SetAngle(Angle);	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle(&Angle)	Получить свойство
iObject->put_Angle(Angle);	Установить свойство

Примечание:

Свойство позволяет получить и установить угол наклона отрезка.

Length – Длина отрезка

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = iObject.Length;                Получить свойство(* )
iObject.Length = Length;                Установить свойство (* )
Length = iObject.GetLength();           Получить свойство (**)
iObject.SetLength( Length );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Length(&Length);           Получить свойство
iObject->put_Length( Length );          Установить свойство
```

Примечание:

Свойство позволяет получить и установить длину отрезка.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
style = iObject.style;                  Получить свойство(* )
iObject.style = style;                  Установить свойство (* )
style = iObject.GetStyle();             Получить свойство (**)
iObject.SetStyle( style );              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Style( &Style );           Получить свойство
iObject->put_Style( style );            Установить свойство
```

Примечание:

Свойство позволяет получить и установить стиль линии отрезка.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1;                    Получить свойство(* )
iObject.X1 = X1;                    Установить свойство (* )
X1 = iObject.GetX1();               Получить свойство (**)
iObject.SetX1( X1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1(&X1);               Получить свойство
iObject->put_X1( X1 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки отрезка по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2;                    Получить свойство(* )
iObject.X2 = X2;                    Установить свойство (* )
X2 = iObject.GetX2();               Получить свойство (**)
iObject.SetX2( X2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2(&X2);               Получить свойство
iObject->put_X2( X2 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки отрезка по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1;                    Получить свойство(* )
iObject.Y1 = Y1;                    Установить свойство (* )
Y1 = iObject.GetY1();               Получить свойство (**)
iObject.SetY1( Y1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1(&Y1);          Получить свойство
iObject->put_Y1( Y1 );         Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки отрезка по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2;               Получить свойство(* )
iObject.Y2 = Y2;               Установить свойство (* )
Y2 = iObject.GetY2();          Получить свойство (**)
iObject.SetY2( Y2 );           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2(&Y2);          Получить свойство
iObject->put_Y2( Y2 );         Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки отрезка по оси Y.

Интерфейс IMultiline

Интерфейс мультилинии.

Иерархия:

IDispatch
    IKompasAPIObject
        IDrawingObject
            IMultiline

Примечание:

1. Интерфейс позволяет получить и задать свойства объекта "мультилиния". Для создания объекта должен быть задан базовый контур и хотя бы одна линия.
2. Получить интерфейс можно, используя свойство IMultilines::MultiLine или метод IMultilines::Add интерфейса коллекции мультилиний.
3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

IMultiline – свойства

BaseContour – Базовый контур

Интерфейс...

Тип данных: указатель на интерфейс контура IContour.

Синтаксис Automation:

```
BaseContour = iObject.BaseContour();        Получить свойство(* )
BaseContour = iObject.GetBaseContour();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_BaseContour( &BaseContour )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE    - мультилиния замкнута,
FALSE   - мультилиния разомкнута.
```

Синтаксис Automation:

```
Closed = iObject.Closed;            Получить свойство(* )
iObject.Closed = Closed;            Установить свойство (* )
Closed = iObject.GetClosed();       Получить свойство (**)
iObject.SetClosed(Closed);          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Closed( &Closed )      Получить свойство
iObject->put_Closed( Closed )       Установить свойство
```

Примечание:

1. Свойство позволяет устанавливать и получать замкнутость.
2. Если при создании замкнутой линии не совпадают точки начала и конца,будет создан дополнительный сегмент.

3. Ограничители у замкнутой мультилинии не учитываются.

EndLimiter – Типы ограничений на концах мультилинии

Интерфейс...

Тип данных: из перечисления ksMlEndLimiterEnum

Синтаксис Automation:

EndLimiter = iObject.EndLimiter(First);	Получить свойство(*)
iObject.EndLimiter(First) = EndLimiter;	Установить свойство (*)
EndLimiter = iObject.GetEndLimiter(First);	Получить свойство (**)
iObject.SetEndLimiter(First, EndLimiter);	Установить свойство (**)

Синтаксис COM:

iObject->get_EndLimiter(First, &EndLimiter)	Получить свойство
iObject->put_EndLimiter(First, EndLimiter)	Установить свойство

Входные параметры:

First	- номер ограничителя (TRUE - первая, FALSE - вторая).

Примечание:

Свойство позволяет устанавливать и получать типы ограничений на концах мультилинии.

EndParameter – Параметр ограничителя в %(0–100)

Интерфейс...

Тип данных: double

Синтаксис Automation:

EndParameter = iObject.EndParameter(First);	Получить свойство(*)
iObject.EndParameter(First) = EndParameter;	Установить свойство (*)
EndParameter = iObject.EndParameter(First);	Получить свойство (**)
iObject.EndParameter(First, EndParameter);	Установить свойство (**)

Синтаксис COM:

iObject->get_EndParameter(First, &EndParameter)	Получить свойство
iObject->put_EndParameter(First, EndParameter)	Установить свойство

Входные параметры:

First - номер ограничителя (TRUE - первая, FALSE - вторая).

Примечание:

Свойство позволяет устанавливать и получать параметр ограничителя в процентах от ширины мультилинии.

EndStyle – Стиль линии кривой в ограничителе

Интерфейс...

Тип данных: long (соответствует перечислению ksCurveStyleEnum)

Синтаксис Automation:

EndStyle = iObject.EndStyle(First);	Получить свойство(*)
iObject.EndStyle(First) = EndStyle;	Установить свойство (*)
EndStyle = iObject.GetEndStyle(First);	Получить свойство (**)
iObject.SetEndStyle(First, EndStyle);	Установить свойство (**)

Синтаксис COM:

iObject->get_EndStyle(First, &EndStyle)	Получить свойство
iObject->put_EndStyle(First, EndStyle)	Установить свойство

Входные параметры:

First - номер ограничителя (TRUE - первая, FALSE - вторая).

Примечание:

Свойство позволяет устанавливать и получать стиль линии в ограничителе.

LineCount – Количество линий

Интерфейс...

Тип данных: long

Синтаксис Automation:

LineCount = iObject.LineCount();	Получить свойство(*)
LineCount = iObject.GetLineCount();	Получить свойство (**)

Синтаксис COM:

iObject->get_LineCount(&LineCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество линий в мультилинии.
2. Свойство доступно только для чтения.

LineOffset – Смещение линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineOffset = iObject.LineOffset(LineIndex);	Получить свойство(*)
iObject.LineOffset(LineIndex) = LineOffset;	Установить свойство (*)
LineOffset = iObject.GetLineOffset(LineIndex);	Получить свойство (**)
iObject.SetLineOffset(LineIndex, LineOffset););	Установить свойство (**)

Синтаксис COM:

iObject->get_LineOffset(LineIndex, &LineOffset)	Получить свойство
iObject->put_LineOffset(LineIndex, LineOffset)	Установить свойство

Входные параметры:

long LineIndex	- индекс линии.

Примечание:

Свойство позволяет устанавливать смещение линии относительно базового контура.

LineOffsets – Массив смещений в виде массива SAFEARRAY double – VT_ARRAY | VT_R8

Интерфейс...

Тип данных: VARIANT

Значения свойства:

Массив SafeArray типа VT_ARRAY I VT_R8.

Синтаксис Automation:

LineOffsets = iObject.LineOffsets;	Получить свойство(*)
LineOffsets = iObject.GetLineOffsets();	Получить свойство (**)

Синтаксис COM:

iObject->get_LineOffsets(&LineOffsets)	Получить свойство

Примечание:

1. Свойство позволяет получить массив SAFEARRAY смещения всех линий.
2. Свойство доступно только для чтения.

LineStyle – Стиль линии

Интерфейс...

Тип данных: long (соответствует перечислению ksCurveStyleEnum)

Синтаксис Automation:

LineStyle = iObject.LineStyle(LineIndex);	Получить свойство(*)
iObject.LineStyle(LineIndex) = LineStyle;	Установить свойство (*)
LineStyle = iObject.GetLineStyle(LineIndex);	Получить свойство (**)
iObject.SetLineStyle(LineIndex, LineStyle);	Установить свойство (**)

Синтаксис COM:

iObject->get_LineStyle(LineIndex, &LineStyle)	Получить свойство
iObject->put_LineStyle(LineIndex, LineStyle)	Установить свойство

Входные параметры:

long LineIndex - индекс линии.

Примечание:

Свойство позволяет устанавливать и получать стиль линии.

TrackingType – Тип обхода вершины

Интерфейс...

Тип данных: из перечисления ksMlVertexTrackingEnum

Синтаксис Automation:

TrackingType = iObject.TrackingType(LineIndex);	Получить свойство(*)
iObject.TrackingType(LineIndex) = TrackingType;	Установить свойство (*)
TrackingType = iObject.GetTrackingType(LineIndex);	Получить свойство (**)
iObject.SetTrackingType(LineIndex, TrackingType);	Установить свойство (**)

Синтаксис COM:

iObject->get_TrackingType(LineIndex, &TrackingType)	Получить свойство
iObject->put_TrackingType(LineIndex, TrackingType)	Установить свойство

Входные параметры:

long LineIndex - индекс линии.

Примечание:

Свойство позволяет устанавливать и получать тип обхода вершины мультилинии.

VertexCount – Количество вершин без учета концов

Интерфейс...

Тип данных: указатель на интерфейс контура IContour

Синтаксис Automation:

VertexCount iObject.VertexCount();	=	Получить свойство(*)
VertexCount iObject.GetVertexCount();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_VertexCount(&VertexCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество вершин мультилинии.
2. Свойство доступно только для чтения.

VertexDirection – Направление дуги в вершине

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- по часовой стрелке,
FALSE	- против часовой стрелки.

Синтаксис Automation:

VertexDirection = iObject.VertexDirection(VertexIndex);	=	Получить свойство(*)
iObject.VertexDirection(VertexIndex) = VertexDirection;		Установить свойство (*)
VertexDirection = iObject.GetVertexDirection(VertexIndex);	=	Получить свойство (**)
iObject.SetVertexDirection(VertexIndex, VertexDirection);		Установить свойство (**)

Синтаксис COM:

iObject->get_VertexDirection(VertexIndex, &VertexDirection)	Получить свойство
iObject->put_VertexDirection(VertexIndex, VertexDirection)	Установить свойство

Входные параметры:

long VertexIndex	- индекс вершины.

Примечание:

1. Свойство позволяет устанавливать и получать направление дуги в вершине мультилинии.
2. Свойство доступно только при типе обхода вершины ksMTrShear (ksMlEndLimiterEnum).

VertexLimiter – Типы ограничений в вершинах мультилинии

Интерфейс...

Тип данных: из перечисления ksMlVertexLimiterEnum

Синтаксис Automation:

VertexLimiter = iObject.VertexLimiter(VertexIndex);	=	Получить свойство(*)
iObject.VertexLimiter(VertexIndex) = VertexLimiter;		Установить свойство (*)
VertexLimiter = iObject.GetVertexLimiter(VertexIndex);	=	Получить свойство (**)
iObject.SetVertexLimiter(VertexIndex, VertexLimiter);		Установить свойство (**)

Синтаксис COM:

iObject->get_VertexLimiter(VertexIndex, &VertexLimiter)	Получить свойство
iObject->put_VertexLimiter(VertexIndex, VertexLimiter)	Установить свойство

Входные параметры:

long VertexIndex	- индекс вершины.

Примечание:

1. Свойство позволяет устанавливать и получать тип ограничения в вершине мультилинии.
2. Свойство доступно только при типе обхода вершины ksMTrShear (ksMlEndLimiterEnum).

VertexRadius – Радиус дуги в вершине

Интерфейс...

Тип данных: double

Синтаксис Automation:

VertexRadius = iObject.VertexRadius(VertexIndex);	=	Получить свойство(*)
iObject.VertexRadius(VertexIndex) = VertexRadius;		Установить свойство (*)

VertexRadius = Получить свойство (**)
iObject.GetVertexRadius(
VertexIndex);
iObject.SetVertexRadius(Установить свойство (**)
VertexIndex, VertexRadius);

Синтаксис COM:

iObject->get_VertexRadius(Получить свойство
VertexIndex, &VertexRadius)
iObject->put_VertexRadius(Установить свойство
VertexIndex, VertexRadius)

Входные параметры:

long VertexIndex - индекс вершины.

Примечание:

1. Свойство позволяет устанавливать и получать радиус дуги в вершине мультилинии.
2. Свойство доступно только при типе обхода вершины ksMTrShear (ksMlEndLimiterEnum).

VertexStyle – Стиль кривой в вершине

Интерфейс...

Тип данных: long (соответствует перечислению ksCurveStyleEnum)

Синтаксис Automation:

VertexStyle = Получить свойство(*)
iObject.VertexStyle(VertexIndex
);
iObject.VertexStyle(VertexIndex Установить свойство (*)
) = VertexStyle;
VertexStyle = Получить свойство (**)
iObject.GetVertexStyle(
VertexIndex);
iObject.SetVertexStyle(Установить свойство (**)
VertexIndex, VertexStyle);

Синтаксис COM:

iObject->get_VertexStyle(Получить свойство
VertexIndex, &VertexStyle)
iObject->put_VertexStyle(Установить свойство
VertexIndex, VertexStyle)

Входные параметры:

long VertexIndex	- индекс вершины.

Примечание:

1. Свойство позволяет устанавливать и получать тип ограничения в вершине мультилинии.
2. Свойство доступно только при типе обхода вершины ksMTrShear (ksMlEndLimiterEnum).

IMultiline – методы

AddLine – Добавить линию

Интерфейс...

Синтаксис Automation:

BOOL AddLine(double Offset, long Style);

Синтаксис COM:

HRESULT AddLine([in]double Offset,
[in]long Style,
[out, retval] VARIANT_BOOL * PVal);

Входные параметры:

Offset	- смещение линии относительно базового контура,
Style	- стиль линии из перечисления ksCurveStyleEnum.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

DeleteLine – Удалить линию

Интерфейс...

Синтаксис Automation:

BOOL DeleteLine(long LineIndex);

Синтаксис COM:

HRESULT DeleteLine([in]long LineIndex,
[out, retval] VARIANT_BOOL * PVal);

Входные параметры:

LineIndex	- индекс линии.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

FindLine – Найти индекс линии по смещению

Интерфейс...

Синтаксис Automation:

long FindLine(double Offset);

Синтаксис COM:

HRESULT FindLine([in]double Offset,
[out, retval] long * Result);

Входные параметры:

Offset	- смещение линии относительно базового контура.

Возвращаемое значение:

- индекс линии, имеющей указанное смещение.

Интерфейс INurbs

Справка системы КОМПАС...

KOMPAS.chm::/149_17_1_Lomanaja.htm

Интерфейс NURBS-кривой.

Иерархия:

IDispatch
IKompasAPIObject
IDrawingObject
INurbs

Описание:

Интерфейс позволяет задавать параметры NURBS-кривой.

Примечание:

1. Интерфейс можно получить у коллекции NURBS-кривых, используя свойство INurbses::Nurbs или метод INurbses::Add.
2. После задания параметров NURBS-кривой требуется вызвать метод IDrawingObject::Update.

INurbs – свойства

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Closed = Object.Closed	Получить свойство(*)
Object.Closed = Closed	Установить свойство (*)
Closed = Object.GetClosed()	Получить свойство (**)
Object.SetClosed(Closed)	Установить свойство (**)

Синтаксис COM:

Object.get_Closed(&Closed)	Получить свойство
Object.put_Closed(Closed)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак замкнутости NURBS-кривой.

Degree – Порядок кривой

Интерфейс...

Тип данных: long

Синтаксис Automation:

Degree = Object.Degree	Получить свойство(*)
Object.Degree = Degree	Установить свойство (*)
Degree = Object.GetDegree()	Получить свойство (**)
Object.SetDegree(Degree)	Установить свойство (**)

Синтаксис COM:

Object.get_Degree(&Degree)	Получить свойство
Object.put_Degree(Degree)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать порядок NURBS-кривой.

Для изменения порядка NURBS-кривой нужно, чтобы количество точек было равно или больше устанавливаемого значения, см. Пример...

Periodic – Порядок кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Periodic = Object.Periodic	Получить свойство(*)
Periodic = Object.Periodic	Получить свойство (**)

Синтаксис COM:

Object.get_Degree(&Degree)	Получить свойство

Примечание:

1. Свойство позволяет получить признак периодичности NURBS-кривой.
2. Свойство доступно только для чтения.

PointsCount – Количество точек

Интерфейс...

Тип данных: long

Синтаксис Automation:

PointsCount = Object.PointsCount	=	Получить свойство(*)
PointsCount = Object.GetPointsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointsCount(&PointsCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество точек NURBS-кривой.
2. Свойство доступно только для чтения.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль линии NURBS-кривой.

INurbs – методы

AddPoint – Добавить точку

Интерфейс...

Синтаксис Automation:

BOOL AddPoint(long Index,
double X,
double Y
double Weight);

Синтаксис COM:

HRESULT AddPoint(long Index,
double X,
double Y,
double Weight,
BOOL * Result);

Входные параметры:

Index	- индекс добавляемой точки,
X, Y	- координаты добавляемой точки,
Weight	- вес точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить точку в NURBS-кривую.

Clear – Удалить все точки

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет удалить все точки NURBS-кривой.

DeletePoint – Удалить точку

Интерфейс...

Синтаксис Automation:

BOOL DeletePoint(long Index);

Синтаксис COM:

HRESULT DeletePoint(long Index,
BOOL * Result);

Входные параметры:

Index - индекс удаляемой точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет удалить заданную точку NURBS-кривой.

GetNurbsParams – Получить параметры NURBS-кривой

Интерфейс...

Синтаксис Automation:

BOOL GetNurbsParams(VARIANT * Points,
VARIANT * Weight,
VARIANT * Knots);

Синтаксис COM:

HRESULT GetNurbsParams(VARIANT * Points,
VARIANT * Weight,
VARIANT * Knots,
BOOL * Result);

Выходные параметры:

Points - массив SAFEARRAY | VT_R8 координат точек кривой,
Weight - массив SAFEARRAY | VT_R8 весов точек,
Knots - массив SAFEARRAY | VT_R8 узлов сплайна.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры NURBS-кривой в виде массивов SAFEARRAY I VT_R8.

GetPoint – Получить параметры точки

Интерфейс...

Синтаксис Automation:

```
BOOL GetPoint( long Index,
double * X,
double * Y,
double * Weight );
```

Синтаксис COM:

```
HRESULT GetPoint( long Index,
double * X,
double * Y,
double * Weight,
BOOL * Result );
```

Входные параметры:

Index	- индекс точки.

Выходные параметры:

x, y	- координаты точки,
Weight	- массив SAFEARRAY I VT_R8 весов точек.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры точки NURBS-кривой в виде массивов SAFEARRAY I VT_R8.

ReadFromFile – Прочитать файл с данными

Интерфейс...

Синтаксис Automation:

```
BOOL ReadFromFile( LPCTSTR FileName );
```

Синтаксис COM:

HRESULT ReadFromFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае удачи.

SetNurbsParams – Установить параметры NURBS-кривой

Интерфейс...

Синтаксис Automation:

BOOL SetNurbsParams(VARIANT Points,
VARIANT Weight,
VARIANT Knots,
long Degree,
BOOL Closed);

Синтаксис COM:

HRESULT SetNurbsParams(VARIANT Points,
VARIANT Weight,
VARIANT Knots,
long Degree,
BOOL Closed,
BOOL * Result);

Входные параметры:

Points	- массив SAFEARRAY | VT_R8 координат точек кривой,
Weight	- массив SAFEARRAY | VT_R8 весов точек,
Knots	- массив SAFEARRAY | VT_R8 узлов сплайна,
Degree	- порядок кривой,
Closed	- признак замкнутости (нужно задавать после задания точек).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры NURBS-кривой в виде массивов SAFEARRAY | VT_R8.

WriteToFile – Записать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(LPCTSTR FileName);

Синтаксис COM:

HRESULT WriteToFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс INurbsByPoints

Интерфейс NURBS-кривой по точкам.

Иерархия:

IDispatch
- IKompasAPIObject
  - IDrawingObject
    - INurbs
  - INurbsByPoints

Интерфейс можно получить у коллекции NUTBS-кривых, полученной с помощью свойства IDrawingContainer::NurbsesByPoints.

INurbsByPoints- свойства

CurvatureInPoint – Величина кривизны в указанной точке

Интерфейс...

Тип данных: double

Синтаксис Automation:

CurvatureInPoint Object.CurvatureInPoint(PointIndex)	=	Получить свойство(*)
Object.CurvatureInPoint(PointIndex) = CurvatureInPoint		Установить свойство (*)

CurvatureInPoint = Object.GetCurvatureInPoint(PointIndex)	Получить свойство (**)
Object.SetCurvatureInPoint(PointIndex, CurvatureInPoint)	Установить свойство (**)

Синтаксис COM:

Object.get_CurvatureInPoint(PointIndex, &CurvatureInPoint)	Получить свойство
Object.put_CurvatureInPoint(PointIndex, CurvatureInPoint)	Установить свойство

Входные параметры:

long	- PointIndex - индекс точки.

DerivativeAngleInPoint – Угол вектора производной в указанной точке

Интерфейс...

Тип данных: double

Синтаксис Automation:

DerivativeAngleInPoint = Object.DerivativeAngleInPoint(PointIndex)	Получить свойство(*)
Object.DerivativeAngleInPoint(PointIndex) = DerivativeAngleInPoint	Установить свойство (*)
DerivativeAngleInPoint = Object.GetDerivativeAngleInPoint(PointIndex)	Получить свойство (**)
Object.SetDerivativeAngleInPoint(PointIndex, DerivativeAngleInPoint)	Установить свойство (**)

Синтаксис COM:

Object.get_DerivativeAngleInPoint(PointIndex, &DerivativeAngleInPoint)	Получить свойство
Object.put_DerivativeAngleInPoint(PointIndex, DerivativeAngleInPoint)	Установить свойство

Входные параметры:

long	- PointIndex - индекс точки.

DerivativeLenghtInPoint – Длина вектора производной в указанной точке

Интерфейс...

Тип данных: double

Синтаксис Automation:

DerivativeLenghtInPoint = Object.DerivativeLenghtInPoint(PointIndex)	Получить свойство(*)
Object.DerivativeLenghtInPoint(PointIndex) = DerivativeLenghtInPoint	Установить свойство (*)
DerivativeLenghtInPoint = Object.GetDerivativeLenghtInPoint(PointIndex)	Получить свойство (**)
Object.SetDerivativeLenghtInPoint(PointIndex, DerivativeLenghtInPoint)	Установить свойство (**)

Синтаксис COM:

Object.get_DerivativeLenghtInPoint(PointIndex, &DerivativeLenghtInPoint)	Получить свойство
Object.put_DerivativeLenghtInPoint(PointIndex, DerivativeLenghtInPoint)	Установить свойство

Входные параметры:

long	- PointIndex - индекс точки.

PointConstraints – Вариант управления точкой сплайна

Интерфейс...

Тип данных: из перечисления ksNurbsByPointsPointConstraintsEnum

Синтаксис Automation:

PointConstraints = Object.PointConstraints(PointIndex)	=	Получить свойство(*)
Object.PointConstraints(PointIndex) = PointConstraints		Установить свойство (*)
PointConstraints = Object.GetPointConstraints(PointIndex)	=	Получить свойство (**)
Object.SetPointConstraints(PointIndex, PointConstraints)		Установить свойство (**)

Синтаксис COM:

Object.get_PointConstraints(PointIndex, &PointConstraints)	Получить свойство
Object.put_PointConstraints(PointIndex, PointConstraints)	Установить свойство

Входные параметры:

long	- PointIndex - индекс точки.

PointsBuildingType – Способ формирования точек сплайна

Интерфейс...

Тип данных: из перечисления ksNurbsByPointsBuildingTypeEnum

Синтаксис Automation:

PointsBuildingType = Object.PointsBuildingType	=	Получить свойство(*)
Object.PointsBuildingType = PointsBuildingType	=	Установить свойство (*)
PointsBuildingType = Object.GetPointsBuildingType()	=	Получить свойство (**)
Object.SetPointsBuildingType(PointsBuildingType)		Установить свойство (**)

Синтаксис COM:

Object.get_PointsBuildingType(&PointsBuildingType)	Получить свойство
Object.put_PointsBuildingType(PointsBuildingType)	Установить свойство

INurbsByPoints – методы

AddPointByParam – Добавить новую точку с параметрами

Интерфейс...

Синтаксис Automation:

BOOL AddPointByParam(long Index, double X, double Y,
ksNurbsByPointsPointConstraintsEnum ConstraintsType,
double * DerivativeLenght, double * DerivativeAngle, double * Curvature);

Синтаксис COM:

HRESULT AddPointByParam(long Index, double X, double Y,
ksNurbsByPointsPointConstraintsEnum ConstraintsType, double * DerivativeLenght,
double * DerivativeAngle, double * Curvature, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс добавляемой точки,
X	- координата точки по X,
Y	- координата точки по Y,
ConstraintsType	- вариант управления точкой сплайна,
DerivativeLenght	- длина вектора производной,
DerivativeAngle	- угол вектора производной,
Curvature	- величина кривизны.

GetPointParam – Получить параметры точек сплайна

Интерфейс...

Синтаксис Automation:

BOOL GetPointParam(long Index, double * X, double * Y,
ksNurbsByPointsPointConstraintsEnum * ConstraintsType, double * DerivativeLenght,
double * DerivativeAngle, double * Curvature);

Синтаксис COM:

HRESULT GetPointParam(long Index, double * X, double * Y,
ksNurbsByPointsPointConstraintsEnum * ConstraintsType, double * DerivativeLenght,
double * DerivativeAngle, double * Curvature, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X	- координата точки по X,
Y	- координата точки по Y,
Constrain tsType	- вариант управления точкой сплайна,
Derivative Lenght	- длина вектора производной,
Derivative Angle	- угол вектора производной,
Curvature	- величина кривизны.

Интерфейс IPoint

Справка системы КОМПАС...

KOMPAS.chm::/121_Glava10_Tochki.htm

Интерфейс точки.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IPoint

Примечание:

1. Интерфейс можно получить, используя свойство коллекции точек IPoints::Point или метод IPoints::Add.
2. После задания параметров окружности требуется вызвать метод IDrawingObject::Update.

IPoint – свойства

Angle – Угол для точки со стрелкой

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Angle( &Angle )          Получить свойство
Object.put_Angle(  Angle )          Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол для точки со стрелкой из перечисления ksAnnotativeTerminatorSignEnum.

Style – Стиль точки

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                Получить свойство(* )
Object.Style = Style                Установить свойство (* )
Style = Object.GetStyle()           Получить свойство (**)
Object.SetStyle( Style )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )          Получить свойство
Object.put_Style(  Style )          Установить свойство
```

Примечание:

1. Свойство позволяет устанавливать и получать стиль точки.
2. Стили точек берутся из перечисления ksAnnotationSymbolEnum.

X – Координата точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X = Object.X                        Получить свойство(* )
Object.X = X                        Установить свойство (* )
X = Object.GetX()                   Получить свойство (**)
Object.SetXc( X )                   Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X( &X )                  Получить свойство
Object.put_X( X )                   Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату точки по оси X.

Y – Координата точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки по оси Y.

Интерфейс IPolyLine2D

Справка системы КОМПАС...

KOMPAS.chm::/149_17_1_Lomanaja.htm

Интерфейс полилинии (ломаной).

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IPolyLine2D

Описание:

Интерфейс позволяет задавать параметры ломаной.

Примечание:

1. Интерфейс можно получить у коллекции полилиний, используя свойство IPolyLines2D::PolyLine2D или метод IPolyLines2D::Add.
2. После задания параметров полилинии требуется вызвать метод IDrawingObject::Update.

IPolyLine2D – свойства

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Closed = Object.Closed	Получить свойство(*)
Object.Closed = Closed	Установить свойство (*)
Closed = Object.GetClosed()	Получить свойство (**)
Object.SetClosed(Closed)	Установить свойство (**)

Синтаксис COM:

Object.get_Closed(&Closed)	Получить свойство
Object.put_Closed(Closed)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак замкнутости полилинии.

PointsCount – Количество точек

Интерфейс...

Тип данных: long

Синтаксис Automation:

PointsCount = Object.PointsCount	=	Получить свойство(*)
PointsCount = Object.GetPointsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointsCount(&PointsCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество точек полилинии.
2. Свойство доступно только для чтения.

Points – Массив SAFEARRAY точек полилинии

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

Points = Object.Points	Получить свойство(*)
Object.Points = Points	Установить свойство (*)
Points = Object.GetPoints()	Получить свойство (**)
Object.SetPoints(Points)	Установить свойство (**)

Синтаксис COM:

Object.get_Points(&Points)	Получить свойство
Object.put_Points(Points)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив координат точек полилинии.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль линии.

IPolyLine2D – методы

AddPoint – Добавить точку

Интерфейс...

Синтаксис Automation:

BOOL AddPoint(long Index,
double X,
double Y);

Синтаксис COM:

HRESULT AddPoint(long Index,
double X,
double Y,
BOOL * Result);

Входные параметры:

Index	- индекс добавляемой точки,
X, Y	- координаты добавляемой точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить точку в полилинию.

Clear – Удалить все точки

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все точки полилинии.

DeletePoint – Удалить точку

Интерфейс...

Синтаксис Automation:

BOOL DeletePoint(long Index);

Синтаксис COM:

HRESULT DeletePoint(long Index);

Входные параметры:

Index	- индекс удаляемой точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить заданную точку полилинии.

GetPoint – Получить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long Index,
double * X,
double * Y);

Синтаксис COM:

HRESULT GetPoint(long Index,
double * X,
double * Y,
BOOL * Result);

Входные параметры:

Index	- индекс точки.

Выходные параметры:

X, Y	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры точки полилинии, заданной по индексу.

ReadFromFile – Прочитать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL ReadFromFile(LPCTSTR FileName);

Синтаксис COM:

HRESULT ReadFromFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае удачи.

WriteToFile – Записать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(LPCTSTR FileName);

Синтаксис COM:

HRESULT WriteToFile(BSTR FileName, BOOL * Result);

Входные параметры:

FileName	- полное имя файла с данными.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс IRectangle

Справка системы КОМПАС...

`KOMPAS.chm::/Prjamougolqnik.htm`

Интерфейс прямоугольника.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IRectangle

Описание:

Интерфейс позволяет задавать параметры прямоугольника.

Примечание:

1. Интерфейс можно получить у коллекции прямоугольников, используя свойство IRectangles::Rectangle или метод IRectangles::Add.
2. После задания параметров прямоугольника требуется вызвать метод IDrawingObject::Update.

IRectangle – свойства

Angle – Угол наклона стороны прямоугольника, выходящей из базовой точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол наклона прямоугольника.

ContourSegmentsPoints – Координаты точек сегментов контура многоугольника

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ContourSegmentsPoints = Object.ContourSegmentsPoints	Получить свойство(*)
ContourSegmentsPoints = Object.GetContourSegmentsPoints()	Получить свойство (**)

Синтаксис COM:

Object.get_ContourSegmentsPoints(&ContourSegmentsPoints)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Height – Высота прямоугольника

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство(*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight()	Получить свойство (**)

```
Object.SetHeight( Height )                    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Height( &Height )    Получить свойство
Object.put_Height(  Height )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать высоту прямоугольника.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                Получить свойство(* )
Object.Style = Style                Установить свойство (* )
Style = Object.GetStyle()           Получить свойство (**)
Object.SetStyle( Style )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )    Получить свойство
Object.put_Style(  Style )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать стиль линии прямоугольника.

Width – Ширина прямоугольника

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width                Получить свойство(* )
Object.Width = Width                Установить свойство (* )
Width = Object.GetWidth()           Получить свойство (**)
Object.SetWidth( Width )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width )    Получить свойство
Object.put_Width(  Width )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать ширину прямоугольника.

X – Координата базовой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки прямоугольника по оси X.

Y – Координата базовой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки прямоугольника по оси Y.

IRectangle – методы

GetCorner – Получить параметры скругления/усечения угла

Интерфейс...

Синтаксис Automation:

BOOL GetCorner(long Index,
long * Type,
double * L1,
double * L2);

Синтаксис COM:

HRESULT GetCorner(long Index,
ksCornerTypeEnum * Type,
double * L1,
double * L2,
BOOL* Result);

Входные параметры:

Index	- индекс угла.

Выходные параметры:

Type	- тип угла из перечисления ksCornerTypeEnum,
L1	- длина фаски первого сегмента или радиус скругления,
L2	- длина фаски второго сегмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры скругления/усечения угла прямоугольника.

SetCorner – Установить параметры скругления/усечения угла

Интерфейс...

Синтаксис Automation:

BOOL SetCorner(long Index,
long Type,
double L1,
double L2);

Синтаксис COM:

HRESULT SetCorner(long Index,
ksCornerTypeEnum Type,
double L1,

double L2,
BOOL * Result);

Входные параметры:

Index	- индекс угла,
Type	- тип угла из перечисления ksCornerTypeEnum,
L1	- длина фаски первого сегмента или радиус скругления,
L2	- длина фаски второго сегмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры скругления/усечения угла прямоугольника.

Интерфейс IRegularPolygon

Справка системы КОМПАС...

`KOMPAS.chm::/Mnogougolqnik.htm`

Интерфейс многоугольника.

Иерархия:

IDispatch
    IKompasAPIObject
        IDrawingObject
            IRegularPolygon

Описание:

Интерфейс позволяет задавать параметры многоугольника.

Примечание:

1. Интерфейс можно получить у коллекции многоугольников, используя свойство IRegularPolygons::RegularPolygon или метод IRegularPolygons::Add.
2. После задания параметров многоугольника требуется вызвать метод IDrawingObject::Update.

IRegularPolygon – свойства

Angle – Угол радиус-вектора, направленного от центра к первой вершине

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол радиус-вектора, направленного от центра к первой вершине.

Count – Количество вершин многоугольника

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство(*)
Count = Object.GetCount()	Получить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство

Примечание:

1. Свойство позволяет получить количество вершин многоугольника.
2. Свойство доступно только для чтения.

Describe – Признак построения по описанной или вписанной окружности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Describe = Object.Describe	Получить свойство(*)
Object.Describe = Describe	Установить свойство (*)
Describe = Object.GetDescribe()	Получить свойство (**)
Object.SetDescribe(Describe)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Describe( &Describe )        Получить свойство
Object.put_Describe(  Describe )        Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать признак построения по описанной или вписанной окружности.

Radius – Радиус вписанной или описанной окружности

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius                  Получить свойство(* )
Object.Radius = Radius                  Установить свойство (* )
Radius = Object.GetRadius()             Получить свойство (**)
Object.SetRadius( Radius )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius )            Получить свойство
Object.put_Radius(  Radius )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать радиус вписанной или описанной окружности.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                    Получить свойство(* )
Object.Style = Style                    Установить свойство (* )
Style = Object.GetStyle()               Получить свойство (**)
Object.SetStyle( Style )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )              Получить свойство
Object.put_Style(  Style )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать стиль линии многоугольника.

Xc – Координата точки центра вписанной или описанной окружности по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)
Object.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

Object.get_Xc(&Xc)	Получить свойство
Object.put_Xc(Xc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки центра вписанной или описанной окружности по оси X.

Yc – Координата точки центра вписанной или описанной окружности по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Yc = Object.Yc	Получить свойство(*)
Object.Yc = Yc	Установить свойство (*)
Yc = Object.GetYc()	Получить свойство (**)
Object.SetYc(Yc)	Установить свойство (**)

Синтаксис COM:

Object.get_Yc(&Yc)	Получить свойство
Object.put_Yc(Yc)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки центра вписанной или описанной окружности по оси Y.

IRegularPolygon – методы

GetCorner – Получить параметры скругления/усечения угла

Интерфейс...

Синтаксис Automation:

BOOL GetCorner(long Index,
long * Type,
double * L1,
double * L2);

Синтаксис COM:

HRESULT GetCorner(long Index,
ksCornerTypeEnum * Type,
double * L1,
double * L2,
BOOL* Result);

Входные параметры:

Index	- индекс угла.

Выходные параметры:

Type	- тип угла из перечисления ksCornerTypeEnum,
L1	- длина фаски первого сегмента или радиус скругления,
L2	- длина фаски второго сегмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры скругления/усечения угла многоугольника.

SetCorner – Установить параметры скругления/усечения угла

Интерфейс...

Синтаксис Automation:

BOOL SetCorner(long Index,
long * Type,

double * L1,
double * L2);

Синтаксис COM:

HRESULT SetCorner(long Index,
ksCornerTypeEnum * Type,
double * L1,
double * L2,
BOOL* Result);

Входные параметры:

Index	- индекс угла,
Type	- тип угла из перечисления ksCornerTypeEnum,
L1	- длина фаски первого сегмента или радиус скругления,
L2	- длина фаски второго сегмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры скругления/усечения угла многоугольника.

Интерфейс IRaster

Справка системы КОМПАС...

KOMPAS.chm::/353_Glava40_Ispolqzovanie_rastr.htm

Интерфейс растрового объекта.

Иерархия:

IDispatch
IKompasAPIObject
IDrawingObject
IRaster
IBoundariesObject

Описание:

Интерфейс позволяет задавать параметры растрового объекта.

Примечание:

1. Интерфейс можно получить у коллекции растровых объектов, используя свойство IRasters::Raster или метод IRasters::Add.
2. После задания параметров растрового объекта требуется вызвать метод IDrawingObject::Update.

3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс IBoundariesObject.
4. Для задания границы обрезки растра нужно использовать IBoundariesObject::AddBoundaries. В качестве границы обрезки может быть задан только один замкнутый контур

(подробнее о требованиях к границе обрезки...

KOMPAS.chm::/CM_CLIP.htm

IRaster – свойства

DisplayModePartial – Режим отрисовки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- показывать полностью,
FALSE	- показывать часть.

Синтаксис Automation:

DisplayModePartial = Object.DisplayModePartial	=	Получить свойство(*)
Object.DisplayModePartial = DisplayModePartial	=	Установить свойство (*)
DisplayModePartial = Object.GetDisplayModePartial()	=	Получить свойство (**)
Object.SetDisplayModePartial(DisplayModePartial)		Установить свойство (**)

Синтаксис COM:

Object.get_DisplayModePartial(&DisplayModePartial)	Получить свойство
Object.put_DisplayModePartial(DisplayModePartial)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать режим отрисовки растрового объекта.

FileName – Имя файла растрового объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство(*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя файла растрового объекта.

Height – Высота вставки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство(*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight()	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height)	Получить свойство
Object.put_Height(Height)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать высоту вставки растрового объекта.

InsertionType – Способ вставки растрового объекта

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- внедрить в документ,(*)
FALSE	- вставлять ссылкой.

Синтаксис Automation:

InsertionType Object.InsertionType	=	Получить свойство(*)
Object.InsertionType InsertionType	=	Установить свойство (*)
InsertionType Object.GetInsertionType()	=	Получить свойство (**)
Object.SetInsertionType(InsertionType)		Установить свойство (**)

Синтаксис COM:

Object.get_InsertionType(&InsertionType)	Получить свойство
Object.put_InsertionType(InsertionType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ вставки растрового объекта.

IsCutBoundarySet – Признак задания границы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCutBoundarySet Object.IsCutBoundarySet	=	Получить свойство(*)
IsCutBoundarySet Object.GetIsCutBoundarySet()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsCutBoundarySet(&IsCutBoundarySet)	Получить свойство

Примечание:

1. Свойство позволяет получать признак задания границы растрового объекта.
2. Свойство доступно только для чтения.

IsEnableChangeResolution – Доступность изменения разрешения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsEnableChangeResolution = Object.IsEnableChangeResolution	Получить свойство(*)
IsEnableChangeResolution = Object.GetIsEnableChangeResolution()	Получить свойство (**)

Синтаксис COM:

Object.get_IsEnableChangeResolution(&IsEnableChangeResolution)	Получить свойство

Примечание:

1. Свойство позволяет получать признак доступности изменения разрешения растрового объекта.
2. Свойство доступно только для чтения.

Palette – Цветовая палитра (Количество разрядов)

Интерфейс...

Тип данных: long

Синтаксис Automation:

Palette = Object.Palette	Получить свойство(*)
Palette = Object.GetPalette()	Получить свойство (**)

Синтаксис COM:

Object.get_Palette(&Palette)	Получить свойство

Примечание:

1. Свойство позволяет получать количество разрядов цветовой палитры растрового объекта.
2. Свойство доступно только для чтения

Resolution – Разрешение растра

Интерфейс...

Тип данных: double

Синтаксис Automation:

Resolution = Object.Resolution	Получить свойство(*)
Object.Resolution = Resolution	Установить свойство (*)

Resolution	=	Получить свойство (**)
Object.GetResolution()		
Object.SetResolution(Resolution)		Установить свойство (**)

Синтаксис COM:

Object.get_Resolution(&Resolution)	Получить свойство
Object.put_Resolutione(Resolution)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать разрешение растра.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство(*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать масштаб растрового объекта.

SourceHeight – Высота источника

Интерфейс...

Тип данных: double

Синтаксис Automation:

SourceHeight	=	Получить свойство(*)
Object.SourceHeight		
SourceHeight	=	Получить свойство (**)
Object.GetSourceHeight()		

Синтаксис COM:

Object.get_SourceHeight(&SourceHeight)	Получить свойство

Примечание:

1. Свойство позволяет получать значение высоты источника.
2. Свойство доступно только для чтения.

SourceWidth – Ширина источника

Интерфейс...

Тип данных: double

Синтаксис Automation:

SourceWidth = Object.SourceWidth	=	Получить свойство(*)
SourceWidth = Object.GetSourceWidth()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SourceWidth(&SourceWidth)	Получить свойство

Примечание:

1. Свойство позволяет получать значение ширины источника.
2. Свойство доступно только для чтения.

Width – Ширина вставки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Width = Object.Width	Получить свойство(*)
Object.Width = Width	Установить свойство (*)
Width = Object.GetWidth()	Получить свойство (**)
Object.SetWidth(Width)	Установить свойство (**)

Синтаксис COM:

Object.get_Width(&Width)	Получить свойство
Object.put_Width(Width)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать ширину вставки растрового объекта.

IRaster – методы

GetPlacement – Получить местоположение растрового объекта

Интерфейс...

Синтаксис Automation:

BOOL GetPlacement(double * X,
double * Y,
double * Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

HRESULT GetPlacement(double * X,
double * Y,
double * Angle,
BOOL * MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет получить точку привязки и угол поворота растрового объекта.
2. После применения к растровому объекту операции Симметрия его признак зеркальной симметрии автоматически инвертируется.

SetPlacement – Установить местоположение растрового объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement(double X,
double Y,
double Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

```
HRESULT SetPlacement( double X,
double Y,
double Angle,
BOOL MirrorSymmetry,
BOOL * Result );
```

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота объекта,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет установить точку привязки и угол поворота растрового объекта.
2. После применения к растровому объекту операции Симметрия его признак зеркальной симметрии автоматически инвертируется.

Интерфейс IContour

Интерфейс контура.

Иерархия:

IDispatch
    IKompasAPIObject
        IContour

Описание:

Позволяет создавать и редактировать контур.

Примечание:

1. Интерфейс можно получить у интерфейса мультилинии с помощью свойства IMultiline::BaseContour.
2. Интерфейс IContour является дополнительным для интерфейса IDrawingContour. Его можно получить с помощью метода IUnknown::QueryInterface.
3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update у мультилинии.

IContour – свойства

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- контур замкнут,
FALSE	- контур разомкнут.

Синтаксис Automation:

Closed = iObject.Closed;	Получить свойство(*)
iObject.Closed = Closed;	Установить свойство (*)
Closed = iObject.GetClosed();	Получить свойство (**)
iObject.SetClosed(Closed);	Установить свойство (**)

Синтаксис COM:

iObject->get_Closed(&Closed)	Получить свойство
iObject->put_Closed(Closed)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать замкнутость мультилинии.
2. Если мультилиния замкнута, то контур тоже замкнут.

Count – Количество сегментов

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = iObject.Count();	Получить свойство(*)
Count = iObject.GetCount();	Получить свойство (**)

Синтаксис COM:

iObject->get_Count(&Count)	Получить свойство

Примечание:

1. Свойство позволяет получить количество сегментов контура.
2. Свойство доступно только для чтения.

Segment – Получить сегмент

Интерфейс...

Тип данных: указатель на интерфейс сегмента контура IContourSegment

Синтаксис Automation:

Segment iObject.Segment(Index);	=	Получить свойство(*)
Segment iObject.GetSegment(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_Segment(Index, &Segment)	Получить свойство

Примечание:

1. Свойство позволяет получить сегмент контура.
2. Свойство доступно только для чтения.

TmpObjects – Временные объекты из сегментов в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Значения свойства:

Массив SafeArray типа VT_ARRAY | VT_DISPATCH.

Синтаксис Automation:

TmpObjects iObject.TmpObjects;	=	Получить свойство(*)
TmpObjects iObject.GetTmpObjects();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_TmpObjects(&TmpObjects)	Получить свойство

Примечание:

1. Свойство позволяет получить массив SAFEARRAY временных объектов, входящих в контур.
2. Свойство доступно только для чтения.

IContour – методы

AddSegment – Добавить сегмент

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(long type);

Синтаксис COM:

HRESULT Add([in]ksContourSegmentEnum Type,
IMultiline ** Result);

Входные параметры:

type	- тип сегмента.

Возвращаемое значение:

	- указатель на интерфейс сегмента контура IContourSegment.

Примечание:

Метод позволяет добавить сегмент в контур.

Clear – Очистить контур

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear([out, retval] VARIANT_BOOL * PVal);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все сегменты контура.

CopyCurve – Добавить сегмент

Интерфейс...

Синтаксис Automation:

BOOL CopyCurve(LPDISPATCH Curve,
BOOL DeleteSource);

Синтаксис COM:

HRESULT CopyCurve([in] IDrawingObject * Curve,
[in] VARIANT_BOOL DeleteSource,
[out, retval] VARIANT_BOOL* Result);

Входные параметры:

Curve	- объект для добавления в контур,
DeleteSource	- удалить исходный объект.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

После добавления сегментов в контур требуется вызов метода IDrawingObject::Update, в котором эти сегменты будут добавлены в 2D кривую. (При условии совпадения точек сегментов с точностью до 1e-3).

CopySegments – Объекты вида для добавления в контур в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Синтаксис Automation:

BOOL CopySegments(VARIANT Val,
BOOL DeleteSource);

Синтаксис COM:

HRESULT CopySegments([in] VARIANT Val,
[in] VARIANT_BOOL DeleteSource,
[out, retval] VARIANT_BOOL * PVal);

Входные параметры:

Val	- массив SAFEARRAY типа VT_ARRAY | VT_DISPATCH, содержащий объекты вида (отрезки, дуги и т. п.) для добавления в контур,
DeleteSource	- удалить исходные объекты.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

RemoveContourGaps – Устранить разрывы в контуре

Интерфейс...

Синтаксис Automation:

BOOL RemoveContourGaps(double Accuracy, BOOL CanInsert, BOOL CanReplace);

Синтаксис COM:

HRESULT RemoveContourGaps(double Accuracy, BOOL CanInsert, BOOL CanReplace, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входные параметры:

Accuracy	- размер разрывов,
CanInsert	- разрешение на вставку сегментов,
CanReplace	- разрешение на подмену сегментов.

Интерфейс IContourSegment

Интерфейс сегмента контура.

Иерархия:

IDispatch

IKompasAPIObject

IContourSegment

Описание:

Интерфейс позволяет редактировать сегмент контура.

Примечание:

1. Интерфейс можно получить у интерфейса контура с помощью свойства IContour::Segment или метода IContour::AddSegment.
2. Имеет следующие дополнительные интерфейсы, которые можно получить с помощью метода IUnknown::QueryInterface:

- IContourLineSegment,
- IContourArc.

IContourSegment – свойства

Curve2D – Математическая кривая

Интерфейс...

Тип данных: указатель на интерфейс математической кривой ICurve2D

Синтаксис Automation:

Curve2D = iObject.Curve2D();	Получить свойство(*)
Curve2D = iObject.GetCurve2D();	Получить свойство (**)

Синтаксис COM:

iObject->get_Curve2D(&Curve2D)	Получить свойство

Примечание:

1. Свойство позволяет получить математическую кривую.
2. Свойство доступно только для чтения.

SegmentType – Тип сегмента

Интерфейс...

Тип данных: из перечисления ksContourSegmentEnum

Синтаксис Automation:

SegmentType iObject.SegmentType();	=	Получить свойство(*)
SegmentType iObject.GetSegmentType();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SegmentType(&SegmentType)	Получить свойство

Примечание:

1. Свойство позволяет получить тип сегмента контура.
2. Свойство доступно только для чтения.

Интерфейс IContourArc

Интерфейс сегмента контура – дуги.

Иерархия:

IDispatch

IKompasAPIObject

IContourSegment

IContourArc

Описание:

Интерфейс позволяет создавать и редактировать сегмент контура – дугу.

Примечание:

Интерфейс можно получить следующими способами:

- у интерфейса контура с помощью свойства IContour::Segment или метода IContour::AddSegment.
- у интерфейса IContourSegment с помощью метода IUnknown::QueryInterface.

IContourArc – свойства

Angle1 – Угол первой точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle1 = iObject.Angle1();                  Получить свойство(* )
iObject.Angle1() = Angle1;                  Установить свойство (* )
Angle1 = iObject.GetAngle1();               Получить свойство (**)
iObject.SetAngle1( Angle1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle1( &Angle1 )              Получить свойство
iObject->put_Angle1( Angle1 )               Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать угол первой точки дуги.

Angle2 – Угол второй точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle2 = iObject.Angle2();                  Получить свойство(* )
iObject.Angle2() = Angle2;                  Установить свойство (* )
Angle2 = iObject.GetAngle2();               Получить свойство (**)
iObject.SetAngle2( Angle2 );                Установить свойство (**)
```

Синтаксис COM:

iObject->get_Angle2(&Angle2)	Получить свойство
iObject->put_Angle2(Angle2)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать угол второй точки дуги.

Direction – Направление

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- по часовой стрелке,
FALSE	- против часовой стрелки.

Синтаксис Automation:

Direction = iObject.Direction();	Получить свойство(*)
iObject.Direction() = Direction;	Установить свойство (*)
Direction = iObject.GetDirection();	Получить свойство (**)
iObject.SetDirection(Direction);	Установить свойство (**)

Синтаксис COM:

iObject->get_Direction(&Direction)	Получить свойство
iObject->put_Direction(Direction)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать направление дуги.

Radius – Радиус дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = iObject.Radius();	Получить свойство(*)
iObject.Radius() = Radius;	Установить свойство (*)

```
Radius =                    Получить свойство (**)
iObject.GetRadius();
iObject.SetRadius( Radius   Установить свойство (**)
);
```

Синтаксис COM:

```
iObject->get_Radius(        Получить свойство
&Radius )
iObject->put_Radius(        Установить свойство
Radius )
```

Примечание.

Свойство позволяет устанавливать и получать радиус дуги.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1();          Получить свойство(* )
iObject.X1() = X1;          Установить свойство (* )
X1 = iObject.GetX1();       Получить свойство (**)
iObject.SetX1( X1 );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1( &X1 )      Получить свойство
iObject->put_X1( X1 )       Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату первой точки дуги по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2();          Получить свойство(* )
iObject.X2() = X2;          Установить свойство (* )
X2 = iObject.GetX2();       Получить свойство (**)
iObject.SetX2( X2 );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2( &X2 )      Получить свойство
```

```
iObject->put_X2( X2 )          Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату второй точки дуги по оси X.

X3 – Координата третьей точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = iObject.X3();             Получить свойство(* )
iObject.X3() = X3;             Установить свойство (* )
X3 = iObject.GetX3();          Получить свойство (**)
iObject.SetX3( X3 );           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X3( &X3 )         Получить свойство
iObject->put_X3( X3 )          Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату третьей точки дуги по оси X.

Xc – Координата центра по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = iObject.Xc();             Получить свойство(* )
iObject.Xc() = Xc;             Установить свойство (* )
Xc = iObject.GetXc();          Получить свойство (**)
iObject.SetXc( Xc );           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X3( &X3 )         Получить свойство
iObject->put_Xc( Xc )          Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату центра дуги по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1();              Получить свойство(* )
iObject.Y1() = Y1;              Установить свойство (* )
Y1 = iObject.GetY1();           Получить свойство (**)
iObject.SetY1( Y1 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1( &Y1 )          Получить свойство
iObject->put_Y1( Y1 )           Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату первой точки дуги по оси X.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2();              Получить свойство(* )
iObject.Y2() = Y2;              Установить свойство (* )
Y2 = iObject.GetY2();           Получить свойство (**)
iObject.SetY2( Y2 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2( &Y2 )          Получить свойство
iObject->put_Y2( Y2 )           Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату второй точки дуги по оси X.

Y3 – Координата третьей точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y3 = iObject.Y3();              Получить свойство(* )
iObject.Y3() = Y3;              Установить свойство (* )
Y3 = iObject.GetY3();           Получить свойство (**)
iObject.SetY3( Y3 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y3( &Y3 )          Получить свойство
iObject->put_Y3( Y3 )           Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату третьей точки дуги по оси X.

Yc – Координата центра по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = iObject.Yc();          Получить свойство(* )
iObject.Yc() = Yc;          Установить свойство (* )
Yc = iObject.GetYc();       Получить свойство (**)
iObject.SetYc( Yc );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y3( &Y3 )      Получить свойство
iObject->put_Yc( Yc )       Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату центра дуги по оси X.

Интерфейс IContourLineSegment

Интерфейс сегмента контура – отрезка.

Иерархия:

IDispatch

IKompasAPIObject

IContourSegment

IContourLineSegment

Описание:

Интерфейс позволяет создавать и редактировать сегмент контура – отрезок.

Примечание:

Интерфейс можно получить следующими способами:

- у интерфейса контура с помощью свойства IContour::Segment или метода IContour::AddSegment.
- у интерфейса IContourSegment с помощью метода IUnknown::QueryInterface.

IContourLineSegment – свойства

Angle – Угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = iObject.Angle();        Получить свойство(* )
iObject.Angle() = Angle;        Установить свойство (* )
Angle = iObject.GetAngle();     Получить свойство (**)
iObject.SetAngle( Angle );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle( &Angle )        Получить свойство
iObject->put_Angle( Angle )         Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать угол наклона отрезка.

Length – Длина.

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = iObject.Length();          Получить свойство(* )
iObject.Length() = Length;          Установить свойство (* )
Length = iObject.GetLength();       Получить свойство (**)
iObject.SetLength( Length );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Length(        Получить свойство
&Length )
iObject->put_Length(        Установить свойство
Length )
```

Примечание.

Свойство позволяет устанавливать и получать длину отрезка.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1();          Получить свойство(* )
iObject.X1() = X1;          Установить свойство (* )
X1 = iObject.GetX1();       Получить свойство (**)
iObject.SetX1( X1 );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1( &X1 )      Получить свойство
iObject->put_X1( X1 )       Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату X первой точки отрезка.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2();          Получить свойство(* )
iObject.X2() = X2;          Установить свойство (* )
X2 = iObject.GetX2();       Получить свойство (**)
iObject.SetX21( X2 );       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2( &X2 )      Получить свойство
iObject->put_X2( X2 )       Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату X второй точки отрезка.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1();          Получить свойство(* )
iObject.Y1() = Y1;          Установить свойство (* )
Y1 = iObject.GetY1();       Получить свойство (**)
iObject.SetY1( Y1 );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1( &Y1 )      Получить свойство
iObject->put_Y1( Y1 )       Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату Y первой точки отрезка.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2();              Получить свойство(* )
iObject.Y2() = Y2;              Установить свойство (* )
Y2 = iObject.GetY2();           Получить свойство (**)
iObject.SetY2( Y2 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2( &Y2 )          Получить свойство
iObject->put_Y2( Y2 )           Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать координату Y второй точки отрезка.

Интерфейс ICurve2D

Интерфейс математической кривой 2D.

Иерархия:

IDispatch

IKompasAPIObject

ICurve2D

Описание:

Позволяет создавать и редактировать математическую кривую 2D.

Примечание:

Интерфейс можно получить у интерфейса сегмента контура с помощью свойства IContourSegment::Curve2D.

ICurve2D – свойства

IsClosed – Замкнутость кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsClosed = Object.IsClosed()          Получить свойство(* )
IsClosed = Object.GetIsClosed()       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsClosed( &IsClosed )      Получить свойство
```

Примечание:

1. Свойство позволяет получать признак замкнутости кривой.
2. Свойство доступно только для чтения.

IsSelfIntersect – Самопересечение кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsSelfIntersect Object.IsSelfIntersect()	=	Получить свойство(*)
IsSelfIntersect Object.GetIsSelfIntersect()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsSelfIntersect(&IsSelfIntersect)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Length – Длина кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

Length = Object.Length()	Получить свойство(*)
Length = Object.GetLength()	Получить свойство (**)

Синтаксис COM:

Object.get_Length(&Length)	Получить свойство

Примечание:

1. Свойство позволяет получать длину кривой.
2. Свойство доступно только для чтения.

ParamMin – Минимальный параметр

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamMin = Object.ParamMin()		Получить свойство(*)
ParamMin Object.GetParamMin()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ParamMin(&ParamMin)	Получить свойство

Примечание:

1. Свойство позволяет получать минимальный параметр кривой.
2. Свойство доступно только для чтения.

ParamMax – Максимальный параметр

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamMax Object.ParamMaxLength()	=	Получить свойство(*)
ParamMax Object.GetParamMax()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ParamMax(&ParamMax)	Получить свойство

Примечание:

1. Свойство позволяет получать максимальный параметр кривой.
2. Свойство доступно только для чтения.

ICurve2D – методы

CalculatePolygonByStep – Получить массив равномерно расположенных на кривой точек

Интерфейс...

Синтаксис Automation:

VARIANT CalculatePolygonByStep(double Step);

Синтаксис COM:

HRESULT CalculatePolygonByStep(double Step,
VARIANT * Result);

Входные параметры:

Step	- шаг,
Angle	- угол нормали.

Возвращаемое значение:

Массив SAFEARRAY точек VT_ARRAY | VT_R8.

Примечание:

Метод позволяет получить массив равномерно расположенных на кривой точек.

CouplingCurvCurv – Расчет скруглений для двух кривых

Интерфейс...

Синтаксис Automation:

VARIANT CouplingCurvCurv(LPDISPATCH * Curve2,
double Radius);

Синтаксис COM:

HRESULT CouplingCurvCurv(ICurve2D * Curve2,
double Radius,
VARIANT * Result);

Входные параметры:

Curve2	- указатель на интерфейс второй кривой,
Radius	- радиус скругления.

Возвращаемое значение:

- Массив SAFEARRAY структуры сопряжения VT_ARRAY | VT_R8.

Примечание:

Метод позволяет рассчитать скругления для двух кривых.

GetDistancePointPoint – Получить расстояние между двумя точками на кривой

Интерфейс...

Синтаксис Automation:

```
double GetDistancePointPoint( double X1,
double Y1,
double X2,
double Y2 );
```

Синтаксис COM:

```
HRESULT GetDistancePointPoint( double  X1,
double  Y1,
double  X2,
double  Y2,
double * Result );
```

Входные параметры:

X1, Y1	- координаты первой точки,
X2, Y2	- координаты второй точки.

Возвращаемое значение:

- Расстояние между двумя точками.

Примечание:

Метод позволяет получить расстояние между двумя точками на кривой.

GetDistanceToPoint – Получить расстояние между точкой и проекцией точки на кривую

Интерфейс...

Синтаксис Automation:

```
double GetDistanceToPoint( double X,
double Y );
```

Синтаксис COM:

```
HRESULT GetDistanceToPoint( double  X,
double  Y,
double * Result );
```

Входные параметры:

X, Y	- координаты точки.

Возвращаемое значение:

- Расстояние между точкой и ее проекцией на кривую.

Примечание:

Метод позволяет получить расстояние между точкой и проекцией точки на кривую.

GetMetricLength – Метрическая длина кривой

Интерфейс...

Синтаксис Automation:

```
double GetMetricLength( double T1,
double T2 );
```

Синтаксис COM:

```
HRESULT GetMetricLength( double  T1,
double  T2,
double * Result );
```

Входные параметры:

T1, T2 - параметры кривой.

Возвращаемое значение:

- Метрическая длина кривой.

Примечание:

Метод позволяет получить метрическую длину кривой.

GetNurbsParams – Получить параметры в NURBS-представлении

Интерфейс...

Синтаксис Automation:

```
BOOL GetNurbsParams( BOOL UnClamped,
VARIANT * Points,
VARIANT * Weights,
VARIANT * Knots,
double  * TMin,
double  * TMax );
```

Синтаксис COM:

```
HRESULT GetNurbsParams( BOOL UnClamped,
VARIANT * Points,
VARIANT * Weights,
VARIANT * Knots,
```

double * TMin,
double * TMax,
BOOL * Result);

Входные параметры:

UnClampe d	- TRUE - вернуть параметры точек для замкнутого представления, - FALSE - вернуть параметры точек для разомкнутого представления.

Входные параметры:

Points	- SAFEARRAY координат точек VT_ARRAY | VT_R8,
Weights	- SAFEARRAY весов точек VT_ARRAY | VT_R8,
Knots	- SAFEARRAY узлов точек VT_ARRAY | VT_R8,
TMin	- минимальный параметр кривой,
TMax	- максимальный параметр кривой.

Возвращаемое значение:

TRU E	- в случае успеха,
FAL SE	- в случае неудачи.

Примечание:

Метод позволяет получить параметры кривой в NURBS-представлении.

GetPointLocation – Положение точки относительно кривой

Интерфейс...

Синтаксис Automation:

ksPointLocationTypeEnum GetPointLocation(double X, double Y);

Синтаксис COM:

HRESULT GetPointLocation(double X, double Y, ksPointLocationTypeEnum * Result);

Возвращаемое значение:

Res ult	- положение двумерной точки относительно двумерной кривой.

Входные параметры:

X	- первая координата точки,

Y - вторая координата точки.

Intersect – Пересечение двух кривых

Интерфейс...

Синтаксис Automation:

VARIANT Intersect(LPDISPATCH * Curve2);

Синтаксис COM:

HRESULT Intersect(ICurve2D * Curve2,
VARIANT * Result);

Входные параметры:

Curve2 - указатель на интерфейс кривой, которая пересекает данную кривую.

Возвращаемое значение:

- Массив SAFEARRAY точек пересечений VT_ARRAY | VT_R8

Примечание:

Метод позволяет получить координаты точек пересечения двух кривых.

MovePoint – Сдвинуть точку на расстояние Lenght по кривой

Интерфейс...

Синтаксис Automation:

BOOL MovePoint(double * X,
double * Y,
double Lenght,
BOOL Direction);

Синтаксис COM:

HRESULT MovePoint(double * X,
double * Y,
double Lenght,
BOOL Direction,
BOOL * Result);

Входные параметры:

X, Y - координаты точки,

Length	- расстояние, на которое нужно сместить точку,
Direction	- направление смещения.

Возвращаемое значение:

- Массив SAFEARRAY точек VT_ARRAY | VT_R8.

Примечание:

Метод позволяет получить массив равномерно расположенных на кривой точек.

PointOn – Получить координаты точки по параметру t

Интерфейс...

Синтаксис Automation:

```
BOOL PointOn( double T,
double * X,
double * Y );
```

Синтаксис COM:

```
HRESULT PointOn( double  T,
double * X,
double * Y,
BOOL  * Result );
```

Входные параметры:

T	- параметр кривой.

Выходные параметры:

X, Y	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить координаты точки по параметру кривой.

PointProjection – Проекция точки на кривую

Интерфейс...

Синтаксис Automation:

BOOL PointProjection(double X,
double Y,
double * Kx,
double * Ky,
double * T,
double * Angle);

Синтаксис COM:

HRESULT PointProjection(double X,
double Y,
double * Kx,
double * Ky,
double * T,
double * Angle,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки.

Выходные параметры:

Kx, Ky	- координаты проекции точки,
T	- параметр кривой,
Angle	- угол нормали.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет построить проекцию точки на кривую.

Tangent – Касательные для двух кривых

Интерфейс...

Синтаксис Automation:

BOOL Tangent(LPDISPATCH * Curve2,
VARIANT * Curve1Points,
VARIANT * Curve2Points);

Синтаксис COM:

HRESULT Tangent(ICurve2D * Curve2,
VARIANT * Curve1Points,

VARIANT * Curve2Points,
BOOL * Result);

Входные параметры:

Curve2	- указатель на интерфейс второй кривой.

Выходные параметры:

Curve1Points	- массив SAFEARRAY точек на первой кривой VT_ARRAY | VT_R8,
Curve2Points	- SAFEARRAY точек на второй кривой VT_ARRAY | VT_R8.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить касательные для двух кривых.

TangentLinePoint – Точки касания кривой и прямой из заданной точки

Интерфейс...

Синтаксис Automation:

VARIANT TangentLinePoint(double X,
double Y);

Синтаксис COM:

HRESULT TangentLinePoint(double X,
double Y,
VARIANT * Result);

Входные параметры:

X, Y	- координаты точки.

Возвращаемое значение:

- Массив SAFEARRAY точек на кривой VT_ARRAY | VT_R8.

Примечание:

Метод позволяет получить координаты точек касания кривой и прямой из заданной точки.

Группы

Интерфейс IDrawingGroup

Интерфейс группы (постоянной, временной, именованной).

Иерархия:

IDispatch

IKompasAPIObject

IDrawingGroup

Описание:

Интерфейс позволяет задавать параметры группы (постоянной, временной или именованной).

Примечание:

Интерфейс можно получить у коллекции групп, используя свойство IDrawingGroups::Item или метод IDrawingGroups::Add или у дополнительного интерфейса документа 2D с помощью свойства IKompasDocument2D1::CurrentGroup.

IDrawingGroup – свойства

Current – Текущая группа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Current = Object.Current	Получить свойство(*)
Current = Object.GetCurrent()	Получить свойство (**)

Синтаксис COM:

Object.get_Current(&Current)	Получить свойство

Примечание:

1. Свойство позволяет получать признак того, что группа является текущей.
2. Свойство доступно только для чтения.

Name – Имя именованной группы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = Object.Name                  Получить свойство(* )
Object.Name = Name                  Установить свойство (* )
Name = Object.GetName()             Получить свойство (**)
Object.SetName( Name )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Name( &Name )            Получить свойство
Object.put_Name(  Name )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать имя группы и добавляет группу в именованные.

Objects – Получить массив объектов SAFEARRAY|VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

```
Objects = Object.Objects            Получить свойство(* )
Objects = Object.GetObjects()       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Objects( &Objects )      Получить свойство
```

Примечание:

1. Свойство позволяет получать массив объектов, входящих в группу. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

ObjectsCount – Количество объектов в группе

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
ObjectsCount          =     Получить свойство(* )
Object.ObjectsCount
ObjectsCount          =     Получить свойство (**)
Object.GetObjectsCount()
```

Синтаксис COM:

Object.get_ObjectsCount(&ObjectsCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество объектов в группе.
2. Свойство доступно только для чтения.

IDrawingGroup – методы

AddObjects – Добавить объекты в группу

Интерфейс...

Синтаксис Automation:

BOOL AddObjects(VARIANT Objects);

Синтаксис COM:

HRESULT AddObjects(VARIANT Objects,
BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов для добавления в группу (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить объекты в группу.

AddRegion – Добавить объекты в группу рамкой

Интерфейс...

Синтаксис Automation:

BOOL AddRegion(long RegionType,
double XMin,
double YMin,
double XMax,
double YMax);

Синтаксис COM:

HRESULT AddRegion(ksRegionTypeEnum RegionType,
double XMin,
double YMin,
double XMax,
double YMax,
BOOL * Result);

Входные параметры:

RegionType	- тип рамки из перечисления ksRegionTypeEnum,
XMin, YMin	- координаты нижнего левого угла рамки,
XMax, YMax	- координаты верхнего правого угла рамки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить объекты в группу рамкой.

Clear – Очистить группу

Интерфейс...

Синтаксис Automation:

BOOL Clear(BOOL DeleteTmp);

Синтаксис COM:

HRESULT Clear(BOOL DeleteTmp,
BOOL * Result);

Входные параметры:

DeleteTmp	- удалять ли временные объекты.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все объекты из группы.

Close – Закрыть группу

Интерфейс...

Синтаксис Automation:

BOOL Close();

Синтаксис COM:

HRESULT Close(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет открыть группу для редактирования.

Delete – Удалить группу

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить группу.

DetachObjects – Исключить объекты из группы

Интерфейс...

Синтаксис Automation:

BOOL DetachObjects(VARIANT Objects,
BOOL DeleteTmp);

Синтаксис COM:

HRESULT DetachObjects(VARIANT Objects,
BOOL DeleteTmp,
BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов для удаления из группы(VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH),
DeleteTmp	- удалять ли временные объекты.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет исключить объекты из группы.

Draw – Отрисовать в окне

Интерфейс...

Синтаксис Automation:

BOOL Draw(OLE_HANDLE HWnd);

Синтаксис COM:

HRESULT Draw(OLE_HANDLE HWnd,
BOOL * Result);

Входные параметры:

HWnd	- дескриптор окна.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет отрисовать группу в окне.

ExistObject – Проверить, есть ли объект в группе

Интерфейс...

Синтаксис Automation:

BOOL ExistObject(LPDISPATCH Object);

Синтаксис COM:

HRESULT ExistObject(IDrawingObject * Object,
BOOL * Result);

Входные параметры:

Object	- объект, наличие которого в группе надо проверить.

Возвращаемое значение:

TRUE	- объект есть в группе,

FALSE - объекта нет в группе.

Примечание:

Метод позволяет определить, есть ли в группе заданный объект.

Open – Открыть группу

Интерфейс...

Синтаксис Automation:

BOOL Open();

Синтаксис COM:

HRESULT Open(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет открыть группу для редактирования.

ReadFragment – Добавить объекты в группу из фрагмента

Интерфейс...

Синтаксис Automation:

BOOL ReadFragment(BSTR FileName,
BOOL CurentLayer,
double XBase,
double YBase,
double Scale,
double Angle,
BOOL ScaleProjLinesSize);

Синтаксис COM:

HRESULT ReadFragment(BSTR FileName,
BOOL CurentLayer,
double XBase,
double YBase,
double Scale,
double Angle,
BOOL ScaleProjLinesSize,
BOOL * Result);

Входные параметры:

FileName - имя файла фрагмента,

CurentLayer - тип размещения по слоям
FALSE - на свои слои,
TRUE - в текущий слой,
XBase, YBase - координаты базовой точки в СК вида,
Scale - масштаб,
Angle - угол поворота в СК вида,
ScaleProjLinesSize - масштабировать ли проекционные линии.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет добавить объекты в группу из фрагмента.

ReadFromClip – Считать объекты из клипа

Интерфейс...

Синтаксис Automation:

BOOL ReadFromClip(BOOL AttrCopy,
BOOL SpcObjCopy);

Синтаксис COM:

HRESULT ReadFromClip(BOOL AttrCopy,
BOOL SpcObjCopy,
BOOL * Result);

Входные параметры:

AttrCopy - копировать ли атрибуты,
SpcObjCopy - копировать ли объекты спецификации.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет считать объекты из клипа в группу.

Store – Вставить временную группу в документ

Интерфейс...

Синтаксис Automation:

BOOL Store();

Синтаксис COM:

HRESULT Store(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет вставить временную группу в документ.

WriteFragment – Записать в фрагмент

Интерфейс...

Синтаксис Automation:

BOOL WriteFragment(BSTR FileName,
BSTR Comment,
double XBase,
double YBase);

Синтаксис COM:

HRESULT WriteFragment(BSTR FileName,
BSTR Comment,
double XBase,
double YBase,
BOOL * Result);

Входные параметры:

FileName	- имя файла фрагмента,
Comment	- комментарий,
XBase, YBase	- координаты базовой точки в СК вида,

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет записать группу в фрагмент.

WriteToClip – Записать объекты в клип

Интерфейс...

Синтаксис Automation:

BOOL WriteToClip(BOOL AttrCopy,
BOOL SpcObjCopy);

Синтаксис COM:

```
HRESULT WriteToClip( BOOL  AttrCopy,
BOOL  SpcObjCopy,
BOOL * Result );
```

Входные параметры:

AttrCopy	- копировать ли атрибуты,
SpcObjCopy	- копировать ли объекты спецификации.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет скопировать объекты из группы в клип.

Интерфейс IDrawingGroups

Интерфейс коллекции групп (именованных, рабочих временных и постоянных).

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingGroups

Описание:

Позволяет создавать и получать группы (именованные, рабочие временные и постоянные).

Примечание:

Данный интерфейс можно получить у дополнительного интерфейса документа 2D с помощью свойства IKompasDocument2D1::DrawingGroups.

IDrawingGroups – свойства

Item – Элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс группы IDrawingGroup

Синтаксис Automation:

```
Item = iObject.Item  ( Index );          Получить свойство(* )
Item = iObject.GetItem( Index );         Получить свойство (**)
```

Синтаксис COM:

iObject->get_Item(Index, &Item) Получить свойство

Примечание:

Свойство доступно только для чтения.

IDrawingGroups – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(BOOL temp,
BSTR Name);

Синтаксис COM:

HRESULT Add(BOOL temp,
BSTR Name,
IDrawingGroup ** Result);

Входные параметры:

temp - является ли группа временной,
Name - имя группы.

Возвращаемое значение:

- Указатель на интерфейс группы IDrawingGroup.

Листы и оформление

Интерфейс ILayoutSheets

Справка системы КОМПАС...

KOMPAS.chm::/1132_Glava133_Osnovnaja_nadpisq.htm

Интерфейс коллекции листов оформления.

Иерархия:

IKompasAPIObject

IKompasCollection

ILayoutSheets

Описание:

Управляет массивом интерфейсов листов оформления ILayoutSheet.

Примечание:

1. После вызова метода Add лист оформления будет добавлен с умолчательными параметрами (заданными в настройках). Для изменения параметров листа нужно изменить их у полученного объекта и вызывать ILayoutSheet::Update.
2. Данный интерфейс может быть получен от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::LayoutSheets.

ILayoutSheets – свойства

Item – Лист оформления, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ILayoutSheet листа оформления

Синтаксис Automation:

Sheet = iObject.Item (Index)	Получить свойство(*)
Sheet = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &Sheet)	(Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

ItemByNumber – Лист оформления, заданный по номеру

Интерфейс...

Тип данных: указатель на интерфейс ILayoutSheet листа оформления

Синтаксис Automation:

Sheet = iObject.ItemByNumber (Number)	Получить свойство(*)
Sheet = iObject.GetItemByNumber (Number)	Получить свойство (**)

Синтаксис COM:

iObject->get_ItemByNumber (Number, &Sheet)	Получить свойство

Входные параметры:

Number	- номер листа, отображается в основной надписи.

Примечание:

Свойство доступно только для чтения.

ILayoutSheets – методы

Add – Создать лист оформления (добавляет лист оформления в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add ([out, retval] ILayoutSheet ** Result);

Возвращаемое значение:

- Указатель на интерфейс листа оформления ILayoutSheet.

Примечание:

Лист оформления будет добавлен с умолчательными параметрами (заданными в настройках). Для изменения параметров листа в документе нужно задать его параметры и вызвать метод ILayoutSheet::Update.

Интерфейс ILayoutSheet

Справка системы КОМПАС...

`kompas.chm::/1132_Glava133_Osnovnaja_nadpisq.htm`

Интерфейс параметров листа оформления.

Иерархия:

IKompasAPIObject

ILayoutSheet

Описание:

В чертежно-графическом редакторе КОМПАС-График при работе с документами используется понятие оформления. Оформление включает основную надпись, а также внешнюю и внутреннюю рамки. Данный интерфейс позволяет выбрать оформление, которое будет использоваться для листов документа. Оформления документов хранятся в специальных системных библиотеках - файлах с расширением lyt.

Примечание:

Данный интерфейс можно получить у коллекции листов оформления ILayoutSheets с помощью свойства ILayoutSheets::Item, ILayoutSheets::ItemByNumber, метода ILayoutSheets::Add.

ILayoutSheet – свойства

Format – Формат листа

Интерфейс...

Тип данных: указатель на интерфейс ISheetFormat формата листа оформления

Синтаксис Automation:

Format = iObject.Format	Получить свойство(*)
Format = iObject.GetFormat()	Получить свойство (**)

Синтаксис COM:

iObject->get_Format (&Format)	Получить свойство

Примечание:

1. Данное свойство определяет формат листа оформления. После изменения значений свойств формата листа оформления, для того чтобы изменения вступили в силу, надо вызвать метод Update.
2. Свойство доступно только для чтения.

LayoutLibraryFileName – Имя файла библиотеки стилей оформления

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

LayoutLibraryFileName = iObject.LayoutLibraryFileName	Получить свойство(*)
iObject.LayoutLibraryFileName = LayoutLibraryFileName	Установить свойство (*)
LayoutLibraryFileName = iObject.GetLayoutLibraryFileName()	Получить свойство (**)
iObject.SetLayoutLibraryFileName (LayoutLibraryFileName)	Установить свойство (**)

Синтаксис COM:

iObject->get_LayoutLibraryFileName (&LayoutLibraryFileName)	Получить свойство
iObject->put_LayoutLibraryFileName (LayoutLibraryFileName)	Установить свойство

Примечание:

Данное свойство позволяет получить/изменить полное имя файла библиотеки стилей оформлений, из которой взят данный стиль. После изменения значений свойств формата листа оформления, для того чтобы изменения вступили в силу, надо вызвать метод Update.

LayoutStyleNumber – Номер стиля оформления

Интерфейс...

Тип данных: double

Синтаксис Automation:

LayoutStyleNumber = iObject.LayoutStyleNumber		Получить свойство(*)
iObject.LayoutStyleNumber = LayoutStyleNumber		Установить свойство (*)
LayoutStyleNumber = iObject.GetLayoutStyleNumber()		Получить свойство (**)
iObject.SetLayoutStyleNumber (LayoutStyleNumber)		Установить свойство (**)

Синтаксис COM:

iObject->get_LayoutStyleNumber (&LayoutStyleNumber)	Получить свойство
iObject->put_LayoutStyleNumber (LayoutStyleNumber)	Установить свойство

Примечание:

Данное свойство позволяет получить\изменить номер используемого стиля в библиотеке стилей оформлений (см. LayoutLibraryFileName). После изменения значений свойств формата листа оформления, для того чтобы изменения вступили в силу, надо вызвать метод Update.

SheetType – Тип листа

Интерфейс...

Тип данных: из перечисления ksSheetTypeEnum

Синтаксис Automation:

SheetType = iObject.SheetType		Получить свойство(*)
iObject.SheetType = SheetType		Установить свойство (*)
SheetType = iObject.GetSheetType()		Получить свойство (**)

iObject.SetSheetType (SheetType)	Установить свойство (**)

Синтаксис COM:

iObject->get_SheetType (&SheetType)	Получить свойство
iObject->put_SheetType (SheetType)	Установить свойство

Stamp – Основная надпись листа

Интерфейс...

Тип данных: указатель на интерфейс IStamp

Синтаксис Automation:

Stamp = Object.Stamp	Получить свойство(*)
Stamp = Object.GetStamp()	Получить свойство (**)

Синтаксис COM:

Object.get_Stamp(&Stamp)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс основной надписи.
2. Свойство доступно только для чтения.

ILayoutSheet – методы

Delete – Удалить лист

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Удаляет лист оформления из документа.

GetPlaceInsideFrames – Выдать свободное место внутри рамок

Интерфейс...

Синтаксис Automation:

BOOL GetPlaceInsideFrames(double* Left, double* Top, double* Right, double* Bottom);

Синтаксис COM:

HRESULT GetPlaceInsideFrames([out] double* Left, [out] double* Top, [out] double* Right, [out] double* Bottom, [out, retval] VARIANT_BOOL * Val);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Выходные параметры:

Left, Top	- координата верхней левой точки,
Right, Bottom	- координата нижней правой точки.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- обновление прошло успешно,
FALSE	- обновить настройки не удалось.

Примечание:

Все изменения, сделанные в данном интерфейсе и интерфейсе формата листа, вступят в силу только после вызова этого метода.

Интерфейс ISheetFormat

Справка системы КОМПАС...

kompas.chm::/1132_Glava133_Osnovnaja_nadpisq.htm

Интерфейс формата листа.

Иерархия:

IKompasAPIObject

ISheetFormat

Примечание:

1. Данный интерфейс позволяет получить и изменить текущий формат листа для документа.
2. Данный интерфейс может быть получен следующими способами:

- от интерфейса стиля спецификации ISpecificationStyle с помощью свойства ISpecificationStyle::Format,
- от интерфейса параметров листа оформления ILayoutSheet с помощью свойства ILayoutSheet::Format.

ISheetFormat – свойства

Format – Формат листа

Интерфейс...

Тип данных: из перечисления ksDocumentFormatEnum

Синтаксис Automation:

Format = iObject.Format	Получить свойство(*)
iObject.Format = Format	Установить свойство (*)
Format = iObject.GetFormat()	Получить свойство (**)
iObject.SetFormat (Format)	Установить свойство (**)

Синтаксис COM:

iObject->get_Format (&Format)	Получить свойство
iObject->put_Format (Format)	Установить свойство

FormatHeight – Высота пользовательского формата

Интерфейс...

Тип данных: long

Синтаксис Automation:

FormatHeight = iObject.FormatHeight	=	Получить свойство(*)
iObject.FormatHeight = FormatHeight	=	Установить свойство (*)
FormatHeight = iObject.GetFormatHeight()	=	Получить свойство (**)
iObject.SetFormatHeight (FormatHeight)		Установить свойство (**)

Синтаксис COM:

iObject->get_FormatHeight (&FormatHeight)	Получить свойство
iObject->put_FormatHeight (FormatHeight)	Установить свойство

Примечание:

Свойство используется только для пользовательских форматов листа (см. ISheetFormat::Format).

FormatMultiplicity – Кратность формата листа

Интерфейс...

Тип данных: long

Синтаксис Automation:

FormatMultiplicity = iObject.FormatMultiplicity	=	Получить свойство(*)
iObject.FormatMultiplicity = FormatMultiplicity	=	Установить свойство (*)
FormatMultiplicity = iObject.GetFormatMultiplicity()	=	Получить свойство (**)
iObject.SetFormatMultiplicity (FormatMultiplicity)		Установить свойство (**)

Синтаксис COM:

iObject->get_FormatMultiplicity (&FormatMultiplicity)	Получить свойство
iObject->put_FormatMultiplicity (FormatMultiplicity)	Установить свойство

Примечание:

1. Свойство используется только для стандартных форматов листа (см. ISheetFormat::Format).
2. Свойство не используется для текстовых документов.

FormatWidth – Ширина пользовательского формата

Интерфейс...

Тип данных: long

Синтаксис Automation:

FormatWidth = iObject.FormatWidth	=	Получить свойство(*)

iObject.FormatWidth = FormatWidth	=	Установить свойство (*)
FormatWidth = iObject.GetFormatWidth()	=	Получить свойство (**)
iObject.SetFormatWidth (FormatWidth)		Установить свойство (**)

Синтаксис COM:

iObject->get_FormatWidth (&FormatWidth)	Получить свойство
iObject->put_FormatWidth (FormatWidth)	Установить свойство

Примечание:

Свойство используется только для пользовательских форматов листа (см. ISheetFormat::Format).

VerticalOrientation – Вертикальная ориентация формата листа

Интерфейс...

Тип данных: long

Синтаксис Automation:

VerticalOrientation = iObject.VerticalOrientation	=	Получить свойство(*)
iObject.VerticalOrientation = VerticalOrientation	=	Установить свойство (*)
VerticalOrientation = iObject.GetVerticalOrientation()	=	Получить свойство (**)
iObject.SetVerticalOrientation (VerticalOrientation)		Установить свойство (**)

Синтаксис COM:

iObject->get_VerticalOrientation (&VerticalOrientation)	Получить свойство
iObject->put_VerticalOrientation (VerticalOrientation)	Установить свойство

Примечание:

Свойство используется только для стандартных форматов листа (см. ISheetFormat::Format).

Интерфейс ISpecRough

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Неуказанная шероховатость.

Иерархия:

IDispatch

IKompasAPIObject

ISpecRough

Описание:

Интерфейс позволяет задавать и получать параметры неуказанной шероховатости.

Примечание:

Интерфейс можно получить у интерфейса чертежа с помощью свойства IDrawingDocument::SpecRough.

ISpecRough – свойства

AddSign – Добавить знак в скобках

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AddSign = Object.AddSign	Получить свойство(*)
Object.AddSign = AddSign	Установить свойство (*)
AddSign = Object.GetAddSign()	Получить свойство (**)
Object.SetAddSign(AddSign)	Установить свойство (**)

Синтаксис COM:

Object.get_AddSign(&AddSign)	Получить свойство
Object.put_AddSign(AddSign)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак знака в скобках.

AutoPlacement – Авторазмещение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoPlacement = Object.AutoPlacement	=	Получить свойство(*)

Object.AutoPlacement = AutoPlacement	=	Установить свойство (*)
AutoPlacement = Object.GetAutoPlacement()	=	Получить свойство (**)
Object.SetAutoPlacement(AutoPlacement)		Установить свойство (**)

Синтаксис COM:

Object.get_AutoPlacement(&AutoPlacement)	Получить свойство
Object.put_AutoPlacement(AutoPlacement)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак авторазмещения неуказанной шероховатости.

Crossed – Признак необходимости перестроения неуказанной шероховатости

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Crossed = Object.Crossed	Получить свойство(*)
Crossed = Object.GetCrossed()	Получить свойство (**)

Синтаксис COM:

Object.get_Crossed(&Crossed)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

Distance – Расстояние до знака

Интерфейс...

Тип данных: double

Синтаксис Automation:

Distance = Object.Distance	Получить свойство(*)
Object.Distance = Distance	Установить свойство (*)
Distance = Object.GetDistance()	Получить свойство (**)
Object.SetDistance(Distance)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Distance( &Distance )        Получить свойство
Object.put_Distance(  Distance )        Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать расстояние до знака неуказанной шероховатости.

IsCreated – Признак отображения в документе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsCreated = Object.IsCreated        Получить свойство(* )
IsCreated                  =        Получить свойство (**)
Object.GetIsCreated()
```

Синтаксис COM:

```
Object.get_IsCreated(               Получить свойство
&IsCreated )
```

Примечание:

1. Свойство позволяет получать признак отображения неуказанной шероховатости в документе.
2. Свойство доступно только для чтения.

SignType – Тип значка

Интерфейс...

Тип данных: из перечисления ksRoughSignEnum

Синтаксис Automation:

```
SignType = Object.SignType          Получить свойство(* )
Object.SignType = SignType          Установить свойство (* )
SignType = Object.GetSignType()     Получить свойство (**)
Object.SetSignType( SignType )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SignType( &SignType )    Получить свойство
Object.put_SignType(  SignType )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать тип значка неуказанной шероховатости.

Text – Текст

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Text = Object.Text                  Получить свойство(* )
Object.Text = Text                  Установить свойство (* )
Text = Object.GetText()             Получить свойство (**)
Object.SetText( Text )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Text( &Text )            Получить свойство
Object.put_Text(  Text )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать текст неуказанной шероховатости.

X – Координата начала выносной линии или положение знака по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X = Object.X                        Получить свойство(* )
Object.X = X                        Установить свойство (* )
X = Object.GetX()                   Получить свойство (**)
Object.SetX( X )                    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X( &X )                  Получить свойство
Object.put_X(  X )                  Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату X положения знака неуказанной шероховатости.

Y – Координата начала выносной линии или положение знака по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату Y положения знака неуказанной шероховатости.

ISpecRough – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить неуказанную шероховатость.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет применить заданные параметры неуказанной шероховатости.

Интерфейс IStamp

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Интерфейс основной надписи (штампа).

Иерархия:

IDispatch

IKompasAPIObject

IStamp

Описание:

Интерфейс позволяет задавать и получать параметры штампа.

Примечание:

Интерфейс можно получить у интерфейса параметров листа оформления с помощью свойства ILayoutSheet::Stamp.

IStamp – свойства

Crossed – Признак необходимости перестроения штампа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Crossed = Object.Crossed	Получить свойство(*)
Crossed = Object.GetCrossed()	Получить свойство (**)

Синтаксис COM:

Object.get_Crossed(&Crossed)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

Text = Object.Text(Id)	Получить свойство(*)
Text = Object.GetText(Id)	Получить свойство (**)

Синтаксис COM:

Object.get_Text(Id, &Text)	Получить свойство

Входные параметры:

Id (long)	- идентификатор ячейки.

Примечание:

1. Свойство позволяет получать интерфейс текста ячейки штампа.
2. Свойство доступно только для чтения.

IStamp – методы

Clear – Очистить штамп

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет очистить штамп.

GetNextColumnId – Получить идентификатор следующей непустой ячейки штампа

Интерфейс...

Синтаксис Automation:

long GetNextColumnId(long Id);

Синтаксис COM:

HRESULT GetNextColumnId(long Id,
long * Result);

Входные параметры:

Id - идентификатор ячейки штампа.

Возвращаемое значение:

- Идентификатор следующей не пустой ячейки штампа.

Примечание:

Метод позволяет получить идентификатор следующей непустой ячейки штампа.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет применить заданные параметры штампа.

Интерфейс ITechnicalDemand

Справка системы КОМПАС...

`KOMPAS.chm::/413_Glava47_Tekhnicheskie_trebo.htm`

Интерфейс технических требований.

Иерархия:

IDispatch
    IKompasAPIObject
        ITechnicalDemand

Описание:

Интерфейс позволяет задавать и получать параметры технических требований.

Примечание:

Интерфейс можно получить у интерфейса чертежа с помощью свойства IDrawingDocument::TechnicalDemand.

ITechnicalDemand – свойства

AutoPlacement – Авторазмещение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoPlacement = Object.AutoPlacement	=	Получить свойство(*)
Object.AutoPlacement = AutoPlacement	=	Установить свойство (*)
AutoPlacement = Object.GetAutoPlacement()	=	Получить свойство (**)
Object.SetAutoPlacement(AutoPlacement)		Установить свойство (**)

Синтаксис COM:

Object.get_AutoPlacement(&AutoPlacement)	Получить свойство
Object.put_AutoPlacement(AutoPlacementX1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак авторазмещения технических требований.

BlocksGabarits – Габариты

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

BlocksGabarits = Object.BlocksGabarits	=	Получить свойство(*)
Object.BlocksGabarits = BlocksGabarits	=	Установить свойство (*)
BlocksGabarits = Object.GetBlocksGabarits()	=	Получить свойство (**)
Object.SetBlocksGabarits(BlocksGabarits)		Установить свойство (**)

Синтаксис COM:

Object.get_BlocksGabarits(&BlocksGabarits)	Получить свойство

Object.put_BlocksGabarits(BlocksGabarits)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать габариты технических требований.

BlocksStartLineNumbers – Получить список номеров строк в начале блоков

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_I4

Синтаксис Automation:

BlocksStartLineNumbers = Object.BlocksStartLineNumbers	Получить свойство(*)
BlocksStartLineNumbers = Object.GetBlocksStartLineNumbers()	Получить свойство (**)

Синтаксис COM:

Object.get_BlocksStartLineNumbers(&BlocksStartLineNumbers)	Получить свойство

Примечание:

1. Свойство позволяет получать список номеров строк в начале блоков технических требований.
2. Свойство доступно только для чтения.

IsCreated – Признак отображения в документе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCreated = Object.IsCreated		Получить свойство(*)
IsCreated Object.GetIsCreated()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsCreated(&IsCreated)	Получить свойство

Примечание:

1. Свойство позволяет получать признак отображения технических требований в документе.
2. Свойство доступно только для чтения.

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

Text = Object.Text	Получить свойство(*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс текста технических требований.
2. Свойство доступно только для чтения.

ITechnicalDemand – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить технические требования.

Synchronize – Синхронизация

Интерфейс...

Синтаксис Automation:

BOOL Synchronize();

Синтаксис COM:

HRESULT Synchronize(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Метод позволяет при создании ассоциативного вида синхронизировать технические требования чертежа с техническими требованиями документа 3D, с которого создается вид.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет применить заданные параметры технических требований.

Макрообъекты

Интерфейс IMacroObjects

Справка системы КОМПАС...

`KOMPAS.chm::/351_Glava39_Ispolqzovanie_makro.htm`

Интерфейс коллекции макроэлементов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IMacroObjects

IAnnotativeContainer

Описание:

Позволяет создавать и получать макроэлементы.

Примечание:

1. Получить интерфейс можно, используя метод контейнера графических объектов IDrawingContainer::MacroObjects.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить дополнительный интерфейс IAnnotativeContainer.

IMacroObjects – свойства

MacroObject – Макроэлемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс макроэлемента IMacroObject

Синтаксис Automation:

MacroObject = iObject.MacroObject (Index);	Получить свойство(*)
MacroObject = iObject.GetMacroObject(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_MacroObject(Index, &MacroObject)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IMacroObjects – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IMacroObject ** Result);

Возвращаемое значение:

- Указатель на интерфейс макроэлемента IMacroObject.

Интерфейс IMacroObject

Справка системы КОМПАС...

KOMPAS.chm::/351_Glava39_Ispolqzovanie_makro.htm

Интерфейс нетипизированного макроэлемента.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IMacroObject

IAnnotativeContainer

IDrawingContainer

IBuildingContainer

ISymbols2DContainer

IPropertyKeeper

Описание:

Интерфейс позволяет задавать параметры макроэлемента.

Примечание:

1. Интерфейс можно получить, используя свойство коллекции макроэлементов IMacroObjects::MacroObject или метод IMacroObjects::Add.
2. После задания параметров макроэлемента требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif) у данного интерфейса можно получить дополнительные интерфейсы:

- интерфейс контейнера аннотационных объектов IAnnotativeContainer,
- интерфейс контейнера объектов вида графического документа IDrawingContainer,
- интерфейс контейнера объектов СПДС IBuildingContainer,
- интерфейс контейнера условных обозначений ISymbols2DContainer,
- интерфейс получения/редактирования значения свойств IPropertyKeeper.

IMacroObject – свойства

AttachedLeaders – Присоединенные линии выноски

Интерфейс...

Тип данных: Указатель на интерфейс ILeaders

Синтаксис Automation:

AttachedLeaders = Object.AttachedLeaders	Получить свойство(*)
AttachedLeaders = Object.GetAttachedLeaders()	Получить свойство (**)

Синтаксис COM:

Object.get_AttachedLeaders(&AttachedLeaders)	Получить свойство

Свойство позволяет получить коллекцию присоединенных линий-выносок макроэлемента.

Примечание:

Свойство доступно только для чтения.

BreakObjectsEnabled – Учитывать разрыв вида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BreakObjectsEnabled = Object.BreakObjectsEnabled	Получить свойство(*)
Object.BreakObjectsEnabled = BreakObjectsEnabled	Установить свойство (*)
BreakObjectsEnabled = Object.GetBreakObjectsEnabled( )	Получить свойство (**)
Object.SetBreakObjectsEnabled(BreakObjectsEnabled)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakObjectsEnabled(&BreakObjectsEnabled)	Получить свойство
Object.put_BreakObjectsEnabled(BreakObjectsEnabled)	Установить свойство

Версия Компас v18.1

Command – Номер команды редактирования

Интерфейс...

Тип данных: long

Синтаксис Automation:

Command = Object.Command	Получить свойство(*)
Object.Command = Command	Установить свойство (*)
Command = Object.GetCommand()	Получить свойство (**)
Object.SetCommand(Command)	Установить свойство (**)

Синтаксис COM:

Object.get_Command(&Command)	Получить свойство
Object.put_Command(Command)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать номер команды редактирования в меню библиотеки.

CreateSpcObjects – Создавать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateSpcObjects = Object.CreateSpcObjects	Получить свойство(*)
Object.CreateSpcObjects = CreateSpcObjects	Установить свойство (*)
CreateSpcObjects = Object.GetCreateSpcObjects()	Получить свойство (**)
Object.SetCreateSpcObjects(CreateSpcObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateSpcObjects(&CreateSpcObjects)	Получить свойство
Object.put_CreateSpcObjects(CreateSpcObjects)	Установить свойство

DoubleClickEditable – Редактирование по двойному клику поддерживается

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DoubleClickEditable = Object.DoubleClickEditable	Получить свойство(*)
Object.DoubleClickEditable = DoubleClickEditable	Установить свойство (*)
DoubleClickEditable = Object.GetDoubleClickEditable()	Получить свойство (**)
Object.SetDoubleClickEditable(DoubleClickEditable)	Установить свойство (**)

Синтаксис COM:

Object.get_DoubleClickEditable(&DoubleClickEditable)	Получить свойство
Object.put_DoubleClickEditable(DoubleClickEditable)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак редактирования по двойному щелчку мыши.

HotPointsEditable – Редактирование при помощи характерных точек поддерживается

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HotPointsEditable = Object.HotPointsEditable	Получить свойство(*)
Object.HotPointsEditable = HotPointsEditable	Установить свойство (*)
HotPointsEditable = Object.GetHotPointsEditable()	Получить свойство (**)
Object.SetHotPointsEditable(HotPointsEditable)	Установить свойство (**)

Синтаксис COM:

Object.get_HotPointsEditable(&HotPointsEditable)	Получить свойство
Object.put_HotPointsEditable(HotPointsEditable)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак редактирования при помощи характерных точек.

ExternalEditable – Поддерживается интерфейс внешнего управления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ExternalEditable = Object.ExternalEditable	Получить свойство(*)
Object.ExternalEditable = ExternalEditable	Установить свойство (*)
ExternalEditable = Object.GetExternalEditable()	Получить свойство (**)
Object.SetExternalEditable(ExternalEditable)	Установить свойство (**)

Синтаксис COM:

Object.get_ExternalEditable(&ExternalEditable)	Получить свойство

Object.put_ExternalEditable(ExternalEditable)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак поддержки интерфейса внешнего управления.

IName – Имя макроэлемента в Дереве чертежа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство(*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

LibraryFileName – Имя файла библиотеки, в которой находится функция редактирования

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LibraryFileName = Object.LibraryFileName	Получить свойство(*)
Object.LibraryFileName = LibraryFileName	Установить свойство (*)
LibraryFileName = Object.GetLibraryFileName()	Получить свойство (**)
Object.SetLibraryFileName(LibraryFileName)	Установить свойство (**)

Синтаксис COM:

Object.get_LibraryFileName(&LibraryFileName)	Получить свойство
Object.put_LibraryFileName(LibraryFileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя файла библиотеки, в которой находится функция редактирования макроэлемента.

LibraryName – Имя библиотеки, в которой находится функция редактирования

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LibraryName = Object.LibraryName	Получить свойство(*)
Object.LibraryName = LibraryName	Установить свойство (*)
LibraryName = Object.GetLibraryName()	Получить свойство (**)
Object.SetLibraryName(LibraryName)	Установить свойство (**)

Синтаксис COM:

Object.get_LibraryFileName(&LibraryFileName)	Получить свойство
Object.put_LibraryFileName(LibraryFileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя библиотеки, в которой находится функция редактирования макроэлемента (отображаемое имя библиотеки).

PropertyObjectEditable – Поддерживается интерфейс внешних свойств объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PropertyObjectEditable = Object.PropertyObjectEditable	Получить свойство(*)
Object.PropertyObjectEditable = PropertyObjectEditable	Установить свойство (*)
PropertyObjectEditable = Object.GetPropertyObjectEditable()	Получить свойство (**)
Object.SetPropertyObjectEditable(PropertyObjectEditable)	Установить свойство (**)

Синтаксис COM:

Object.get_PropertyObjectEditable(&PropertyObjectEditable)	Получить свойство
Object.put_PropertyObjectEditable(PropertyObjectEditable)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак поддержки интерфейса внешних свойств объекта ILibPropertyObject.

UserParams – Массив SAFEARRAY байтов пользовательских параметров объекта

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
HotPointsEditable = Object.HotPointsEditable                    Получить свойство(* )
Object.HotPointsEditable = HotPointsEditable                    Установить свойство (* )
HotPointsEditable = Object.GetHotPointsEditable()               Получить свойство (**)
Object.SetHotPointsEditable( HotPointsEditable )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HotPointsEditable( &HotPointsEditable )              Получить свойство
Object.put_HotPointsEditable(  HotPointsEditable )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать массив SAFEARRAY со структурой пользовательских параметров объекта.

IMacroObject – методы

AddDefaultHotPoint – Задать характерную точку макроэлемента

Интерфейс...

Синтаксис Automation:

BOOL AddDefaultHotPoint(double X, double Y);

Синтаксис COM:

HRESULT AddDefaultHotPoint(double X, double Y, BOOL * Result);

Входные параметры:

X, Y	- координаты характерной точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

AddObjects – Добавить объект, слой, вид, группу или SAFEARRAY объектов в макрообъект

Интерфейс...

Синтаксис Automation:

BOOL AddObjects(VARIANT Objects);

Синтаксис COM:

HRESULT Add(VARIANT Objects, BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов, добавляемых в макроэлемент.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавлять объект, слой, вид, группу или SAFEARRAY объектов в макроэлемент

DeleteAttachedLeaders – Удалить присоединенные линии выноски

Интерфейс...

Синтаксис Automation:

BOOL DeleteAttachedLeaders();

Синтаксис COM:

HRESULT DeleteAttachedLeaders(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DeleteDefaultHotPoint – Удалить характерную точку макроэлемента

Интерфейс...

Синтаксис Automation:

BOOL DeleteDefaultHotPoint();

Синтаксис COM:

HRESULT DeleteDefaultHotPoint(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

FindObject – Найти объект, ближайший к заданной точке, удовлетворяющий параметрам поиска

Интерфейс...

Синтаксис Automation:

LPDISPATCH FindObject(double X, double Y, double Limit, IFindObjectParameters * Param);

Синтаксис COM:

HRESULT FindObject(double X, double Y, double Limit, IFindObjectParameters * Param, IDrawingObject * * Result);

Возвращаемое значение:

- Указатель на интерфейс IDrawingObject.

Входные параметры:

X	- первая координата точки,
Y	- вторая координата точки,
Limit	- предел поиска и найденное расстояние,
Param	- параметры поиска объектов IFindObjectParameters

Примечание:

Для поиска объектов в точке могут быть заданы расширенные параметры поиска.

Интерфейс расширенных параметров поиска IFindObjectParameters можно получить с помощью метода IKompasDocument1::GetInterface
в 2D документах при использовании константы ksObjectFindObjectParameters.

Если расширенные параметры поиска не требуются, необходимо передать NULL в функцию.

GetPlacement – Получить точку привязки и угол поворота системы координат макроэлемента

Интерфейс...

Синтаксис Automation:

BOOL GetPlacement(double * X,
double * Y,
double * Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

HRESULT GetPlacement(double * X,
double * Y,
double * Angle,
BOOL * MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота макроэлемента,
MirrorSymmetry	- признак зеркальной симметрии объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить точку привязки и угол поворота макроэлемента.

SetPlacement – Установить точку привязки и угол поворота системы координат макроэлемента

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement(double X,
double Y,
double Angle,
BOOL * MirrorSymmetry);

Синтаксис COM:

HRESULT SetPlacement(double X,
double Y,
double Angle,
BOOL * MirrorSymmetry,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки,
Angle	- угол поворота макроэлемента,
MirrorSymmetry	- признак зеркальной симметрии объекта (NULL – не изменять признак).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить точку привязки и угол поворота макроэлемента.

Смотрите также:

Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена относительно исходного ее построения

Интерфейс IAnnotativeContainer

Интерфейс контейнера аннотационных объектов.

Иерархия:

IDispatch

IAnnotativeContainer

Описание:

Интерфейс позволяет создавать и получать аннотационные объекты.

Примечание:

Интерфейс является дополнительным к интерфейсу макрообъекта IMacroObject. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IAnnotativeContainer – свойства

Count – Количество объектов

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Count = Object.Count          Получить свойство(* )
Count = Object.GetCount()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Count( &Count )          Получить свойство
```

Примечание:

1. Свойство позволяет получать количество объектов в контейнере.
2. Свойство доступно только для чтения.

Item – Получить элемент по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAnnotativeObject

Синтаксис Automation:

```
Item = Object.Item( Index )     Получить свойство(* )
```

Item = Object.GetItem(Index) — Получить свойство (**)

Синтаксис COM:

Object.get_Item(Index, &Item) — Получить свойство

Входные параметры:

Index - Индекс элемента.

Примечание:

1. Свойство позволяет получать аннотационные объекты.
2. Свойство доступно только для чтения.

IAnnotativeContainer – методы

Add – Добавить объект

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(long type);

Синтаксис COM:

HRESULT Add(DrawingObjectTypeEnum type, IAnnotativeObject ** Result);

Входные параметры:

type - тип добавляемого объекта из перечисления DrawingObjectTypeEnum.

Возвращаемое значение:

- указатель на интерфейс аннотационного объекта IAnnotativeObject.

Примечание:

Метод позволяет добавить объект в контейнер.

AddObjects – Добавить объекты в группу

Интерфейс...

Синтаксис Automation:

BOOL AddObjects(VARIANT Objects);

Синтаксис COM:

HRESULT AddObjects(VARIANT Objects, BOOL * Result);

Входные параметры:

Objects	- объект или массив объектов для добавления в группу (VARIANT типа VT_DISPATCH или VT_ARRAY | VT_DISPATCH).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет добавить объекты в группу.

CreateByGeomObject – Преобразовать геометрический объект в аннотационный

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateByGeomObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT CreateByGeomObject(IDrawingObject * obj,
IAnnotativeObject ** Result);

Входные параметры:

obj	- указатель на интерфейс геометрического объекта IDrawingObject.

Возвращаемое значение:

- указатель на интерфейс аннотационного объекта IAnnotativeObject.

Примечание:

Метод позволяет преобразовать геометрический объект в аннотационный и добавить его в контейнер.

Интерфейс IAnnotativeObject

Интерфейс аннотационных объектов.

Иерархия:

IDispatch

IAnnotativeObject

Описание:

Интерфейс позволяет задать параметры аннотационного объекта.

Примечание:

Интерфейс можно получить у контейнера аннотационных объектов с помощью свойства IAnnotativeContainer::Item и методов IAnnotativeContainer::Add и IAnnotativeContainer::CreateByGeomObject.

IAnnotativeObject – свойства

Sign1 – Тип значка

Интерфейс...

Тип данных: из перечисления ksAnnotativeTerminatorSignEnum

Синтаксис Automation:

```
Sign1 = Object.Sign1                    Получить свойство(* )
Object.Sign1 = Sign1                    Установить свойство (* )
Sign1 = Object.GetSign1()               Получить свойство (**)
Object.SetSign1( Sign1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sign1( &Sign1 )              Получить свойство
Object.put_Sign1(  Sign1 )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать тип значка 1.

Sign2 – Тип значка

Интерфейс...

Тип данных: из перечисления ksAnnotativeTerminatorSignEnum.

Синтаксис Automation:

```
Sign2 = Object.Sign2                    Получить свойство(* )
Object.Sign2 = Sign2                    Установить свойство (* )
Sign2 = Object.GetSign2()               Получить свойство (**)
Object.SetSign2( Sign2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sign2( &Sign2 )              Получить свойство
Object.put_Sign2(  Sign2 )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать тип значка 2.

Обозначения и размеры

Контейнер

Интерфейс ISymbols2DContainer

Интерфейс контейнера условных обозначений.

Иерархия:

IDispatch

ISymbols2DContainer

Описание:

Позволяет получить коллекции 2D размеров и обозначений.

Примечание:

Дополнительный интерфейс вида. Данный интерфейс можно получить у интерфейса вида IView посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ISymbols2DContainer – свойства

AngleDimensions – Угловые размеры

Интерфейс...

Тип данных: IAngleDimensions.

Синтаксис Automation:

AngleDimensions = Object.AngleDimensions	Получить свойство(*)
AngleDimensions = Object.GetAngleDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_AngleDimensions(&AngleDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать угловые и угловые с обрывом размеры.
2. Свойство доступно только для чтения.

ArcDimensions– Размеры дуг окружностей

Интерфейс...

Тип данных: IArcDimensions

Синтаксис Automation:

ArcDimensions = Object.ArcDimensions	Получить свойство(*)
ArcDimensions = Object.GetArcDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_ArcDimensions(&ArcDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать размеры дуг окружностей.
2. Свойство доступно только для чтения.

AssociationTables – Ассоциативные таблицы отчетов

Интерфейс...

Тип данных: IAssociationTables

Синтаксис Automation:

AssociationTables = Object.AssociationTables	Получить свойство(*)
AssociationTables = Object.GetAssociationTables()	Получить свойство (**)

Синтаксис COM:

Object.get_AssociationTables(&AssociationTables)	Получить свойство

Примечание:

Свойство доступно только для чтения.

AxisLines – Коллекция осевых линий

Интерфейс...

Тип данных: указатель на интерфейс IAxisLines

Синтаксис Automation:

AxisLines = Object.AxisLines	Получить свойство(*)
AxisLines = Object.GetAxisLines()	Получить свойство (**)

Синтаксис COM:

Object.get_AxisLines(&AxisLines)	Получить свойство

Свойство позволяет получать интерфейс коллекции осевых линий.

Примечание:

Свойство доступно только для чтения.

Bases – Обозначения базы

Интерфейс...

Тип данных: IBases

Синтаксис Automation:

Bases = Object.Bases	Получить свойство(*)
Bases = Object.GetBases()	Получить свойство (**)

Синтаксис COM:

Object.get_Bases(&Bases)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать обозначения базы.
2. Свойство доступно только для чтения.

BreakLineDimensions – Линейные размеры с обрывом

Интерфейс...

Тип данных: IBreakLineDimensions

Синтаксис Automation:

BreakLineDimensions = Object.BreakLineDimensions	Получить свойство(*)
BreakLineDimensions = Object.GetBreakLineDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_BreakLineDimensions(&BreakLineDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать линейные размеры с обрывом.
2. Свойство доступно только для чтения.

BreakRadialDimensions – Радиальные размеры с изломом

Интерфейс...

Тип данных: IBreakRadialDimensions

Синтаксис Automation:

BreakRadialDimensions = Object.BreakRadialDimensions	Получить свойство(*)
BreakRadialDimensions = Object.GetBreakRadialDimensions ()	Получить свойство (**)

Синтаксис COM:

Object.get_BreakRadialDimensions(&BreakRadialDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать радиальные размеры с изломом.
2. Свойство доступно только для чтения.

BrokenLines – Коллекция линий обрыва с изломами

Интерфейс...

Тип данных: указатель на интерфейс IBrokenLines

Синтаксис Automation:

BrokenLines = Object.BrokenLiness	Получить свойство(*)
BrokenLines = Object.GetBrokenLines()	Получить свойство (**)

Синтаксис COM:

Object.get_BrokenLines(&BrokenLines)	Получить свойство

Свойство позволяет получать интерфейс коллекции линий обрыва с изломами.

Примечание:

Свойство доступно только для чтения.

CentreMarkers – Коллекция обозначений центра

Интерфейс...

Тип данных: указатель на интерфейс ICentreMarkers

Синтаксис Automation:

CentreMarkers = Object.CentreMarkers	Получить свойство(*)
CentreMarkers = Object.GetCentreMarkers()	Получить свойство (**)

Синтаксис COM:

Object.get_CentreMarkers(&CentreMarkers)	Получить свойство

Свойство позволяет получать интерфейс коллекции обозначений центра.

Примечание:

Свойство доступно только для чтения.

CircularsCentries – Коллекция круговых сеток центров

Интерфейс...

Тип данных: Указатель на интерфейс ICircularsCentries.

Синтаксис Automation:

CircularsCentries = Object.CircularsCentries	Получить свойство(*)
CircularsCentries = Object.GetCircularsCentries()	Получить свойство (**)

Синтаксис COM:

Object.get_CircularsCentries(&CircularsCentries)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

ConditionIntersects – Коллекция условных пересечений

Интерфейс...

Тип данных: Указатель на интерфейс IConditionIntersects

Синтаксис Automation:

ConditionIntersects = Object.ConditionIntersects	Получить свойство(*)
ConditionIntersects = Object.GetConditionIntersects()	Получить свойство (**)

Синтаксис COM:

Object.get_ConditionIntersects(&ConditionIntersects)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CutLines – Линии разреза/сечения

Интерфейс...

Тип данных: ICutLines

Синтаксис Automation:

CutLines = Object.CutLines	Получить свойство(*)
CutLines = Object.GetCutLines()	Получить свойство (**)

Синтаксис COM:

Object.get_CutLines(&CutLines)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать линии разреза/сечения.
2. Свойство доступно только для чтения.

DiametralDimensions – Диаметральные размеры

Интерфейс...

Тип данных: IDiametralDimensions

Синтаксис Automation:

DiametralDimensions = Object.DiametralDimensions	Получить свойство(*)
DiametralDimensions = Object.GetDiametralDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_DiametralDimensions(&DiametralDimensions) — Получить свойство

Примечание:

1. Свойство позволяет получать и создавать диаметральные размеры.
2. Свойство доступно только для чтения.

DrawingTables – Таблицы

Интерфейс...

Тип данных: IDrawingTables

Синтаксис Automation:

DrawingTables = Object.DrawingTables — Получить свойство(*)
DrawingTables = Object.GetDrawingTables() — Получить свойство (**)

Синтаксис COM:

Object.get_DrawingTables(&DrawingTables) — Получить свойство

Примечание:

1. Свойство позволяет получать и создавать таблицы.
2. Свойство доступно только для чтения.

HeightDimensions – Размеры высоты

Интерфейс...

Тип данных: IHeightDimensions

Синтаксис Automation:

HeightDimensions = Object.HeightDimensions — Получить свойство(*)
HeightDimensions = Object.GetHeightDimensions() — Получить свойство (**)

Синтаксис COM:

Object.get_HeightDimensions(&HeightDimensions) — Получить свойство

Примечание:

1. Свойство позволяет получать и создавать размеры высоты.
2. Свойство доступно только для чтения.

Leaders – Линия-выноска

Интерфейс...

Тип данных: ILeaders

Синтаксис Automation:

Leaders = Object.Leaders	Получить свойство(*)
Object.get_Leaders(&Leaders)	Получить свойство (**)

Синтаксис COM:

Object.get_Leaders(&Leaders)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать линии-выноски.
2. Свойство доступно только для чтения.

LineDimensions – Линейные размеры

Интерфейс...

Тип данных: ILineDimensions

Синтаксис Automation:

LineDimensions = Object.LineDimensions	Получить свойство(*)
LineDimensions = Object.GetLineDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_LineDimensions(&LineDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать линейные размеры.
2. Свойство доступно только для чтения.

LinearsCentries – Коллекция линейных сеток центров

Интерфейс...

Тип данных: Указатель на интерфейс ILinearsCentries.

Синтаксис Automation:

LinearsCentries = Object.LinearsCentries	Получить свойство(*)
LinearsCentries = Object.GetLinearsCentries()	Получить свойство (**)

Синтаксис COM:

Object.get_LinearsCentries(&LinearsCentries)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

RadialDimensions – Радиальные размеры

Интерфейс...

Тип данных: IRadialDimensions

Синтаксис Automation:

RadialDimensions = Object.RadialDimensions	Получить свойство(*)
RadialDimensions = Object.GetRadialDimensions()	Получить свойство (**)

Синтаксис COM:

Object.get_RadialDimensions(&RadialDimensions)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать радиальные размеры.
2. Свойство доступно только для чтения.

RemoteElements – Коллекция выносных элементов

Интерфейс...

Тип данных: указатель на интерфейс IRemoteElements

Синтаксис Automation:

RemoteElements = Object.RemoteElements	Получить свойство(*)
RemoteElements = Object.GetRemoteElements()	Получить свойство (**)

Синтаксис COM:

Object.get_RemoteElements(&RemoteElements)	Получить свойство

Свойство позволяет получать интерфейс коллекции выносных элементов.

Примечание:

Свойство доступно только для чтения.

Roughs – Обозначения шероховатости

Интерфейс...

Тип данных: IRoughs

Синтаксис Automation:

Roughs = Object.Roughs	Получить свойство(*)
Roughs = Object.GetRoughs()	Получить свойство (**)

Синтаксис COM:

Object.get_Roughs(&Roughs)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать обозначения шероховатости.
2. Свойство доступно только для чтения.

Tolerances – Допуск формы

Интерфейс...

Тип данных: ITolerances

Синтаксис Automation:

Tolerances = Object.Tolerances	Получить свойство(*)
Tolerances = Object.GetTolerances()	Получить свойство (**)

Синтаксис COM:

Object.get_Tolerances(&Tolerances)	Получить свойство

Примечание:

1. Свойство позволяет получать и создавать допуски формы.
2. Свойство доступно только для чтения.

ViewPointers – Стрелки взгляда

Интерфейс...

Тип данных: IViewPointers

Синтаксис Automation:

```
ViewPointers =                    Получить свойство(* )
Object.ViewPointers
ViewPointers =                    Получить свойство (**)
Object.GetViewPointers()
```

Синтаксис COM:

```
Object.get_ViewPointers(          Получить свойство
&ViewPointers )
```

Примечание:

1. Свойство позволяет получать и создавать линии стрелки взгляда.
2. Свойство доступно только для чтения.

WaveLines – Коллекция волнистых линий

Интерфейс...

Тип данных: указатель на интерфейс IWaveLines

Синтаксис Automation:

```
WaveLines = Object.WaveLines      Получить свойство(* )
WaveLines =                       Получить свойство (**)
Object.GetWaveLines()
```

Синтаксис COM:

```
Object.get_WaveLines(             Получить свойство
&WaveLines )
```

Примечание:

1. Свойство позволяет получать интерфейс коллекции волнистых линий.
2. Свойство доступно только для чтения.

Элементы обозначений

Интерфейс IBranchs

Справка системы КОМПАС...

kompas.chm::/1024_113_2_2_Dobavlenie_i_udalenie_otv.htm

Интерфейс для работы с ответвлениями.

Иерархия:

IDispatch

IBranchs

Примечание:

Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейсов IBaseLeader, ITolerance, IMarkLeader, IPositionLeader, IChangeLeader, IBrandLeader, IRough.

IBranchs – свойства

BranchCount – Число ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchCount = Object.BranchCount	Получить свойство(*)
BranchCount = Object.GetBranchCount()	Получить свойство (**)

Синтаксис COM:

Object.get_BranchCount(&BranchCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BranchPoints – Массив SAFEARRAY координат точек ответвления

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY I VT_R8

Синтаксис Automation:

BranchPoints = Object.BranchPoints(Index)	Получить свойство(*)

Object.BranchPoints(Index) = BranchPoints	Установить свойство (*)
BranchPoints = Object.GetBranchPoints(Index)	Получить свойство (**)
Object.SetBranchPoints(Index, BranchPoints)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchPoints(Index, &BranchPoints)	Получить свойство
Object.put_BranchPoints(Index, BranchPoints)	Установить свойство

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, x1, y1, ...xi, yi.
2. Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

BranchPointsCount – Число ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchPointsCount = Object.BranchPointsCount()	Получить свойство(*)
BranchPointsCount = Object.GetBranchPointsCount ()	Получить свойство (**)

Синтаксис COM:

Object.get_BranchPointsCount(&BranchPointsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BranchX0 – Координата конечной точки ответвления по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchX0 = Object.BranchX0	Получить свойство(*)
Object.BranchX0 = BranchX0	Установить свойство (*)
BranchX0 = Object.GetBranchX0()	Получить свойство (**)
Object.SetBranchX0(BranchX0)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchX0(&BranchX0)	Получить свойство
Object.put_BranchX0(BranchX0)	Установить свойство

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

BranchY0 – Координата конечной точки ответвления по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchY0 = Object.BranchY0	Получить свойство(*)
Object.BranchY0 = BranchY0	Установить свойство (*)
BranchY0 = Object.GetBranchY0()	Получить свойство (**)
Object.SetBranchY0(BranchY 0)	Установить свойство (**)

Синтаксис COM:

Object.get_Branch Y0(&BranchY 0)	Получить свойство
Object.put_Branch Y0(BranchY0)	Установить свойство

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

X0 – Координата начала полки или точка привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X0 = Object.X0                  Получить свойство(* )
Object.X0 = X0                  Установить свойство (* )
X0 = Object.GetX0()             Получить свойство (**)
Object.SetX0( X0 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X0( &X0 )            Получить свойство
Object.put_X0(  X0 )            Установить свойство
```

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Y0 – Координата начала полки или точка привязки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y0 = Object.Y0                  Получить свойство(* )
Object.Y0 = Y0                  Установить свойство (* )
Y0 = Object.GetY0()             Получить свойство (**)
Object.SetY0( Y0 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y0( &Y0 )            Получить свойство
Object.put_Y0(  Y0 )            Установить свойство
```

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

IBranchs – методы

AddBranch – Добавить ответвление

Интерфейс...

Синтаксис Automation:

```
BOOL AddBranch( long Index,
VARIANT Points );
```

Синтаксис COM:

HRESULT AddBranch(long Index,
VARIANT Points,
BOOL * Result);

Входные параметры:

Index	- индекс, с которым добавляется ответвление,
Points	- массив SafeArray координат точек.

Возвращаемое значение:

TRUE	- ответвление добавлено.

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, x1, y1, ...xi, yi.
2. Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

AddBranchByPoint – Добавить прямолинейное ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranchByPoint(long Index,
double X,
double Y);

Синтаксис COM:

HRESULT AddBranchByPoint(long Index,
double X,
double Y,
BOOL * Result);

Входные параметры:

Index	- индекс, с которым добавляется ответвление,
x, y	- координаты точки целеуказания.

Возвращаемое значение:

TRUE	- ответвление добавлено.

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

DeleteBranch – Удалить ответвление

Интерфейс...

Синтаксис Automation:

BOOL DeleteBranch(long Index);

Синтаксис COM:

HRESULT DeleteBranch(long Index,
BOOL * Result);

Входные параметры:

Index	- индекс, с которым добавляется ответвление.

Возвращаемое значение:

TRUE	- ответвление удалено.

Интерфейс IBrandLeader

Справка системы КОМПАС: Обозначение клеймения 2D

`kompas.chm::/CM_BRANDLEADER.htm`

Обозначение клеймения 3D

`kompas.chm::/CM_BRANDLEADER_3D.htm`

Интерфейс параметров обозначения клеймения.

Иерархия:

IDispatch

IBrandLeader

IBranchs

IBranchs3D

Описание:

1. Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейсов IBaseLeader или IBaseLeader3D.
2. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс ответвлений IBranchs или IBranchs3D.

IBrandLeader – свойства

Designation – Обозначение

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Designation = Object.Designation	Получить свойство(*)
Designation = Object.GetDesignation()	Получить свойство (**)

Синтаксис COM:

Object.get_Designation(&Designation)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста обозначения.
2. Свойство доступно только для чтения.

Direction – Направление

Интерфейс...

Тип данных: BOOL

Значения свойства:

FALSE	- знак влево,
TRUE	- знак вправо.

Синтаксис Automation:

Direction = Object.Direction	Получить свойство(*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection( )	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство(*)
Object.put_Direction(Direction)	Установить свойство (*)

Примечание:

Свойство позволяет устанавливать и получать направление знака.

TextOnBranch – Текст над ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextOnBranch = Object.TextOnBranch	Получить свойство(*)
TextOnBranch = Object.GetTextOnBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextOnBranch(&TextOnBranch)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста над ножкой.
2. Свойство доступно только для чтения.

TextUnderBranch – Текст под ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUnderBranch = Object.TextUnderBranch	Получить свойство(*)
TextUnderBranch = Object.GetTextUnderBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextUnderBranch(&TextUnderBranch)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста под ножкой.
2. Свойство доступно только для чтения.

Интерфейс IChangeLeader

Справка системы КОМПАС...

`KOMPAS.chm::/CM_CHANGELEADER.htm`

Справка системы КОМПАС: команда Знак изменения

`kompas.chm::/CM_CHANGELEADER.htm`

Интерфейс параметров знака изменения.

Иерархия:

IDispatch

IChangeLeader

IBranchs

IBranchs3D

Примечание:

1. Дополнительный интерфейс для IBaseLeader.
2. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс ответвлений IBranchs или IBranchs3D.

IChangeLeader – свойства

Designation – Обозначение

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Designation = Object.Designation	Получить свойство(*)
Designation = Object.GetDesignation()	Получить свойство (**)

Синтаксис COM:

Object.get_Designation(&Designation)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста обозначения.
2. Свойство доступно только для чтения.

FullLeaderLength – На всю длину

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- выноска полной длины,
FALSE	- ограниченный отрезок.

Синтаксис Automation:

FullLeaderLength = Object.FullLeaderLength	Получить свойство(*)
Object.FullLeaderLength = FullLeaderLength	Установить свойство (*)
FullLeaderLength = Object.GetFullLeaderLength( )	Получить свойство (**)
Object.SetFullLeaderLength(FullLeaderLength)	Установить свойство (**)

Синтаксис COM:

Object.get_FullLeaderLength(&FullLeaderLength)	Получить свойство
Object.put_FullLeaderLength(FullLeaderLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип выноски: выноска полной длины или ограниченный отрезок.

LeaderLength – Длина выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeaderLength = Object.LeaderLength	Получить свойство(*)
Object.LeaderLength = LeaderLength	Установить свойство (*)
LeaderLength = Object.GetLeaderLength( )	Получить свойство (**)
Object.SetLeaderLength(LeaderLength)	Установить свойство (**)

Синтаксис COM:

Object.get_LeaderLength(&LeaderLength)	Получить свойство
Object.put_LeaderLength(LeaderLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину выноски.

SignHeight – Высота знака

Интерфейс...

Тип данных: double

Синтаксис Automation:

SignHeight = Object.SignHeight	Получить свойство(*)
Object.SignHeight = SignHeight	Установить свойство (*)
SignHeight = Object.GetSignHeight( )	Получить свойство (**)
Object.SetSignHeight(SignHeight)	Установить свойство (**)

Синтаксис COM:

Object.get_SignHeight(&SignHeight)	Получить свойство
Object.put_SignHeight(SignHeight)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать высоту знака.

SignType – Тип значка

Интерфейс...

Тип данных: из перечисления ksChangeLeaderSignEnum

Синтаксис Automation:

SignType = Object.SignType	Получить свойство(*)
Object.SignType = SignType	Установить свойство (*)
SignType = Object.GetSignType( )	Получить свойство (**)
Object.SetSignType(SignType)	Установить свойство (**)

Синтаксис COM:

Object.get_SignType(&SignType)	Получить свойство
Object.put_SignType(SignType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип значка.

Интерфейс ILeader

Справка системы КОМПАС...

KOMPAS.chm::/225_28_4_2_Nastrojka_otrisovki_.htm

Интерфейс параметров линии-выноски.

Иерархия:

IDispatch

ILeader

Примечание:

1. Дополнительный интерфейс для IBaseLeader и IBaseLeader3D.
2. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ILeader – свойства

Arround – Признак обработки по контуру

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Arround = Object.Arround	Получить свойство(*)
Object.Arround = Arround	Установить свойство (*)
Arround = Object.GetArround()	Получить свойство (**)
Object.SetArround(Arround)	Установить свойство (**)

Синтаксис COM:

Object.get_Arround(&Arround)	Получить свойство
Object.put_Arround(Arround)	Установить свойство

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

BranchBegin – Начало ответвления

Интерфейс...

Тип данных: BOOL

Значение свойства:

TRUE	- от начала полки,
FALSE	- от конца полки.

Синтаксис Automation:

BranchBegin = Object.BranchBegin(Index)	Получить свойство(*)
Object.BranchBegin(Index) = BranchBegin	Установить свойство (*)
BranchBegin = Object.GetBranchBegin(Index)	Получить свойство (**)
Object.SetBranchBegin(Index, BranchBegin)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchBegin(&BranchBegin)	Index,	Получить свойство
Object.put_BranchBegin(BranchBegin)	Index,	Установить свойство

Входные параметры:

Index	- индекс ответвления.

ParallelBranch – Параллельные ответвления

Интерфейс...

Тип данных: BOOL

Значение свойства:

TRUE	- от начала полки,
FALSE	- от конца полки.

Синтаксис Automation:

ParallelBranch = Object.ParallelBranch	Получить свойство(*)
Object.ParallelBranch = ParallelBranch	Установить свойство (*)
ParallelBranch = Object.GetParallelBranch()	Получить свойство (**)
Object.SetParallelBranch(ParallelBranch)	Установить свойство (**)

Синтаксис COM:

Object.get_ParallelBranch(&ParallelBranch)	Получить свойство
Object.put_ParallelBranch(ParallelBranch)	Установить свойство

Входные параметры:

Index	- индекс ответвления.

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = Object.ShelfDirection	Получить свойство(*)
Object.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = Object.GetShelfDirection()	Получить свойство (**)
Object.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfDirection(&ShelfDirection)	Получить свойство
Object.put_ShelfDirection(ShelfDirection)	Установить свойство

SignType – Тип значка

Интерфейс...

Тип данных: из перечисления ksLeaderSignEnum

Синтаксис Automation:

SignType = Object.SignType	Получить свойство(*)
Object.SignType = SignType	Установить свойство (*)
SignType = Object.GetSignType()	Получить свойство (**)
Object.SetSignType(SignType)	Установить свойство (**)

Синтаксис COM:

Object.get_SignType(&SignType)	Получить свойство
Object.put_SignType(SignType)	Установить свойство

Примечание:

Свойство позволяет выбрать значок для обозначения соединения.

TextAfterShelf – Текст за полкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextAfterShelf = Object.TextAfterShelf	Получить свойство(*)
TextAfterShelf = Object.GetTextAfterShelf()	Получить свойство (**)

Синтаксис COM:

Object.get_TextAfterShelf(&TextAfterShelf)	Получить свойство

Входные параметры:

Index	- индекс ответвления.

Примечание:

Свойство доступно только для чтения.

TextOnBranch – Текст над ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextOnBranch = Object.TextOnBranch	Получить свойство(*)
TextOnBranch = Object.GetTextOnBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextOnBranch(&TextOnBranch)	Получить свойство

Входные параметры:

Index	- индекс ответвления.

Примечание:

Свойство доступно только для чтения.

TextOnShelf – Текст над полкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextOnShelf = Object.TextOnShelf	Получить свойство(*)
TextOnShelf = Object.GetTextOnShelf()	Получить свойство (**)

Синтаксис COM:

Object.get_TextOnShelf(&TextOnShelf)	Получить свойство

Входные параметры:

Index	- индекс ответвления.

Примечание:

Свойство доступно только для чтения.

TextUnderBranch – Текст под ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUnderBranch = Object.TextUnderBranch	Получить свойство(*)
TextUnderBranch = Object.GetTextUnderBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextUnderBranch(&TextUnderBranch)	Получить свойство

Входные параметры:

Index	- индекс ответвления.

Примечание:

Свойство доступно только для чтения.

TextUnderShelf – Текст под полкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUnderShelf = Object.TextUnderShelf	Получить свойство(*)
TextUnderShelf = Object.GetTextUnderShelf()	Получить свойство (**)

Синтаксис COM:

Object.get_TextUnderShelf(&TextUnderShelf)	Получить свойство

Входные параметры:

Index	- индекс ответвления.

Примечание:

Свойство доступно только для чтения.

Интерфейс IPositionLeader

Справка системы КОМПАС: Команда Обозначение позиции (документ-модель)

`kompas.chm::/CM_POSITIONLEADER.htm`

Основные сведения...

`kompas.chm::/233_28_7_2_Nastrojka_otrisovki_.htm#otrisovka_obozn_poziciy`

Интерфейс параметров линии-выноски для обозначения позиции.

Иерархия:

IDispatch

IPositionLeader

IBranchs

IBranchs3D

Примечание:

1. Дополнительный интерфейс для IBaseLeader и IBaseLeader3D.
2. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс ответвлений IBranchs или IBranchs3D.

IPositionLeader – свойства

Form – Тип формы

Интерфейс...

Тип данных: из перечисления ksPositionLederFormEnum

Синтаксис Automation:

Form = Object.Form	Получить свойство(*)
Object.Form = Form	Установить свойство (*)

Form = Object.GetForm( )	Получить свойство (**)
Object.SetForm(Form)	Установить свойство (**)

Синтаксис COM:

Object.get_Form(&Form)	Получить свойство
Object.put_Form(Form)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип формы.

Horizontally – Горизонтальное расположение позиций

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Horizontally = Object.Horizontally	Получить свойство(*)
Object.Horizontally = Horizontally	Установить свойство (*)
Horizontally = Object.GetHorizontally( )	Получить свойство (**)
Object.SetHorizontally(Horizontally)	Установить свойство (**)

Синтаксис COM:

Object.get_Horizontally(&Horizontally)	Получить свойство
Object.put_Horizontally(Horizontally)	Установить свойство

Positions – Позиции

Интерфейс...

Тип данных: указатель на интерфейс IText.

Синтаксис Automation:

Positions = Object.Positions	Получить свойство(*)
Positions = Object.GetPositions()	Получить свойство (**)

Синтаксис COM:

Object.get_Positions(&Positions)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста позиций.
2. Свойство доступно только для чтения.

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

```
ShelfDirection = Object.ShelfDirection          Получить свойство(* )
Object.ShelfDirection = ShelfDirection          Установить свойство (* )
ShelfDirection = Object.GetShelfDirection()     Получить свойство (**)
Object.SetShelfDirection( ShelfDirection )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfDirection(                      Получить свойство
&ShelfDirection )
Object.put_ShelfDirection(                      Установить свойство
ShelfDirection )
```

Примечание:

Свойство позволяет устанавливать и получать направление полки.

ShelfVisible – Отрисовка полки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ShelfVisible = Object.ShelfVisible              Получить свойство(* )
Object.ShelfVisible = ShelfVisible              Установить свойство (* )
ShelfVisible = Object.GetShelfVisible( )        Получить свойство (**)
Object.SetShelfVisible( ShelfVisible )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfVisible(                        Получить свойство
&ShelfVisible )
Object.put_ShelfVisible(                        Установить свойство
ShelfVisible )
```

Примечание:

Свойство позволяет устанавливать и получать признак отрисовки полки.

TextDirection – Направление текста

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- вверх,
FALSE	- вниз.

Синтаксис Automation:

TextDirection = Object.TextDirection	Получить свойство(*)
Object.TextDirection = TextDirection	Установить свойство (*)
TextDirection = Object.GetTextDirection( )	Получить свойство (**)
Object.SetTextDirection(TextDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_TextDirection(&TextDirection)	Получить свойство
Object.put_TextDirection(TextDirection)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать направление текста.

UnderPositionText – Текст под позицией

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

UnderPositionText = Object.UnderPositionText	Получить свойство(*)
UnderPositionText = Object.GetUnderPositionText()	Получить свойство (**)

Синтаксис COM:

Object.get_UnderPositionText(&UnderPositionText)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста под позицией.
2. Свойство доступно только для чтения.

Интерфейс IDimensionText

Справка системы КОМПАС: управление размерной надписью

kompas.chm::/176_Glava23_Obshchie_svedenija_.htm

Интерфейс параметров текста размерной надписи.

Иерархия:

IDispatch

IDimensionText

Примечание:

Является дополнительным интерфейсом для 2D и 3D размеров (интерфейсы IAngleDimension3D, IAngleDimension, IArcDimension, IBaseLineDimension3D, IBreakAngleDimension, IBreakLineDimension, IBreakRadialDimension, IDiametralDimension3D, IDiametralDimension, IHeightDimension, ILineDimension3D, ILineDimension, IRadialDimension3D, IRadialDimension). Позволяет сформировать нужный текст размерной надписи.

IDimensionText – свойства

Accuracy – Количество знаков после запятой

Интерфейс...

Тип данных: из перечисления ksAccuracyEnum

Синтаксис Automation:

Accuracy = Object.Accuracy	Получить свойство(*)
Object.Accuracy = Accuracy	Установить свойство (*)
Accuracy = Object.GetAccuracy()	Получить свойство (**)
Object.SetAccuracy(Accuracy)	Установить свойство (**)

Синтаксис COM:

Object.get_Accuracy(&Accuracy)	Получить свойство(*)
Object.put_Accuracy(Accuracy)	Установить свойство (*)

AccuracyDecimalsCount – Количество знаков после запятой, используемое при округлении размера

Интерфейс...

Тип данных: long

Синтаксис Automation:

AccuracyDecimalsCount = Object.AccuracyDecimalsCount	Получить свойство(*)
AccuracyDecimalsCount = Object.GetAccuracyDecimalsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_AccuracyDecimalsCount(AccuracyDecimalsCount)	&	Получить свойство

Примечание:

Свойство доступно только для чтения.

AutoNominalValue – Автоматическое определение номинального значения размера

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoNominalValue = Object.AutoNominalValue	Получить свойство(*)
Object.AutoNominalValue = AutoNominalValue	Установить свойство (*)
AutoNominalValue = Object.GetAutoNominalValue()	Получить свойство (**)
Object.SetAutoNominalValue(AutoNominalValue)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoNominalValue(&AutoNominalValue)	Получить свойство(*)
Object.put_AutoNominalValue(AutoNominalValue)	Установить свойство (*)

Примечание:

По умолчанию признак включен и задает автоматическое вычисление значения размера. Если признак снят, то значение вводили вручную. При необходимости можно вернуть автоматическое вычисление значения размера. Для этого нужно установить флаг в TRUE. При этом отображаемое значение изменится. Точность отображения автоматически вычисленного значения можно задать:

в диалоге настройки точности размерных надписей

kompas.chm::/186_23_5_Nastrojka_razmerov_v_t.htm

Если значение размера введено вручную, то настройка точности на него не влияет.

Brackets – Размер в скобках

Интерфейс...

Тип данных: из перечисления ksDimensionTextBracketsEnum

Синтаксис Automation:

Brackets = Object.Brackets	Получить свойство(*)
Object.Brackets = Brackets	Установить свойство (*)
Brackets = Object.GetBrackets()	Получить свойство (**)
Object.SetBrackets(Brackets)	Установить свойство (**)

Синтаксис COM:

Object.get_Brackets(&Brackets)	Получить свойство(*)
Object.put_Brackets(Brackets)	Установить свойство (*)

Примечание:

Позволяет управлять отображением значений полей **Символ**, **Значение** и **Отклонение** в размерной надписи, обрамленных квадратными или круглыми скобками.

Если для элементов размерной надписи включено отображение рамки и/или подчеркивания, то они также заключаются в скобки.

DeviationOn – Включить отклонения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DeviationOn = Object.DeviationOn	Получить свойство(*)
Object.DeviationOn = DeviationOn	Установить свойство (*)
DeviationOn = Object.GetDeviationOn()	Получить свойство (**)
Object.SetDeviationOn(DeviationOn)	Установить свойство (**)

Синтаксис COM:

Object.get_DeviationOn(&DeviationOn)	Получить свойство(*)
Object.put_DeviationOn(DeviationOn)	Установить свойство (*)

Примечание:

Установите свойство в TRUE, чтобы отклонения или пределы были отрисованы в размерной надписи. Если пределы включены в размерную надпись, а квалитет нет, то номинальное значение в размерной надписи не отображается.

Если отображение предельных значений размера включено, а квалитет не задан, то изменение геометрии размера (например, при перестроении ассоциативного размера) не приводит к пересчету предельных значений.

DeviationType – Тип отклонений

Интерфейс...

Тип данных: Из перечисления ksDimensionDeviationEnum

Синтаксис Automation:

DeviationType = Object.DeviationType	Получить свойство(*)
Object.DeviationType = DeviationType	Установить свойство (*)
DeviationType = Object.GetDeviationType()	Получить свойство (**)
Object.SetDeviationType(DeviationType)	Установить свойство (**)

Синтаксис COM:

Object.get_DeviationType(&DeviationType)	Получить свойство(*)
Object.put_DeviationType(DeviationType)	Установить свойство (*)

Примечание:

Установите значение свойства равным:

- ksDimDeviation, если требуется внести в размерную надпись предельные отклонения размера,
- ksDimLimits, если требуется внести предельные значения размера (предельные значения записываются одно над другим) или
- ksDimLineLimits, если требуется внести предельные значения размера (предельные значения записываются друг за другом через дефис).

Отклонения или предельные значения размера вычисляются автоматически в соответствии с выбранным квалитетом и номинальным значением размера. При необходимости вы можете ввести значения отклонений или пределов вручную.

Если значения отклонений равны по модулю, но отличаются по знаку, то перед значением отклонения отображается знак ±.

После ввода отклонений или пределов вручную поле **Квалитет** очищается.

HasTolerance – Отключить допуск

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HasTolerance = Object.HasTolerance	Получить свойство(*)
Object.HasTolerance = HasTolerance	Установить свойство (*)
HasTolerance = Object.GetHasTolerance()	Получить свойство (**)
Object.SetHasTolerance(HasTolerance)	Установить свойство (**)

Синтаксис COM:

Object.get_HasTolerance(&HasTolerance)	Получить свойство(*)
Object.put_HasTolerance(HasTolerance)	Установить свойство (*)

Примечание:

Свойство позволяет устанавливать и получать признак отключения допуска.

HighDeviation – Верхнее отклонение

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

HighDeviation = Object.HighDeviation	Получить свойство(*)
HighDeviation = Object.GetHighDeviation()	Получить свойство (**)

Синтаксис COM:

Object.get_HighDeviation(&HighDeviation)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать верхнее отклонение или предельное значение для размера.
2. Свойство доступно только для чтения.

HighDeviationValue – Верхнее отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

HighDeviationValue = Object.HighDeviationValue	Получить свойство(*)
HighDeviationValue = Object.GetHighDeviationValue()	Получить свойство (**)

Синтаксис COM:

Object.get_HighDeviationValue(&HighDeviationValue)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LowDeviation – Нижнее отклонение

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

LowDeviation = Object.LowDeviation	Получить свойство(*)
LowDeviation = Object.GetLowDeviation()	Получить свойство (**)

Синтаксис COM:

Object.get_LowDeviation(&LowDeviation)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать нижнее отклонение или предельное значение для размера.
2. Свойство доступно только для чтения.

LowDeviationValue – Нижнее отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

LowDeviationValue = Object.LowDeviationValue	Получить свойство(*)
LowDeviationValue = Object.GetLowDeviationValue()	Получить свойство (**)

Синтаксис COM:

Object.get_LowDeviationValue(&LowDeviationValue)	Получить свойство

Примечание:

Свойство доступно только для чтения.

NominalText – Текст номинального значения

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

NominalText = Object.NominalText	Получить свойство(*)
NominalText = Object.GetNominalText()	Получить свойство (**)

Синтаксис COM:

Object.get_NominalText(&NominalText)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать текст значения размера, заданного пользователем.
2. Для задания текста нужно снять признак IDimensionText::AutoNominalValue.
3. Свойство доступно только для чтения.

NominalValue – Значение размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

NominalValue = Object.NominalValue	Получить свойство(*)
Object.NominalValue = NominalValue	Установить свойство (*)
NominalValue = Object.GetNominalValue()	Получить свойство (**)
Object.SetNominalValue(NominalValue)	Установить свойство (**)

Синтаксис COM:

Object.get_NominalValue(&NominalValue)	Получить свойство(*)
Object.put_NominalValue(NominalValue)	Установить свойство (*)

Примечание:

1. Позволяет получить и установить значение размера.
2. При изменении свойства выполняются следующие действия:

- формируется текстовое значение номинала IDimensionText::NominalText;
- у размера снимается признак автоматического определения номинального значения IDimensionText::AutoNominalValue;

Значение размера при этом не изменяется. Значение размера изменяется при изменении параметров размера, например, при изменении расстояния между выносными линиями линейного размера.

Prefix – Текст до (префикс)

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

Prefix = Object.Prefix	Получить свойство(*)
Prefix = Object.GetPrefix()	Получить свойство (**)

Синтаксис COM:

Object.get_Prefix(&Prefix)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать текст, предшествующий основному содержанию размерной надписи.
2. Свойство доступно только для чтения.

Rectangle – Текст в рамке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Rectangle = Object.Rectangle	Получить свойство(*)
Object.Rectangle = Rectangle	Установить свойство (*)
Rectangle = Object.GetRectangle()	Получить свойство (**)
Object.SetRectangle(Rectangle)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Rectangle( &Rectangle )    Получить свойство(* )
Object.put_Rectangle( Rectangle )     Установить свойство (* )
```

Примечание:

Позволяет управлять отрисовкой рамки вокруг полей **Символ**, **Значение** и **Отклонение** в размерной надписи.

Sign – Номер условного значка перед номиналом

Интерфейс...

Тип данных: long

Значения свойства:

```
0     - нет значка,
1     - диаметр,
2     - квадрат,
3     - K - радиус,
4     - M - метрическая резьба,
>4    - символ из любого шрифта, задаваемого через свойство
        IDimensionText::SignFont (по умолчанию 'Symbol type A').
```

Синтаксис Automation:

```
Sign = Object.Sign              Получить свойство(* )
Object.Sign = Sign              Установить свойство (* )
Sign = Object.GetSign()         Получить свойство (**)
Object.SetSign( Sign )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sign( &Sign )        Получить свойство(* )
Object.put_Sign( Sign )         Установить свойство (* )
```

Примечание:

Позволяет получить установить номер условного значка перед номиналом.

SignFont – Шрифт для условного значка перед номиналом

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
SignFont = Object.SignFont                  Получить свойство(* )
Object.SignFont = SignFont                  Установить свойство (* )
SignFont = Object.GetSignFont()             Получить свойство (**)
Object.SetSignFont( SignFont )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SignFont( &SignFont )    Получить свойство(* )
Object.put_SignFont(  SignFont )    Установить свойство (* )
```

Примечание:

1. Позволяет получить установить шрифт для условного значка перед номиналом.
2. Используется для символов с номером IDimensionText::Sign> 4.
3. По умолчанию используется шрифт Symbol type A.

Style – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                Получить свойство(* )
Object.Style = Style                Установить свойство (* )
Style = Object.GetStyle()           Получить свойство (**)
Object.SetStyle( Style )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )          Получить свойство(* )
Object.put_Style( Style )           Установить свойство (* )
```

Примечание:

Позволяет получить и установить стиль для текстов размера. 0 - умолчательный стиль для текстов размера.

Suffix – Текст после (суффикс)

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

```
Suffix = Object.Suffix              Получить свойство(* )
Suffix = Object.GetSuffix()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Suffix( &Suffix )          Получить свойство(* )
```

Примечание:

1. Свойство позволяет получить и задать текст, следующий за основным содержанием размерной надписи.
2. Свойство доступно только для чтения.

TextAlign – Выравнивание текста

Интерфейс...

Тип данных: из перечисления ksDimensionTextAlignEnum

Синтаксис Automation:

```
TextAlign = Object.TextAlign              Получить свойство(* )
Object.TextAlign = TextAlign              Установить свойство (* )
TextAlign = Object.GetTextAlign()         Получить свойство (**)
Object.SetTextAlign( TextAlign )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_TextAlign( &TextAlign      Получить свойство(* )
)
Object.put_TextAlign(  TextAlign )    Установить свойство (* )
```

Примечание:

Свойство позволяет выбрать способ расположения отклонений или предельных значений относительно номинального значения.

TextFormat – Формат текста

Интерфейс...

Тип данных: из перечисления ksDimTextFormatEnum

Синтаксис Automation:

```
TextFormat = Object.TextFormat            Получить свойство(* )
Object.TextFormat = TextFormat            Установить свойство (* )
TextFormat = Object.GetTextFormat()       Получить свойство (**)
Object.SetTextFormat( TextFormat )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_TextFormat(                Получить свойство(* )
&TextFormat )
```

Object.put_TextFormat(TextFormat)	Установить свойство (*)

Примечание:

1. Свойство позволяет получить и задать формат отображаемого значения размера.
2. Свойство используется для углового размера и углового размера с обрывом.

TextUnder – Текст под размерной надписью

Интерфейс...

Тип данных: интерфейс текста для работы с аннотационными объектами IText

Синтаксис Automation:

TextUnder = Object.TextUnder	Получить свойство(*)
TextUnder = Object.GetTextUnder()	Получить свойство (**)

Синтаксис COM:

Object.get_TextUnder(&TextUnder)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать текст, который располагается под размерной надписью.
2. Свойство доступно только для чтения.

Tolerance – Текст номинального значения

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Tolerance = Object.Tolerance	Получить свойство(*)
Object.Tolerance = Tolerance	Установить свойство (*)
Tolerance = Object.GetTolerance()	Получить свойство (**)
Object.SetTolerance(Tolerance)	Установить свойство (**)

Синтаксис COM:

Object.get_Tolerance(&Tolerance)	Получить свойство(*)
Object.put_Tolerance(Tolerance)	Установить свойство (*)

Примечание:

Свойство позволяет получить и задать квалитет для размера.

ToleranceOn – Включить квалитет

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ToleranceOn = Object.ToleranceOn	Получить свойство(*)
Object.ToleranceOn = ToleranceOn	Установить свойство (*)
ToleranceOn = Object.GetToleranceOn()	Получить свойство (**)
Object.SetToleranceOn(ToleranceOn)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceOn(&ToleranceOn)	Получить свойство(*)
Object.put_ToleranceOn(ToleranceOn)	Установить свойство (*)

Примечание:

Установите свойство в TRUE, чтобы обозначение квалитета было отрисовано в размерной надписи. Значение квалитета может не отображаться, если выбран номер квалитета, для которого отключена вставка в размерную надпись.

Номер, начиная с которого квалитет не вносится в надпись, задается

в диалоге настройки точности размерных надписей

kompas.chm::/186_23_5_Nastrojka_razmerov_v_t.htm

UnderLine – Подчеркнутый текст

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UnderLine = Object.UnderLine	Получить свойство(*)
Object.UnderLine = UnderLine	Установить свойство (*)
UnderLine = Object.GetUnderLine()	Получить свойство (**)
Object.SetUnderLine(UnderLine)	Установить свойство (**)

Синтаксис COM:

Object.get_UnderLine(&UnderLine)	Получить свойство(*)
Object.put_UnderLine(UnderLine)	Установить свойство (*)

Примечание:

Позволяет управлять подчеркиванием значений полей **Символ**, **Значение** и **Отклонение** в размерной надписи.

Unit – Единица измерения

Интерфейс...

Тип данных: интерфейс строки текста ITextLine

Синтаксис Automation:

Unit = Object.Unit	Получить свойство(*)
Unit = Object.GetUnit()	Получить свойство (**)

Синтаксис COM:

Object.get_Unit(&Unit)	Получить свойство(*)

Примечание:

1. Свойство позволяет получить и задать текст для обозначения единиц измерения проставляемого размера.
2. Текст будет отрисован в размерной надписи сразу после предельных отклонений.
3. Свойство доступно только для чтения.

IDimensionText – методы

InitDeviations – Установить отклонения

Интерфейс...

Синтаксис Automation:

BOOL InitDeviations(double HighDeviationValue, double LowDeviationValue);

Синтаксис COM:

HRESULT InitDeviations(double HighDeviationValue, double LowDeviationValue, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

HighDeviationValue	- значение верхнего отклонения,
LowDeviationValue	- значение нижнего отклонения.

Интерфейс IMarkLeader

Справка системы КОМПАС: Обозначение маркировки 2D

kompas.chm::/CM_BRANDLEADER.htm

Обозначение маркировки 3D

kompas.chm::/CM_BRANDLEADER_3D.htm

Интерфейс параметров обозначения маркировки.

Иерархия:

IDispatch

IMarkLeader

IBranchs

IBranchs3D

Описание:

1. Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейсов IBaseLeader или IBaseLeader3D.
2. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс ответвлений IBranchs или IBranchs3D.

IMarkLeader – свойства

Designation – Обозначение

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Designation = Object.Designation	Получить свойство(*)
Designation = Object.GetDesignation()	Получить свойство (**)

Синтаксис COM:

Object.get_Designation(&Designation)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста обозначения.
2. Свойство доступно только для чтения.

TextOnBranch – Текст над ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextOnBranch = Object.TextOnBranch	Получить свойство(*)
TextOnBranch = Object.GetTextOnBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextOnBranch(&TextOnBranch)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста над ножкой.
2. Свойство доступно только для чтения.

TextUnderBranch – Текст под ножкой

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUnderBranch = Object.TextUnderBranch	Получить свойство(*)
TextUnderBranch = Object.GetTextUnderBranch()	Получить свойство (**)

Синтаксис COM:

Object.get_TextUnderBranch(&TextUnderBranch)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста под ножкой.
2. Свойство доступно только для чтения.

Обозначения

Коллекции обозначений

Интерфейс IAssociationTables

Интерфейс коллекции ассоциативных таблиц.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IDrawingObjects

IAssociationTables

Данный интерфейс можно получить у контейнера условных обозначений 2DISymbols2DContainer::AxisLines.

IAssociationTables – свойства

AssociationTable – Получить таблицу по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAssociationTable

Синтаксис Automation:

AssociationTable = Object.AssociationTable(Index)	Получить свойство(*)
AssociationTable = Object.GetAssociationTable(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_AssociationTable(Index, &AssociationTable)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IAssociationTables – методы

Add – Добавить таблицу в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(VARIANT Document, ksReportTypeEnum Type);

Синтаксис COM:

HRESULT Add(VARIANT Document, ksReportTypeEnum Type, IAssociationTable ** Result);

Возвращаемое значение:

- указатель на ассоциативную таблицу IAssociationTable.

Входные параметры:

Document	- имя документа или указатель на документ,
Type	- тип отчета.

Интерфейс IAxisLines

Справка системы КОМПАС...

KOMPAS.chm::/CM_AXEDLINESEG.htm

Интерфейс коллекции осевых линий.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IAxisLines

Описание:

Позволяет создавать и получать осевые линии.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений 2D ISymbols2DContainer::AxisLines.

IAxisLines – свойства

AxisLine – Осевая линия, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс осевой линии IAxisLine

Синтаксис Automation:

AxisLine = iObject.AxisLine (Index);	Получить свойство(*)
AxisLine = iObject.GetAxisLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_AxisLine(Index, &AxisLine) Получить свойство

Примечание:

Свойство доступно только для чтения.

IAxisLines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IAxisLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс осевой линии IAxisLine.

Интерфейс IBases

Справка системы КОМПАС: команда База

KOMPAS.chm::/CM_BASE.htm

Интерфейс коллекции обозначений баз.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBases

Описание:

Интерфейс позволяет получать и создавать обозначение базы.

Примечание:

Получить интерфейс можно, используя свойство контейнера условных обозначений 2D ISymbols2DContainer::Bases.

IBases – свойства

Base – Обозначение базы, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс обозначения базы IBase

Синтаксис Automation:

Base = iObject.Base(Index);	Получить свойство(*)
Base = iObject.GetBase(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Base(Index, &Base)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс обозначения базы.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IBases – методы

Add – Создать обозначение базы (добавить в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IBase ** Result);

Возвращаемое значение:

- Указатель на интерфейс обозначения базы IBase.

Интерфейс IBrokenLines

Справка системы КОМПАС...

KOMPAS.chm::/IDD_BROKENLINE_PARAM_DIALOG.htm

Интерфейс коллекции линий обрыва с изломами.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBrokenLines

Описание:

Позволяет создавать и получать линии обрыва с изломами.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений ISymbols2DContainer::BrokenLines.

IBrokenLines – свойства

BrokenLine – Линия обрыва с изломами, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс линии обрыва с изломами IBrokenLine

Синтаксис Automation:

BrokenLine = iObject.BrokenLine (Index);	Получить свойство(*)
BrokenLine = iObject.GetBrokenLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_BrokenLine(Index, &BrokenLine)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IBrokenLines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IBrokenLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс линии обрыва с изломами IBrokenLine.

Примечание:

Метод позволяет добавить волнистую линию.

Интерфейс ICentreMarkers

Справка системы КОМПАС...

KOMPAS.chm::/CM_CENTRE_MARKER.htm

Интерфейс коллекции обозначений центра.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ICentreMarkers

Описание:

Позволяет создавать и получать обозначения центра.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений 2D ISymbols2DContainer::CentreMarkers.

ICentreMarkers – свойства

CentreMarker – Обозначение центра, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс обозначения центра ICentreMarker

Синтаксис Automation:

CentreMarker = iObject.CentreMarker(Index);	Получить свойство(*)
CentreMarker = iObject.GetCentreMarker(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_CentreMarker(Index, &CentreMarker)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ICentreMarkers – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICentreMarker ** Result);

Возвращаемое значение:

- Указатель на интерфейс обозначения центра ICentreMarker.

Интерфейс ICircularsCentries

Интерфейс коллекции круговых сеток центров.

Справка системы КОМПАС: Команда Круговая сетка центров

`kompas.chm::/CM_CIRCULAR_CENTRES.htm`

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IDrawingObjects
      - ICircularsCentries

Данный интерфейс можно получить, используя свойство ISymbols2DContainer::CircularsCentries.

Версия Компас v18.1

ICircularsCentries – свойства

CircularCentres – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс ICircularCentres.

Синтаксис Automation:

CircularCentres = Object.CircularCentres(Index);	Получить свойство(*)
CircularCentres = Object.GetCircularCentres(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_CircularCentres(Index, &CircularCentres)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ICircularsCentries – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation :

LPDISPATCH Add(VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL Closed, BOOL WithCenter);

Синтаксис COM :

HRESULT Add(VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL Closed, BOOL WithCenter, ICircularCentres * * Result);

Возвращаемое значение:

- Указатель на интерфейс ICircularCentres..

Входные параметры:

BaseObjects	- Базовые окружности.
AutoFindOtherObjects	- Автоматический поиск других подходящих окружностей, центры которых лежат на круговой осевой линии.
Closed	- Замкнуть осевую.
WithCenter	- С обозначением центра.

Примечание

Координаты центра осевой линии определяются по координатам центров первых трех окружностей.

Для остальных окружностей выполняется проверка на одинаковое расстояние от точки центра осевой линии до центра окружности

AddByPoint – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddByPoint(double X0, double Y0, VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL Closed, BOOL WithCenter);

Синтаксис COM:

HRESULT AddByPoint(double X0, double Y0, VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL Closed, BOOL WithCenter, ICircularCentres * * Result);

Возвращаемое значение:

- Указатель на интерфейс ICircularCentres.

Входные параметры:

X0, Y0	- Координаты цента осевой линии.
BaseObjects	- Базовые окружности.
AutoFindOtherObjects	- Автоматический поиск других подходящих окружностей, центры которых лежат на круговой осевой линии.
Closed	- Замкнуть осевую.
WithCenter	- С обозначением центра.

Примечание

Радиус обозначения центра определяется по удалению точки центра первой окружности.

Для остальных окружностей выполняется проверка на одинаковое расстояние от точки центра осевой линии до центра окружности.

Интерфейс IConditionIntersects

Интерфейс коллекции условных пересечений.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IDrawingObjects
      - IConditionIntersects

Примечание:

Получить интерфейс коллекции условных операций можно, используя свойство контейнера условных обозначений 2D ISymbols2DContainer::ConditionIntersects.

Версия: KOMPAS v19

IConditionIntersects – свойства

ConditionIntersect – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IConditionIntersect.

Синтаксис Automation:

ConditionIntersect = Object.ConditionIntersect(Index) Получить свойство (*)
ConditionIntersect = Object.GetConditionIntersect(Index) Получить свойство (**)

Синтаксис COM:

Object.get_ConditionIntersect(Index, &ConditionIntersect) Получить свойство (*)

Входные параметры:

VARIANT Index - индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IConditionIntersects – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IConditionIntersect * * Result);

Возвращаемое значение:

- Указатель на интерфейс IConditionIntersect.

Примечания:

1. Метод позволяет создать новый интерфейс условного пересечения.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IDrawingObject::Update.

Интерфейс ILinearsCentries

Интерфейс коллекции линейных сеток центров.

Справка системы КОМПАС: Команда Линейная сетка центров

`kompas.chm::/CM_LINEAR_CENTRES.htm`

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IDrawingObjects
ILinearsCentries

Данный интерфейс можно получить, используя свойство контейнера объектов 2D ISymbols2DContainer::LinearsCentries

Версия Компас v18.1

ILinearsCentries – свойства

LinearCentres – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс ILinearCentres.

Синтаксис Automation:

LinearCentres = Object.LinearCentres(Index);	Получить свойство(*)
LinearCentres = Object.GetLinearCentres(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_LinearCentres(Index, &LinearCentres)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ILinearsCentries – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL HasBreaks, double AxisAngle, double TurnAngle);

Синтаксис COM:

HRESULT Add(VARIANT BaseObjects, BOOL AutoFindOtherObjects, BOOL HasBreaks, double AxisAngle, double TurnAngle, ILinearCentres * * Result);

Возвращаемое значение:

- Указатель на интерфейс ILinearCentres.

Входные параметры:

BaseObjects	- Базовые окружности.
AutoFindOtherObjects	- Автоматический поиск других подходящих окружностей, центры которых лежат на круговой осевой линии.
HasBreaks	- С разрывами.
AxisAngle	- Наклон первой оси.
TurnAngle	- Угол раствора.

Интерфейс ICutLines

Справка системы КОМПАС: Команда Линия разреза

`kompas.chm::/CM_CUTLINE.htm`

Интерфейс коллекции линий разреза/сечения.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ICutLines

Описание:

Интерфейс позволяет получать и создавать линию разреза/сечения.

Примечание:

1. Интерфейс можно получить, используя свойство контейнера условных обозначений 2D ISymbols2DContainer::CutLines.
2. Кроме того, интерфейс можно получить, используя свойство коллекции линий разреза для СПДС IBuildingContainer::BuildingCutLines.

ICutLines – свойства

CutLine – Линия разреза/сечения, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс линии разреза/сечения ICutLine

Синтаксис Automation:

CutLine = iObject.CutLine(Index);	Получить свойство(*)
CutLine = iObject.GetCutLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_CutLine(&CutLine)	Index,	Получить свойство

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс линии разреза/сечения.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

ICutLines – методы

Add – Создать линию разреза/сечения (добавить в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICutLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс линии разреза/сечения ICutLine.

Интерфейс IDrawingTables

Справка системы КОМПАС: общие сведения о таблицах

kompas.chm::/582_Glava69_Obshchie_svedenija.htm

Интерфейс коллекции таблиц на чертеже.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IDrawingTables

Описание:

Позволяет получать и создавать таблицы на чертеже.

Примечание:

Получить интерфейс коллекции можно используя свойство контейнера условных обозначений ISymbols2DContainer::DrawingTables.

IDrawingTables – свойства

DrawingTable – Указатель на таблицу, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс IDrawingTable

Синтаксис Automation:

DrawingTable = iObject.DrawingTable(Index);	Получить свойство(*)
DrawingTable = iObject.GetDrawingTable(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingTable(Index, &DrawingTable)	Получить свойство

Входные параметры:

Index	- VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IDrawingTables – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(long RowsCount,
long ColumnsCount,
double RowHeigh,
double ColumnsWidth,
long titlePos);

Синтаксис COM:

HRESULT Add(long RowsCount,
long ColumnsCount,
double RowHeigh,
double ColumnsWidth,
ksTableTileLayoutEnum titlePos,
IDrawingTable ** Result);

Входные параметры:

RowsCount	- количество строк таблицы,
ColumnsCount	- количество столбцов таблицы,
RowHeigh	- высота строк,
ColumnsWidth	- ширина столбцов,
titlePos	- расположение заголовка таблицы.

Возвращаемое значение:

- Указатель на интерфейс таблицы на чертеже IDrawingTable.

Load – Загрузить таблицу из файла

Интерфейс...

Синтаксис Automation:

LPDISPATCH Load(BSTR fileName);

Синтаксис COM:

HRESULT Load(BSTR fileName, IDrawingTable ** Result)

Входные параметры:

fileName	- имя файла таблицы.

Возвращаемое значение:

- указатель на интерфейс таблицы на чертеже IDrawingTable.

Интерфейс IDrawingTexts

Справка системы КОМПАС...

KOMPAS.chm::/CM_TEXT.htm

Интерфейс коллекции текстов на чертеже.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IDrawingTexts

Описание:

Интерфейс позволяет получить доступ к текстам на чертеже для указанного вида, а также создавать новые тексты.

Примечание:

Данный интерфейс можно получить у интерфейса IDrawingContainer, используя свойство IDrawingContainer::DrawingTexts.

IDrawingTexts – свойства

DrawingText – Текст на чертеже, заданный по индексу

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_TEXT.htm

Тип данных: указатель на интерфейс текста на чертеже IDrawingText

Синтаксис Automation:

DrawingText = iObject.DrawingText(Index);	Получить свойство(*)
DrawingText = iObject.GetDrawingText(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_DrawingText(Index, &DrawingText);	Получить свойство

Входные параметры:

Index (Variant)	- Индекс текста в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В случае неудачи получения свойства, возвращаемое значение NULL.

IDrawingTexts – методы

Add – Создать текст на чертеже и добавить его в коллекцию

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_TEXT.htm`

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add([out, retval] IDrawingText** Result);

Возвращаемое значение:

- Указатель на интерфейс текста на чертеже IDrawingText.

Примечание:

Метод позволяет добавить текст в коллекцию. После получения нового интерфейса нужно задать параметры текста и вызвать метод IDrawingObject::Update.

Интерфейс ILeaders

Справка системы КОМПАС...

`KOMPAS.chm::/225_28_4_2_Nastrojka_otrisovki_.htm`

Интерфейс коллекции линий-выносок.

Иерархия:

IDispatch
  IKompasAPIObject
    IKompasCollection
      IDrawingObjects
        ILeaders

Описание:

Позволяет получать и создавать линии-выноски.

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера условных обозначений ISymbols2DContainer::Leaders.

ILeaders – свойства

Leader – Указатель на линию-выноску, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс IBaseLeader

Синтаксис Automation:

Leader = Object.Leader(Index)	Получить свойство(*)
Leader = Object.GetLeader(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Leader(&Leader)	Index,	Получить свойство

Входные параметры:

Index	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта - VT_BSTR - имя объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции, reference объекта и имя объекта.
2. Свойство доступно только для чтения.

ILeaders – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(DrawingObjectTypeEnum DimType);

Синтаксис COM:

HRESULT Add(DrawingObjectTypeEnum DimType, IBaseLeader ** Result);

Входные параметры:

DimType - тип линии-выноски.

Типы линий-выносок:

ksDrLeader	20	Простая линия-выноска,
ksDrPosLeader	21	Линия-выноска для обозначения позиции,
ksDrBrandLeader	22	Линия-выноска для обозначения клеймения,
ksDrMarkerLeader	23	Линия-выноска для обозначения маркирования.

Возвращаемое значение:

- Указатель на интерфейс линии выноски IBaseLeader.

Интерфейс IRemoteElements

Справка системы КОМПАС...

`KOMPAS.chm::/CM_REMOTE_ELEMENT.htm`

Интерфейс коллекции выносных элементов.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRemoteElements

Описание:

Позволяет создавать и получать выносные элементы.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений 2D ISymbols2DContainer::RemoteElements.

IRemoteElements – свойства

RemoteElement – Выносной элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс выносного элемента IRemoteElement

Синтаксис Automation:

RemoteElement = iObject.RemoteElement (Index);	Получить свойство(*)
RemoteElement = iObject.GetRemoteElement(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_RemoteElement(Index, &RemoteElement)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IRemoteElements – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRemoteElement ** Result);

Возвращаемое значение:

- Указатель на интерфейс выносного элемента IRemoteElement.

Интерфейс IRoughs

Справка системы КОМПАС: Команда Шероховатость (графический документ)

`kompas.chm::/CM_ROUGH.htm`

Интерфейс коллекции обозначений шероховатости.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRoughs

Описание:

Позволяет получать и создавать обозначения шероховатости. Данный интерфейс можно получить у контейнера условных обозначений 2D, используя свойство ISymbols2DContainer::Roughs.

IRoughs – свойства

Rough – Обозначение шероховатости, заданное по индексу, ссылке или тексту

Интерфейс...

Тип данных: указатель на интерфейс обозначения шероховатости IRough

Синтаксис Automation:

Rough = iObject.Rough(Index);	Получить свойство(*)
Rough = iObject.GetRough(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Rough(&Rough)	Index,	Получить свойство

Входные параметры:

Index (Variant)	- Индекс обозначения узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IRoughs – методы

Add – Добавить обозначение шероховатости в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IRough ** Result);

Возвращаемое значение:

- Указатель на интерфейс обозначения шероховатости IRough.

Интерфейс ITolerances

Справка системы КОМПАС: Команда Допуск формы (графический документ)

kompas.chm::/CM_FORMTOLERANCE.htm

Интерфейс коллекции допусков формы.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ITolerances

Описание:

Позволяет получать и создавать обозначения допуска формы.

Примечание:

Получить интерфейс коллекции можно используя свойство контейнера условных обозначений ISymbols2DContainer::Tolerances.

ITolerances – свойства

Tolerance – Указатель на допуск формы, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ITolerance

Синтаксис Automation:

Tolerance = iObject.Tolerance(Index);	Получить свойство(*)
Tolerance = iObject.GetTolerance(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Tolerance(&Tolerance)	Index,	Получить свойство

Входные параметры:

Index	VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ITolerances – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ITolerance ** Result);

Возвращаемое значение:

- Указатель на интерфейс допуска формы ITolerance.

Интерфейс IViewPointers

Справка системы КОМПАС...

kompas.chm::/CM_VIEWPOINTER.htm

Интерфейс коллекции стрелок взгляда.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IViewPointers

Описание:

Позволяет получать и создавать стрелки взгляда.

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера условных обозначений ISymbols2DContainer::ViewPointers.

IViewPointers – свойства

ViewPointer – Указатель на стрелку взгляда, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс IViewPointer

Синтаксис Automation:

ViewPointer = iObject.ViewPointer(Index);	Получить свойство(*)
ViewPointer = iObject.GetViewPointer(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_ViewPointer(Index, &ViewPointer)	Получить свойство

Входные параметры:

Index	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта - VT_BSTR - имя объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции, reference объекта и имя объекта.
2. Свойство доступно только для чтения.

IViewPointers – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ViewPointer ** Result);

Возвращаемое значение:

- Указатель на интерфейс стрелки взгляда ViewPointer.

Интерфейс IWaveLines

Справка системы КОМПАС...

`KOMPAS.chm::/CM_WAVELINE.htm`

Интерфейс коллекции волнистых линий.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IDrawingObjects
      - IWaveLines

Описание:

Интерфейс позволяет создавать и получать волнистые линии линии.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений ISymbols2DContainer::WaveLines.

IWaveLines – свойства

WaveLine – Волнистая линия, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс волнистой линии IWaveLine

Синтаксис Automation:

WaveLine = iObject.WaveLine (Index);	Получить свойство(*)
WaveLine = iObject.GetWaveLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_WaveLine(Index, &WaveLine)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IWaveLines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IWaveLine ** Result);

Возвращаемое значение:

	- Указатель на интерфейс волнистой линии IWaveLine.

Примечание:

Метод позволяет добавить волнистую линию.

Интерфейс IAssociationTable

Интерфейс ассоциативной таблицы.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IAssociationTable

IAssociationTable – свойства

Actual – Актуальность таблицы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Actual = Object.Actual	Получить свойство(*)
Actual = Object.GetActual()	Получить свойство (**)

Синтаксис COM:

Object.get_Act ual(&Actual)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Report – Получить интерфейс отчета

Интерфейс...

Тип данных: Указатель на интерфейс IReport

Синтаксис Automation:

Report = Object.Report	Получить свойство(*)
Report = Object.GetReport()	Получить свойство (**)

Синтаксис COM:

Object.get_Report(&Report)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TablePlaceType – Тип привязки таблицы

Интерфейс...

Тип данных: из перечисления ksTablePointEnum

Синтаксис Automation:

TablePlaceType = Object.TablePlaceType	=	Получить свойство(*)
Object.TablePlaceType = TablePlaceType	=	Установить свойство (*)
TablePlaceType = Object.GetTablePlaceType()	=	Получить свойство (**)
Object.SetTablePlaceType(TablePlaceType)		Установить свойство (**)

Синтаксис COM:

Object.get_TablePlaceType(&TablePlaceType)	Получить свойство
Object.put_TablePlaceType(TablePlaceType)	Установить свойство

X – Координата точки привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Y – Координата точки привязки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)

Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

IAssociationTable – методы

Rebuild – Перестроить отчет

Интерфейс...

Синтаксис Automation:

BOOL Rebuild();

Синтаксис COM:

HRESULT Rebuild(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IAxisLine

Справка системы КОМПАС...

`KOMPAS.chm::/CM_AXEDLINESEG.htm`

Интерфейс осевой линии.

Иерархия:

IDispatch
- IKompasAPIObject
  - IDrawingObject
    - IAxisLine
      - IAxisLineParam

Описание:

Интерфейс позволяет задавать параметры осевой линии.

Примечание:

1. Интерфейс можно получить у коллекции осевых линий, используя свойство IAxisLines::AxisLine или метод IAxisLines::Add.
2. После задания параметров осевой линии требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс IAxisLineParam.

IAxisLine – свойства

Angle – Угол между осевой и осью OX текущей СК

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол между осевой и осью OX текущей СК.

Length – Длина

Интерфейс...

Тип данных: double

Синтаксис Automation:

Length = Object.Length	Получить свойство(*)
Object.Length = Length	Установить свойство (*)
Length = Object.GetLength()	Получить свойство (**)
Object.SetLength(Length)	Установить свойство (**)

Синтаксис COM:

Object.get_Length(&Length)	Получить свойство
Object.put_Length(Length)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину осевой линии.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)

```
Object.X1 = X1                    Установить свойство (* )
X1 = Object.GetX1()               Получить свойство (**)
Object.SetX1( X1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )              Получить свойство
Object.put_X1(  X1 )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки осевой линии по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                    Получить свойство(* )
Object.X2 = X2                    Установить свойство (* )
X2 = Object.GetX2()               Получить свойство (**)
Object.SetX2( X2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )              Получить свойство
Object.put_X2(  X2 )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату второй точки осевой линии по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1                    Получить свойство(* )
Object.Y1 = Y1                    Установить свойство (* )
Y1 = Object.GetY1()               Получить свойство (**)
Object.SetY1( Y1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )              Получить свойство
Object.put_Y1(  Y1 )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки осевой линии по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату второй точки осевой линии по оси Y.

Интерфейс IBase

Справка системы КОМПАС: команда База

`KOMPAS.chm::/CM_BASE.htm`

Интерфейс обозначения базы.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IBase

Примечание:

1. Интерфейс позволяет получить и задать свойства объекта "база".
2. Получить интерфейс можно, используя свойство IBases::Base или метод IBases::Add интерфейса коллекции обозначений базы.
3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

IBase – свойства

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- включить опцию,
FALSE	- отключить опцию.

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание:

При отключенной опции доступно редактирование текста базы с помощью свойства IBase::Text.

BaseObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство

Object.put_BaseObject(BaseObject)	Установить свойство

Примечание:

После указания базового объекта все устанавливаемые объекту координаты положения знака на поверхности будут проецироваться на базовый объект или его продолжение.

BranchX – Координата X конечной точки выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchX = Object.BranchX	Получить свойство(*)
Object.BranchX = BranchX	Установить свойство (*)
BranchX = Object.GetBranchX()	Получить свойство (**)
Object.SetBranchX(BranchX)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchX(&BranchX)	Получить свойство
Object.put_BranchX(BranchX)	Установить свойство

BranchY – Координата Y конечной точки выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchY = Object.BranchY	Получить свойство(*)
Object.BranchY = BranchY	Установить свойство (*)
BranchY = Object.GetBranchY()	Получить свойство (**)
Object.SetBranchY(BranchY)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchY(&BranchY)	Получить свойство
Object.put_BranchY(BranchY)	Установить свойство

DrawType – Способ отрисовки обозначения базы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DrawType = Object.DrawType	Получить свойство(*)
Object.DrawType = DrawType	Установить свойство (*)

DrawType = Object.GetDrawType()	Получить свойство (**)
Object.SetDrawType(DrawType)	Установить свойство (**)

Синтаксис COM:

Object.get_DrawType(&DrawType)	Получить свойство
Object.put_DrawType(DrawType)	Установить свойство

Text – Текст базы

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = Object.Text	Получить свойство(*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста базы.
3. Указатель на интерфейс IText вернётся только в случае отключенной опции Автосортировка. Если автосортировка текста включена, свойство вернёт NULL.

X0 – Координата X положения знака на поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

X0 = Object.X0	Получить свойство(*)
Object.X0 = X0	Установить свойство (*)
X0 = Object.GetX0()	Получить свойство (**)
Object.SetX0(X0)	Установить свойство (**)

Синтаксис COM:

Object.get_X0(&X0)	Получить свойство
Object.put_X0(X0)	Установить свойство

Y0 – Координата Y положения знака на поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y0 = Object.Y0	Получить свойство(*)
Object.Y0 = Y0	Установить свойство (*)
Y0 = Object.GetY0()	Получить свойство (**)
Object.SetY0(Y0)	Установить свойство (**)

Синтаксис COM:

Object.put_Y0(Y0)	Получить свойство
Object.put_Y0(Y0)	Установить свойство

Интерфейс IBaseLeader

Справка системы КОМПАС: команда Линия-выноска (графический документ)

KOMPAS.chm::/CM_LEADER.htm

Общие сведения

KOMPAS.chm::/223_28_4_Linija-vynoska.htm

Интерфейс обозначения линии-выноски 2D.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IBaseLeader

ILeader

IPositionLeader

IBrandLeader

IMarkLeader

IBranchs

IChangeLeader

Примечание:

1. Получить интерфейс можно, используя свойство интерфейса коллекции линий-выносок ILeaders::Leader или метод ILeaders::Add.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительные интерфейсы:

Объект	DrawingObjectTypeEnum	Интерфейс
Простая линия-выноска	ksDrLeader	ILeader
Линия-выноска для обозначения позиции	ksDrPosLeader	IPositionLeader
Линия-выноска для для обозначения клеймения	ksDrBrandLeader	IBrandLeader
Линия-выноска для обозначения маркирования	ksDrMarkLeader	IMarkLeader
Знак изменения	ksDrChangeLeader	IChangeLeader
Ответвления		IBranchs

3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

IBaseLeader – свойства

ArrowType – Тип стрелки линии-выноски

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = Object.ArrowType	Получить свойство(*)
Object.ArrowType = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType()	Получить свойство (**)
Object.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство
Object.put_ArrowType(ArrowType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип стрелки линии-выноски.

Интерфейс IBrokenLine

Справка системы КОМПАС...

`KOMPAS.chm::/263_28_18_Linija_s_izlomami.htm`

Интерфейс линии обрыва с изломами.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IBrokenLine

Описание:

Интерфейс позволяет задавать и получать параметры линии обрыва с изломами.

Примечание:

1. Интерфейс можно получить у коллекции линий обрыва с изломами используя свойство IBrokenLines::BrokenLine или метод IBrokenLines::Add.
2. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

IBrokenLine – свойства

Angle – Угол между линией и осью OX текущей СК

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол между линией обрыва с изломами и осью OX текущей СК.

AutoBreakAmplitude – Использовать значение амплитуды излома из настроек документа

Интерфейс...

Тип данных: double

Синтаксис Automation:

AutoBreakAmplitude = Object.AutoBreakAmplitude	=	Получить свойство(*)
Object.AutoBreakAmplitude = AutoBreakAmplitude	=	Установить свойство (*)
AutoBreakAmplitude = Object.GetAutoBreakAmplitude()	=	Получить свойство (**)

Object.SetAutoBreakAmplitude(AutoBreakAmplitude)		Установить свойство (**)

Синтаксис COM:

Object.get_AutoBreakAmplitude(&AutoBreakAmplitude)	Получить свойство
Object.put_AutoBreakAmplitude(AutoBreakAmplitude)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования амплитуды излома из настроек документа.

AutoJutValue – Использовать значение выступа линии из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoJutValue = Object.AutoJutValue	=	Получить свойство(*)
Object.AutoJutValue = AutoJutValue	=	Установить свойство (*)
AutoJutValue = Object.GetAutoJutValue()	=	Получить свойство (**)
Object.SetAutoJutValue(AutoJutValue)		Установить свойство (**)

Синтаксис COM:

Object.get_AutoJutValue(&AutoJutValue)	Получить свойство
Object.put_AutoJutValue(AutoJutValue)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования значения выступа линии из настроек документа.

BreakAmplitude – Амплитуда излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakAmplitude Object.BreakAmplitude	=	Получить свойство(*)
Object.BreakAmplitude BreakAmplitude	=	Установить свойство (*)
BreakAmplitude Object.GetBreakAmplitude()	=	Получить свойство (**)
Object.SetBreakAmplitude(BreakAmplitude)		Установить свойство (**)

Синтаксис COM:

Object.get_BreakAmplitude(&BreakAmplitude)	Получить свойство
Object.put_BreakAmplitude(BreakAmplitude)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать амплитуду излома.

BreakDisplacement – Смещение излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakDisplacement Object.BreakDisplacement	=	Получить свойство(*)
Object.BreakDisplacement BreakDisplacement	=	Установить свойство (*)
BreakDisplacement Object.GetBreakDisplacement()	=	Получить свойство (**)
Object.SetBreakDisplacement(BreakDisplacement)		Установить свойство (**)

Синтаксис COM:

Object.get_BreakDisplacement(&BreakDisplacement)	Получить свойство
Object.put_BreakDisplacement(BreakDisplacement)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать смещение излома.

BreaksCount – Количество изломов

Интерфейс...

Тип данных: long

Синтаксис Automation:

BreaksCount = Object.BreaksCount	=	Получить свойство(*)
Object.BreaksCount = BreaksCount	=	Установить свойство (*)
BreaksCount = Object.GetBreaksCount()	=	Получить свойство (**)
Object.SetBreaksCount(BreaksCount)		Установить свойство (**)

Синтаксис COM:

Object.get_BreaksCount(&BreaksCount)	Получить свойство
Object.put_BreaksCount(BreaksCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество изломов.

JutValue – Значение выступа линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

JutValue = Object.JutValue	Получить свойство(*)
Object.JutValue = JutValue	Установить свойство (*)
JutValue = Object.GetJutValue()	Получить свойство (**)
Object.SetJutValue(JutValue)	Установить свойство (**)

Синтаксис COM:

Object.get_JutValue(&JutValue)	Получить свойство
Object.put_JutValue(JutValue)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать значение выступа линии.

Length – Длина

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = Object.Length                  Получить свойство(* )
Object.Length = Length                  Установить свойство (* )
Length = Object.GetLength()             Получить свойство (**)
Object.SetLength( Length )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Length( &Length )            Получить свойство
Object.put_Length(  Length )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать длину линии обрыва с изломами.

Style – Стиль линии

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = Object.Style                    Получить свойство(* )
Object.Style = Style                    Установить свойство (* )
Style = Object.GetStyle()               Получить свойство (**)
Object.SetStyle( Style )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )              Получить свойство
Object.put_Style(  Style )              Установить свойство
```

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

Type1 – Тип

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE                                    - Тип 1,
FALSE                                   - Тип 2.
```

Синтаксис Automation:

```
Type1 = Object.Type1                    Получить свойство(* )
Object.Type1 = Type1                    Установить свойство (* )
```

```
Type1 = Object.GetType1()          Получить свойство (**)
Object.SetType1( Type1 )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Type1( &Type1 )         Получить свойство
Object.put_Type1(  Type1 )         Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать тип линии обрыва с изломами.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X1                     Получить свойство(* )
Object.X1 = X1                     Установить свойство (* )
X1 = Object.GetX1()                Получить свойство (**)
Object.SetX1( X1 )                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )               Получить свойство
Object.put_X1(  X1 )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату первой точки по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                     Получить свойство(* )
Object.X2 = X2                     Установить свойство (* )
X2 = Object.GetX2()                Получить свойство (**)
Object.SetX2( X2 )                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )               Получить свойство
Object.put_X2(  X2 )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать координату второй точки по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1()	Получить свойство (**)
Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство
Object.put_Y1(Y1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату первой точки по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату второй точки по оси Y.

Интерфейс ICentreMarker

Справка системы КОМПАС...

KOMPAS.chm::/CM_CENTRE_MARKER.htm

Интерфейс обозначения центра.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

ICentreMarker

IAxisLineParam

Описание:

Интерфейс позволяет задавать параметры обозначения центра.

Примечание:

1. Интерфейс можно получить у коллекции обозначений центра, используя свойство ICentreMarkers::CentreMarker или метод ICentreMarkers::Add.
2. После задания параметров обозначения центра требуется вызвать метод IDrawingObject::Update.
3. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс IAxisLineParam.

CentreMarker – свойства

Angle – Угол наклона обозначения центра

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол наклона обозначения центра.

BaseObject – Базовая кривая

Интерфейс...

Тип данных: указатель на интерфейс графического объекта IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать базовую кривую обозначения центра.

CrosshairSize – Размер обозначения крестика

Интерфейс...

Тип данных: double

Синтаксис Automation:

CrosshairSize = Object.CrosshairSize	Получить свойство(*)
Object.CrosshairSize = CrosshairSize	Установить свойство (*)
CrosshairSize = Object.GetCrosshairSize()	Получить свойство (**)
Object.SetCrosshairSize(CrosshairSize)	Установить свойство (**)

Синтаксис COM:

Object.get_CrosshairSize(&CrosshairSize)	Получить свойство
Object.put_CrosshairSize(CrosshairSize)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать размер обозначения крестика.

CrosshairSizeModify – Использовать размер обозначения крестика из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CrosshairSizeModify = Object.CrosshairSizeModify	Получить свойство(*)
Object.CrosshairSizeModify = CrosshairSizeModify	Установить свойство (*)
CrosshairSizeModify = Object.GetCrosshairSizeModify()	Получить свойство (**)
Object.SetCrosshairSizeModify(CrosshairSizeModify)	Установить свойство (**)

Синтаксис COM:

Object.get_CrosshairSizeModify(&CrosshairSizeModify)	Получить свойство
Object.put_CrosshairSizeModify(CrosshairSizeModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования крестика из настроек документа.

SemiAxisAutoLength – Ассоциативное задание длины полуоси

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SemiAxisAutoLength = Object.SemiAxisAutoLength	Получить свойство(*)
Object.SemiAxisAutoLength = SemiAxisAutoLength	Установить свойство (*)
SemiAxisAutoLength = Object.GetSemiAxisAutoLength()	Получить свойство (**)
Object.SetSemiAxisAutoLength(SemiAxisAutoLength)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisAutoLength (&SemiAxisAutoLength)	Получить свойство
Object.put_SemiAxisAutoLength (SemiAxisAutoLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак ассоциативного задания длины полуоси.

SemiAxisLength – Длина полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

SemiAxisLength = Object.SemiAxisLength	Получить свойство(*)
Object.SemiAxisLength = SemiAxisLength	Установить свойство (*)
SemiAxisLength = Object.GetSemiAxisLength()	Получить свойство (**)
Object.SetSemiAxisLength(SemiAxisLength)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisLength(&SemiAxisLength)	Получить свойство
Object.put_SemiAxisLength(SemiAxisLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину полуоси обозначения центра.

SignType – Тип обозначения центра

Интерфейс...

Тип данных: из перечисления ksCentreMarkerEnum

Синтаксис Automation:

SignType = Object.SignType	Получить свойство(*)
Object.SignType = SignType	Установить свойство (*)
SignType = Object.GetSignType()	Получить свойство (**)
Object.SetSignType(SignType)	Установить свойство (**)

Синтаксис COM:

Object.get_SignType(&SignType)	Получить свойство
Object.put_SignType(SignType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип обозначения центра.

X – Координата точки вставки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки вставки по оси X.

Y – Координата точки вставки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки вставки по оси Y.

Интерфейс ICircularCentres

Интерфейс параметров круговой сетки центров

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

ICircularCentres

Данный интерфейс можно получить с помощью методов коллекции круговых сеток центров ICircularsCentries::Add и ICircularsCentries::AddByPoint или свойства ICircularsCentries::CircularCentres.

Версия Компас v18.1

ICircularCentres – свойства

AxisesCount – Количество осевых линий в обозначении

Интерфейс...

Тип данных: long

Синтаксис Automation:

AxisesCount = Object.AxisesCount(WithCenter)	Получить свойство(*)
AxisesCount = Object.GetAxisesCount(WithCenter)	Получить свойство (**)

Синтаксис COM:

Object.get_AxisesCount(WithCenter, &AxisesCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BaseObjects – Объекты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BaseObjects = Object.BaseObjects	Получить свойство(*)
Object.BaseObjects = BaseObjects	Установить свойство (*)
BaseObjects = Object.GetBaseObjects()	Получить свойство (**)
Object.SetBaseObjects(BaseObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjects(&BaseObjects)	Получить свойство
Object.put_BaseObjects(BaseObjects)	Установить свойство

Примечание.

Позволяет получать и устанавливать опорный объект или массив опорных объектов.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Centres – Координаты центров осевых линий

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Centres = Object.Centres                    Получить свойство(* )
Centres = Object.GetCentres()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Centres( &Centres )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Closed – Замкнуть осевую

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Closed = Object.Closed                      Получить свойство(* )
Object.Closed = Closed                      Установить свойство (* )
Closed = Object.GetClosed()                 Получить свойство (**)
Object.SetClosed( Closed )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Closed( &Closed )                Получить свойство
Object.put_Closed(  Closed )                Установить свойство
```

Radiuses – Радиусы дуг и окружностей

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Radiuses = Object.Radiuses                  Получить свойство(* )
Radiuses = Object.GetRadiuses()             Получить свойство (**)
```

Синтаксис COM:

Object.get_Radiuses(&Radiuses)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SemiAxisAutoLength – TRUE – ассоциативное задание длины полуоси

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SemiAxisAutoLength = Object.SemiAxisAutoLength(WithCenter, Index, AxisType)	Получить свойство(*)
Object.SemiAxisAutoLength(WithCenter, Index, AxisType) = SemiAxisAutoLength	Установить свойство (*)
SemiAxisAutoLength = Object.GetSemiAxisAutoLength(WithCenter, Index, AxisType)	Получить свойство (**)
Object.SetSemiAxisAutoLength(WithCenter, Index, AxisType, SemiAxisAutoLength)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisAutoLength (WithCenter, Index, AxisType, &SemiAxisAutoLength)	Получить свойство
Object.put_SemiAxisAutoLength (WithCenter, Index, AxisType, SemiAxisAutoLength)	Установить свойство

Входные параметры:

BOOL	WithCenter - TRUE, устанавливать параметры центрального обозначения.
long	Index - индекс оси.
ksSemiAxisTypeEnum	AxisType - индекс полуоси.

SemiAxisLength – Длина полуоси

Интерфейс...

Тип данных: double

Синтаксис Automation:

SemiAxisLength = Object.SemiAxisLength(WithCenter, Index, AxisType)	Получить свойство(*)
Object.SemiAxisLength(WithCenter, Index, AxisType) = SemiAxisLength	Установить свойство (*)
SemiAxisLength = Object.GetSemiAxisLength(WithCenter, Index, AxisType)	Получить свойство (**)
Object.SetSemiAxisLength(WithCenter, Index, AxisType, SemiAxisLength)	Установить свойство (**)

Синтаксис COM:

Object.get_SemiAxisLength(WithCenter, Index, AxisType, &SemiAxisLength)	Получить свойство
Object.put_SemiAxisLength(WithCenter, Index, AxisType, SemiAxisLength)	Установить свойство

Входные параметры:

BOOL	WithCenter - TRUE, устанавливать параметры центрального обозначения.
long	Index - индекс оси.
ksSemiAxisTypeEnum	AxisType - индекс полуоси.

X0 – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X0 = Object.X0	Получить свойство(*)
Object.X0 = X0	Установить свойство (*)
X0 = Object.GetX0()	Получить свойство (**)
Object.SetX0(X0)	Установить свойство (**)

Синтаксис COM:

Object.get_X0(&X0)	Получить свойство
Object.put_X0(X0)	Установить свойство

Y0 – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y0 = Object.Y0	Получить свойство(*)
Object.Y0 = Y0	Установить свойство (*)
Y0 = Object.GetY0()	Получить свойство (**)
Object.SetY0(Y0)	Установить свойство (**)

Синтаксис COM:

Object.get_Y0(&Y0)	Получить свойство
Object.put_Y0(Y0)	Установить свойство

WithCenter – С обозначением центра

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithCenter = Object.WithCenter	Получить свойство(*)
Object.WithCenter = WithCenter	Установить свойство (*)
WithCenter = Object.GetWithCenter()	Получить свойство (**)
Object.SetWithCenter(WithCenter)	Установить свойство (**)

Синтаксис COM:

Object.get_WithCenter(&WithCenter)	Получить свойство
Object.put_WithCenter(WithCenter)	Установить свойство

ICircularCentres – методы

AddCentre – Добавить обозначение центра в сетку

Интерфейс...

Синтаксис Automation:

BOOL AddCentre(double X, double Y, double Radius);

Синтаксис COM :

HRESULT AddCentre(double X, double Y, double Radius, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

X, Y	- координаты центра окружности.
Radius	- радиус.

AddCentreByObject – Добавить обозначение центра в сетку

Интерфейс...

Синтаксис Automation:

BOOL AddCentreByObject(IKompasAPIObject * Object);

Синтаксис COM :

HRESULT AddCentreByObject(IKompasAPIObject * Object, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Object	- указатель на окружность.

Clear – Очистить список центров

Интерфейс...

Синтаксис Automation :

BOOL Clear();

Синтаксис COM :

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

DeleteCentre – Удалить обозначение центра из сетки

Интерфейс...

Синтаксис Automation:

BOOL DeleteCentre(long Index);

Синтаксис COM:

HRESULT DeleteCentre(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Index	- индекс обозначения центра.

DeleteCentreByPoint – Удалить обозначение центра из сетки по координатам

Интерфейс...

Синтаксис Automation:

BOOL DeleteCentreByPoint(double X, double Y);

Синтаксис COM :

HRESULT DeleteCentreByPoint(double X, double Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

X, Y	- координаты обозначения центра.

Интерфейс IConditionIntersect

Интерфейс условного пересечения.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IConditionIntersect

Примечание:

Интерфейс можно получить с помощью метода коллекции условных пересечений IConditionIntersects::Add.

KOMPAS v19

IConditionIntersect – свойства

AssociationObject – Получить кривую

Интерфейс...

Тип данных: Указатель на интерфейс IDrawingObject.

Синтаксис Automation:

AssociationObject Object.AssociationObject(First)	=	Получить свойство (*)
AssociationObject Object.GetAssociationObject(First)	=	Получить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject)	First,	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Gap – Зазор или длина выносной линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Gap = Object.Gap	Получить свойство (*)
Object.Gap = Gap	Установить свойство (*)
Gap = Object.GetGap()	Получить свойство (**)
Object.SetGap(Gap)	Установить свойство (**)

Синтаксис COM:

Object.get_Gap(&Gap)	Получить свойство,
Object.put_Gap(Gap)	Установить свойство.

GapValue – Значение зазора или длины

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
GapValue = Object.GapValue                  Получить свойство (* )
Object.GapValue = GapValue                  Установить свойство (* )
GapValue = Object.GetGapValue()             Получить свойство (**)
Object.SetGapValue( GapValue )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_GapValue( &GapValue )        Получить свойство,
Object.put_GapValue(  GapValue )        Установить свойство.
```

PointStyle – Стиль точки

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum.

Синтаксис Automation:

```
PointStyle = Object.PointStyle              Получить свойство (* )
Object.PointStyle = PointStyle              Установить свойство (* )
PointStyle = Object.GetPointStyle()         Получить свойство (**)
Object.SetPointStyle( PointStyle )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PointStyle( &PointStyle )    Получить свойство,
Object.put_PointStyle(  PointStyle )    Установить свойство.
```

PointVisible – Отрисовывать точку пересечения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
PointVisible = Object.PointVisible          Получить свойство (* )
Object.PointVisible = PointVisible          Установить свойство (* )
PointVisible = Object.GetPointVisible()     Получить свойство (**)
Object.SetPointVisible( PointVisible )      Установить свойство (**)
```

Синтаксис COM:

Object.get_PointVisible(&PointVisible)	Получить свойство,
Object.put_PointVisible(PointVisible)	Установить свойство.

RemoteLine1Visible – Признак отрисовки первой выносной линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

RemoteLine1Visible = Object.RemoteLine1Visible	Получить свойство (*)
Object.RemoteLine1Visible = RemoteLine1Visible	Установить свойство (*)
RemoteLine1Visible = Object.GetRemoteLine1Visible()	Получить свойство (**)
Object.SetRemoteLine1Visible(RemoteLine1Visible)	Установить свойство (**)

Синтаксис COM:

Object.get_RemoteLine1Visible(&RemoteLine1Visible)	Получить свойство,
Object.put_RemoteLine1Visible(RemoteLine1Visible)	Установить свойство.

RemoteLine2Visible – Признак отрисовки второй выносной линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

RemoteLine2Visible = Object.RemoteLine2Visible	Получить свойство (*)
Object.RemoteLine2Visible = RemoteLine2Visible	Установить свойство (*)
RemoteLine2Visible = Object.GetRemoteLine2Visible()	Получить свойство (**)
Object.SetRemoteLine2Visible(RemoteLine2Visible)	Установить свойство (**)

Синтаксис COM:

Object.get_RemoteLine2Visible(&RemoteLine2Visible)	Получить свойство,
Object.put_RemoteLine2Visible(RemoteLine2Visible)	Установить свойство.

IConditionIntersect – методы

GetCurvePoint – Получить точку на кривой

Интерфейс...

Синтаксис Automation:

BOOL GetCurvePoint(BOOL FirstCurve, double * X, double * Y);

Синтаксис COM :

HRESULT GetCurvePoint(BOOL FirstCurve, double * X, double * Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

FirstCurve	- получить точку первой кривой.

Выходные параметры:

X	- первая координата точки,
Y	- вторая координата точки.

GetIntersectPoint – Получить точку пересечения

Интерфейс...

Синтаксис Automation:

BOOL GetIntersectPoint(double * X, double * Y);

Синтаксис COM:

HRESULT GetIntersectPoint(double * X, double * Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Выходные параметры:

X	- первая координата точки,
Y	- вторая координата точки.

InitByObjects – Инициализация по объектам

Интерфейс...

Синтаксис Automation:

BOOL InitByObjects(LPDISPATCH Curve1, LPDISPATCH Curve2, double * X1, double * Y1, double * X2, double * Y2, BOOL RemoteLine1Visible, BOOL RemoteLine2Visible);

Синтаксис COM:

HRESULT InitByObjects(IKompasAPIObject * Curve1, IKompasAPIObject * Curve2, double * X1, double * Y1, double * X2, double * Y2, BOOL RemoteLine1Visible, BOOL RemoteLine2Visible, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Curve1	- первая кривая,
Curve2	- вторая кривая,
X1, Y1	- координаты точки на первой кривой,
X2, Y2	- координаты точки на второй кривой,
RemoteLine1Visible	- признак видимости первой кривой.

Интерфейс ILinearCentres

Интерфейс параметров линейной сетки центров.

Иерархия:

IDispatch
- IKompasAPIObject
  - IDrawingObject
    - ILinearCentres

Данный интерфейс можно получить с помощью метода коллекции линейных сеток центров ILinearsCentries::Add или свойства ILinearsCentries::LinearCentres.

Версия Компас v18.1

ILinearCentres – свойства

AxisAngle – Наклон первой оси

Интерфейс...

Тип данных: double

Синтаксис Automation:

AxisAngle = Object.AxisAngle	Получить свойство(*)
Object.AxisAngle = AxisAngle	Установить свойство (*)
AxisAngle = Object.GetAxisAngle()	Получить свойство (**)
Object.SetAxisAngle(AxisAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_AxisAngle(&AxisAngle)	Получить свойство
Object.put_AxisAngle(AxisAngle)	Установить свойство

BaseObjects – Объекты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BaseObjects = Object.BaseObjects	Получить свойство(*)
Object.BaseObjects = BaseObjects	Установить свойство (*)
BaseObjects = Object.GetBaseObjects()	Получить свойство (**)
Object.SetBaseObjects(BaseObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjects(&BaseObjects)	Получить свойство
Object.put_BaseObjects(BaseObjects)	Установить свойство

Примечание.

Позволяет получать и устанавливать опорный объект или массив опорных объектов.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Centres – Координаты центров осевых линий

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Centres = Object.Centres	Получить свойство(*)
Centres = Object.GetCentres()	Получить свойство (**)

Синтаксис COM:

Object.get_Centres(&Centres)	Получить свойство

Примечание:

Свойство доступно только для чтения.

HasBreaks – С разрывами

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HasBreaks = Object.HasBreaks	Получить свойство(*)
Object.HasBreaks = HasBreaks	Установить свойство (*)
HasBreaks = Object.GetHasBreaks()	Получить свойство (**)
Object.SetHasBreaks(HasBreaks)	Установить свойство (**)

Синтаксис COM:

Object.get_HasBreaks(&HasBreaks)	Получить свойство
Object.put_HasBreaks(HasBreaks)	Установить свойство

Radiuses – Радиусы дуг и окружностей

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Radiuses = Object.Radiuses	Получить свойство(*)
Radiuses = Object.GetRadiuses()	Получить свойство (**)

Синтаксис COM:

Object.get_Radiuses(&Radiuses)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TurnAngle – Угол раствора

Интерфейс...

Тип данных: double

Синтаксис Automation:

TurnAngle = Object.TurnAngle	Получить свойство(*)
Object.TurnAngle = TurnAngle	Установить свойство (*)
TurnAngle = Object.GetTurnAngle()	Получить свойство (**)
Object.SetTurnAngle(TurnAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_TurnAngle(&TurnAngle)	Получить свойство
Object.put_TurnAngle(TurnAngle)	Установить свойство

ILinearCentres – методы

AddCentre – Добавить обозначение центра в сетку

Интерфейс...

Синтаксис Automation:

BOOL AddCentre(double X, double Y, double Radius);

Синтаксис COM:

HRESULT AddCentre(double X, double Y, double Radius, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

X, Y	- координаты обозначения центра.
Radius	- радиус.

AddCentreByObject – Добавить обозначение центра в сетку

Интерфейс...

Синтаксис Automation:

BOOL AddCentreByObject(IKompasAPIObject * Object);

Синтаксис COM:

HRESULT AddCentreByObject(IKompasAPIObject * Object, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Object	- Указатель на окружность.

Clear – Очистить список центров

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

DeleteCentre – Удалить обозначение центра в сетку

Интерфейс...

Синтаксис Automation

BOOL DeleteCentre(long Index);

Синтаксис COM:

HRESULT DeleteCentre(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Index	- индекс обозначения центра.

DeleteCentreByPoint – Удалить обозначение центра из сетки по координатам

Интерфейс...

Синтаксис Automation:

BOOL DeleteCentreByPoint(double X, double Y);

Синтаксис COM:

HRESULT DeleteCentreByPoint(double X, double Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

X,Y	- координаты обозначения центра.

Интерфейс ICutLine

Справка системы КОМПАС: Команда Линия разреза

kompas.chm::/CM_CUTLINE.htm

Интерфейс линии разреза/сечения.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

ICutLine

Примечание:

1. Интерфейс позволяет получить и задать свойства объекта "линия разреза/сечения"
2. Получить интерфейс можно, используя свойство ICutLines::CutLine или метод ICutLines::Add интерфейса коллекции линий разреза/сечения.
3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

ICutLine – свойства

AdditionalText – Дополнительный текст линии разреза/сечения

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

```
AdditionText = Object.AdditionText          Получить свойство(* )
AdditionText = Object.GetAdditionText()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_AdditionText(                    Получить свойство
&AdditionText )
```

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на интерфейс текста IText для редактирования дополнительного текста линии разреза/сечения.

AdditionalTextPos – Размещение дополнительного текста

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - у первой стрелки,
FALSE       - у второй.
```

Синтаксис Automation:

```
AdditionTextPos = Object.AdditionTextPos    Получить свойство(* )
Object.AdditionTextPos = AdditionTextPos    Установить свойство (* )
AdditionTextPos =                           Получить свойство (**)
Object.GetAdditionTextPos()
Object.SetAdditionTextPos(                  Установить свойство (**)
AdditionTextPos )
```

Синтаксис COM:

```
Object.get_AdditionTextPos(                 Получить свойство
&AdditionTextPos )
Object.put_AdditionTextPos(                 Установить свойство
AdditionTextPos )
```

Примечание:

По умолчанию значение свойства равно TRUE.

ArrowPos – Расположение стрелок

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- слева по направлению ломаной,
FALSE	- справа по направлению ломаной.

Синтаксис Automation:

ArrowPos = Object.ArrowPos	Получить свойство(*)
Object.ArrowPos = ArrowPos	Установить свойство (*)
ArrowPos = Object.GetArrowPos()	Получить свойство (**)
Object.SetArrowPos(ArrowPos)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowPos(&ArrowPos)	Получить свойство
Object.put_ArrowPos(ArrowPos)	Установить свойство

ArrowType – Тип стрелки

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = Object.ArrowType	Получить свойство(*)
Object.ArrowType = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType()	Получить свойство (**)
Object.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство
Object.put_ArrowType(ArrowType)	Установить свойство

AutoSheet – Лист

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- включить опцию,
FALSE	- отключить опцию.

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание:

По умолчанию значение свойства равно FALSE.

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- автосортировка включена,
FALSE	- автосортировка выключена.

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание:

1. По умолчанию значение свойства равно TRUE.
2. При отключенной опции доступно редактирование основного текста линии разреза/сечения с помощью свойства ICutLine::Text.

AutoZone – Зона

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- включить опцию,
FALSE	- отключить опцию.

Синтаксис Automation:

AutoZone = Object.AutoZone	Получить свойство(*)
Object.AutoZone = AutoZone	Установить свойство (*)
AutoZone = Object.GetAutoZone()	Получить свойство (**)
Object.SetAutoZone(AutoZone)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoZone(&AutoZone)	Получить свойство
Object.put_AutoZone(AutoZone)	Установить свойство

Примечание:

По умолчанию значение свойства равно FALSE.

Points – Массив координат точек линии разреза/сечения в виде SAFEARRAY double – VT_ARRAY | VT_R8

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Points = Object.Points	Получить свойство(*)
Object.Points = Points	Установить свойство (*)
Points = Object.GetPoints()	Получить свойство (**)
Object.SetPoints(Points)	Установить свойство (**)

Синтаксис COM:

Object.get_Points(&Points)	Получить свойство
Object.put_Points(Points)	Установить свойство

Примечание:

Координаты точек располагаются в массиве в следующем порядке:

x0, y0 (начальная точка), x1, y1, ... x2, y2, ... (точки перегибов), xn, yn (конечная точка). Количество элементов массива должно быть четным.

Text – Текст обозначения линии разреза/сечения

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = Object.Text	Получить свойство(*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на интерфейс текста IText для редактирования основного текста линии разреза/сечения. Указатель на интерфейс IText вернется только в случае отключенной опции Автосортировка. Если автосортировка текста включена, свойство вернёт NULL.

X1 – Координата начального текста по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1()	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

Object.get_X1(&X1)	Получить свойство
Object.put_X1(X1)	Установить свойство

X2 – Координата конечного текста по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X2 = Object.X2	Получить свойство(*)
Object.X2 = X2	Установить свойство (*)

X2 = Object.GetX2()	Получить свойство (**)
Object.SetX2(X2)	Установить свойство (**)

Синтаксис COM:

Object.get_X2(&X2)	Получить свойство
Object.put_X2(X2)	Установить свойство

Y1 – Координата начального текста по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1()	Получить свойство (**)
Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство
Object.put_Y1(Y1)	Установить свойство

Y2 – Координата конечного текста по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Интерфейс IDrawingTable

Справка системы КОМПАС...

`KOMPAS.chm::/594_70_1_Tablicy_v_graficheskom.htm`

Интерфейс таблицы на чертеже.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IDrawingTable

ITable

Примечание:

1. Получить интерфейс можно, используя свойство интерфейса коллекции таблиц IDrawingTables::DrawingTable.
2. Посредством вызова метода IUnknown::QueryInterface интерфейс позволяет получить интерфейс сетки таблицы ITable.
3. Для того, чтобы сузить текст в ячейке таблицы, необходимо:

- снять признак запрета на редактирование ячейки с помощью свойства ICellFormat::ReadOnly и изменить текст,
- вызвать метод IDrawingObject::Update,
- установить признак запрета на редактирование ячейки,
- вызвать метод IDrawingObject::Update.

IDrawingTable – свойства

Angle – Угол наклона таблицы

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle( )	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол наклона таблицы.

FixedCellsSize – Зафиксировать габариты ячеек

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedCellsSize = Object.FixedCellsSize	Получить свойство(*)
Object.FixedCellsSize = FixedCellsSize	Установить свойство (*)
FixedCellsSize = Object.GetFixedCellsSize( )	Получить свойство (**)
Object.SetFixedCellsSize(FixedCellsSize)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedCellsSize(&FixedCellsSize)	Получить свойство
Object.put_FixedCellsSize(FixedCellsSize)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак фиксирования габаритов ячеек.

FixedColumnCount – Запретить изменять число столбцов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedColumnCount = Object.FixedColumnCount	Получить свойство(*)
Object.FixedColumnCount = FixedColumnCount	Установить свойство (*)
FixedColumnCount = Object.GetFixedColumnCount( )	Получить свойство (**)
Object.SetFixedColumnCount(FixedColumnCount)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedColumnCount(&FixedColumnCount)	Получить свойство
Object.put_FixedColumnCount(FixedColumnCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак запрета на изменение числа столбцов.

FixedRowCount – Запретить изменять число строк

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedRowCount = Object.FixedRowCount	Получить свойство(*)
Object.FixedRowCount = FixedRowCount	Установить свойство (*)
FixedRowCount = Object.GetFixedRowCount( )	Получить свойство (**)
Object.SetFixedRowCount(FixedRowCount)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedRowCount(&FixedRowCount)	Получить свойство
Object.put_FixedRowCount(FixedRowCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак запрета на изменение числа строк.

X – Координата точки привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX( )	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки привязки по X.

Y – Координата точки привязки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY( )	Получить свойство (**)

Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки привязки по Y.

IDrawingTable – методы

Save – Сохранить в файл

Интерфейс...

Синтаксис Automation:

BOOL Save(BSTR fileName);

Синтаксис COM:

HRESULT Save(BSTR fileName, BOOL * PVal);

Входные параметры:

fileName	- имя файла таблицы.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет сохранить таблицу в файл.

Интерфейс IDrawingText

Справка системы КОМПАС...

KOMPAS.chm::/CM_TEXT.htm

Интерфейс текста на чертеже.

Иерархия:

IKompasAPIObject

IDrawingObject

IDrawingText

IText

Примечание:

1. Интерфейс позволяет создать текст на чертеже. В общем случае это может быть многострочный, отформатированный по высоте и горизонтали текст.
2. У данного интерфейса можно получить дополнительный интерфейс IText посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).
3. Интерфейс можно получить у интерфейса коллекции текстов на чертеже, используя свойство IDrawingTexts::DrawingText.

IDrawingText – свойства

Allocation – Размещение текста относительно точки привязки

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/41411_1418_sozdanie_nadpisi.htm

Тип данных: из перечисления ksAllocationEnum

Синтаксис Automation:

Allocation = iObject.Allocation;	Получить свойство(*)
iObject.Allocation = Allocation;	Установить свойство (*)
Allocation = iObject.GetAllocation();	Получить свойство (**)
iObject.SetAllocation(Allocation);	Установить свойство (**)

Синтаксис COM:

iObject->get_Allocation (&Allocation);	Получить свойство
iObject->put_Allocation (Allocation);	Установить свойство

Примечание:

Свойство позволяет установить и получить тип размещения точки привязки текста.

Angle – Угол наклона текста

Интерфейс...

Тип данных: double, размерность - градусы

Синтаксис Automation:

```
Angle = iObject.Angle;          Получить свойство(* )
iObject.Angle = Angle;          Установить свойство (* )
Angle = iObject.GetAngle();     Получить свойство (**)
iObject.SetAngle( Angle );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle(&Angle);     Получить свойство
iObject->put_Angle( Angle );    Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол наклона текста.

Height – Высота блока форматирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Height = iObject.Height;        Получить свойство(* )
iObject.Height = Height;        Установить свойство (* )
Height =                        Получить свойство (**)
iObject.GetHeight();
iObject.SetHeight( Height );    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Height( &Height );     Получить свойство
iObject->put_Height( Height );      Установить свойство
```

Примечание:

Свойство позволяет установить и получить высоту блока форматирования.

HFormat – Признак горизонтального форматирования

Интерфейс...

Тип данных: из перечисления ksTextHorizontalFormatEnum

Синтаксис Automation:

HFormat = iObject.HFormat;	Получить свойство(*)
iObject.HFormat = HFormat;	Установить свойство (*)
HFormat = iObject.GetHFormat();	Получить свойство (**)
iObject.SetHFormat(HFormat);	Установить свойство (**)

Синтаксис COM:

iObject->get_HFormat(&HFormat);	Получить свойство
iObject->put_HFormat(HFormat);	Установить свойство

Примечание:

Свойство позволяет установить и получить признак горизонтального форматирования.

MirrorSymmetry – Отображать зеркально

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MirrorSymmetry = Object.MirrorSymmetry	Получить свойство(*)
Object.MirrorSymmetry = MirrorSymmetry	Установить свойство (*)
MirrorSymmetry = Object.GetMirrorSymmetry()	Получить свойство (**)
Object.SetMirrorSymmetry(MirrorSymmetry)	Установить свойство (**)

Синтаксис COM:

Object.get_MirrorSymmetry(&MirrorSymmetry)	Получить свойство
Object.put_MirrorSymmetry(MirrorSymmetry)	Установить свойство

VFormat – Признак вертикального форматирования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
VFormat =                       Получить свойство(* )
iObject.VFormat;
iObject.VFormat =               Установить свойство (* )
VFormat;
VFormat =                       Получить свойство (**)
iObject.GetVFormat();
iObject.SetVFormat(VForm        Установить свойство (**)
at);
```

Синтаксис COM:

```
iObject-                        Получить свойство
>get_VFormat(
&VFormat);
iObject-                        Установить свойство
>put_VFormat(
VFormat);
```

Значения свойства:

```
TRUE                    - изменение шага строк,
FALSE                   - нет форматирования.
```

Примечание:

Свойство позволяет установить и получить признак вертикального форматирования.

Width – Ширина блока форматирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = iObject.Width;          Получить свойство(* )
iObject.Width = Width;          Установить свойство (* )
Width =                         Получить свойство (**)
iObject.GetWidth();
iObject.SetWidth( Width );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Width(             Получить свойство
&Width );
iObject->put_Width( Width       Установить свойство
);
```

Примечание:

Свойство позволяет установить и получить ширину блока форматирования.

X – Координата точки привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X = iObject.X;              Получить свойство(* )
iObject.X = X;              Установить свойство (* )
X = iObject.GetX();         Получить свойство (**)
iObject.SetX( X );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X(&X);         Получить свойство
iObject->put_X( X );        Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату точки привязки по X.

Y – Координата точки привязки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y = iObject.Y;              Получить свойство(* )
iObject.Y = Y;              Установить свойство (* )
Y = iObject.GetY();         Получить свойство (**)
iObject.SetY(Y);            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y(&Y);         Получить свойство
iObject->put_Y(Y);          Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату точки привязки по Y.

Интерфейс IRough

Справка системы КОМПАС: Команда Шероховатость (графический документ)

kompas.chm::/CM_ROUGH.htm

Общие приемы построения

kompas.chm::/220_28_3_Sherokhovatostq.htm

Интерфейс обозначения шероховатости.

Иерархия:

IKompasAPIObject

IDrawingObject

IRough

IRoughParams

Описание:

Интерфейс позволяет получить и задать свойства обозначения шероховатости.

Интерфейс можно получить у коллекции обозначений шероховатости, используя свойство IRoughs::Rough или метод IRoughs::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейс IRoughParams является дополнительным. Его можно получить с помощью метода IUnknown::QueryInterface.

IRough – свойства

BaseObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject (&BaseObject)	Получить свойство
Object.put_BaseObject (BaseObject)	Установить свойство

ShelfX – Координата начала полки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = Object.ShelfX	Получить свойство(*)
Object.ShelfX = ShelfX	Установить свойство (*)
ShelfX = Object.GetShelfX()	Получить свойство (**)
Object.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfX(&ShelfX)	Получить свойство
Object.put_ShelfX(ShelfX)	Установить свойство

ShelfY – Координата начала полки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfY = Object.ShelfY	Получить свойство(*)
Object.ShelfY = ShelfY	Установить свойство (*)
ShelfY = Object.GetShelfY()	Получить свойство (**)
Object.SetShelfY(ShelfY)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfY(&ShelfY)	Получить свойство
Object.put_ShelfY(ShelfY)	Установить свойство

X0 – Координата начала выносной линии или положение знака по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X0 = Object.X0	Получить свойство(*)
Object.X0 = X0	Установить свойство (*)
X0 = Object.GetX0()	Получить свойство (**)
Object.SetX0(X0)	Установить свойство (**)

Синтаксис COM:

Object.get_X0(&X0)	Получить свойство
Object.put_X0(X0)	Установить свойство

BranchY0 – Координата начала выносной линии или положение знака по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchY0 = Object.BranchY0	Получить свойство(*)
Object.BranchY0 = BranchY0	Установить свойство (*)
BranchY0 = Object.GetBranchY0()	Получить свойство (**)
Object.SetBranchY0(BranchY0)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchY0(&BranchY0)	Получить свойство
Object.put_BranchY0(BranchY0)	Установить свойство

BranchArrowType – Тип значка на ответвлении

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

BranchArrowType = Object.BranchArrowType(Index)	Получить свойство(*)
Object.BranchArrowType(Index) = BranchArrowType	Установить свойство (*)
BranchArrowType = Object.GetBranchArrowType(Index)	Получить свойство (**)
Object.SetBranchArrowType(Index, BranchArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchArrowType(Index, &BranchArrowType)	Получить свойство

Object.put_BranchArrowType(Index, BranchArrowType)	Установить свойство

Входные параметры:

Index	- Индекс ответвления

BranchArrowInside – Расположение стрелки линии-выноски (TRUE- внутри, FALSE- снаружи)

Интерфейс...

Тип данных: BOOL

BranchArrowInside = Object.BranchArrowInside(Index)	Получить свойство(*)
Object.BranchArrowInside(Index) = BranchArrowInside)	Установить свойство (*)
BranchArrowInside = Object.GetBranchArrowInside(Index)	Получить свойство (**)
Object.SetBranchArrowInside(Index, BranchArrowInside)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchArrowInside(Index, &BranchArrowInside)	Получить свойство
Object.put_BranchArrowInside(Index, BranchArrowInside)	Установить свойство

Входные параметры:

Index	- Индекс ответвления

Интерфейс ITolerance

Справка системы КОМПАС: Команда Допуск формы (графический документ)

`kompas.chm::/CM_FORMTOLERANCE.htm`

Общие сведения о допуске формы...

`kompas.chm::/246_28_13_Dopusk_formy.htm`

Интерфейс обозначения допуска формы 2D.

Иерархия:

IKompasAPIObject

IDrawingObject

ITolerance

IToleranceParam

IBranchs

ITable

Примечание:

1. Получить интерфейс можно, используя свойство интерфейса коллекции ITolerances::Tolerance или метод ITolerances::Add.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительные интерфейсы:

- IToleranceParam - интерфейс параметров допуска формы,
- IBranchs - интерфейс для работы с ответвлениями,
- ITable - интерфейс таблицы.

3. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

ITolerance – свойства

ArrowType – Тип стрелки ответвления

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- стрелка,
FALSE	- треугольник.

Синтаксис Automation:

ArrowType = Object.ArrowType(Index)	Получить свойство(*)
Object.ArrowType(Index) = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType(Index)	Получить свойство (**)
Object.SetArrowType(Index, ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Index,	Получить свойство
Object.put_ArrowType(ArrowType)	Index,	Установить свойство

Входные параметры:

Ind ex	- индекс ответвления (long).	

Примечание:

Свойство позволяет устанавливать и получать тип стрелки ответвления.

BranchPos – Положение ножки

Интерфейс...

Тип данных: из перечисления ksTablePointEnum

Синтаксис Automation:

BranchPos = Object.BranchPos(Index)	Получить свойство(*)
Object.BranchPos(Index) = BranchPos	Установить свойство (*)
BranchPos = Object.GetBranchPos(Index)	Получить свойство (**)
Object.SetBranchPos(Index, BranchPos)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchPos(&BranchPos)	Index,	Получить свойство
Object.put_BranchPos(BranchPos)	Index,	Установить свойство

Входные параметры:

Index	- индекс ответвления (long).

Примечание:

Свойство позволяет устанавливать и получать тип положения ножки ответвления.

ToleranceArrowType – Тип стрелки ответвления

Интерфейс...

Тип данных: из перечисления ksToleranceArrowType

Синтаксис Automation:

ToleranceArrowType = Object.ToleranceArrowType(Index)	Получить свойство(*)
Object.ToleranceArrowType(Index) = ToleranceArrowType	Установить свойство (*)
ToleranceArrowType = Object.GetToleranceArrowType(Index)	Получить свойство (**)
Object.SetToleranceArrowType(Index, ToleranceArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceArrowType(Index, &ToleranceArrowType)	Получить свойство
Object.put_ToleranceArrowType(Index, ToleranceArrowType)	Установить свойство

Входные параметры:

Index	- индекс ответвления.

Интерфейс IViewPointer

Справка системы КОМПАС: Команда Стрелка взгляда

kompas.chm::/CM_VIEWPOINTER.htm

Интерфейс стрелки взгляда.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IViewPointer

Примечание:

Интерфейс можно получить, используя свойство коллекции стрелок взгляда IViewPointers::ViewPointer или IModelObjects::Item или метод IViewPointers::Add.

IViewPointer – свойства

AdditionalText – Дополнительный текст

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

AdditionalText = Object.AdditionalText	Получить свойство(*)
AdditionalText = Object.GetAdditionalText()	Получить свойство (**)

Синтаксис COM:

Object.get_AdditionalText(&AdditionalText)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс дополнительного текста обозначения направления взгляда.
2. Свойство доступно только для чтения.

ArrowType – Тип стрелки

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = Object.ArrowType	Получить свойство(*)
Object.ArrowType = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType( )	Получить свойство (**)
Object.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство
Object.put_ArrowType(ArrowType)	Установить свойство

AutoSheet – Автоматическое формирование обозначения листа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSheet = Object.AutoSheet	Получить свойство(*)
Object.AutoSheet = AutoSheet	Установить свойство (*)
AutoSheet = Object.GetAutoSheet( )	Получить свойство (**)
Object.SetAutoSheet(AutoSheet)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSheet(&AutoSheet)	Получить свойство
Object.put_AutoSheet(AutoSheet)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автоматического формирования обозначения листа.

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted( )	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автосортировки.

AutoZone – Автоматическое формирование обозначения зоны

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoZone = Object.AutoZone	Получить свойство(*)
Object.AutoZone = AutoZone	Установить свойство (*)
AutoZone = Object.GetAutoZone( )	Получить свойство (**)
Object.SetAutoZone(AutoZone)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoZone(&AutoZone)	Получить свойство
Object.put_AutoZone(AutoZone)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автоматического формирования обозначения зоны.

Text – Обозначение направления взгляда

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = Object.Text	Получить свойство(*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текста обозначения направления взгляда.
2. Свойство доступно только для чтения.

TextX – Координата точки привязки текста по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextX = Object.TextX	Получить свойство(*)
Object.TextX = TextX	Установить свойство (*)
TextX = Object.GetTextX( )	Получить свойство (**)
Object.SetTextX(TextX)	Установить свойство (**)

Синтаксис COM:

Object.get_TextX(&TextX)	Получить свойство
Object.put_TextX(TextX)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки привязки текста по оси X.

TextY – Координата точки привязки текста по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextY = Object.TextY	Получить свойство(*)
Object.TextY = TextY	Установить свойство (*)
TextY = Object.GetTextY( )	Получить свойство (**)
Object.SetTextY(TextY)	Установить свойство (**)

Синтаксис COM:

Object.get_TextY(&TextY)	Получить свойство
Object.put_TextY(TextY)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату точки привязки текста по оси Y.

X1 – Координата начальной точки стрелки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1( )	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

Object.get_X1(&X1)	Получить свойство
Object.put_X1(X1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату начальной точки стрелки по оси X.

X2 – Координата конечной точки стрелки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X2 = Object.X2	Получить свойство(*)
Object.X2 = X2	Установить свойство (*)
X2 = Object.GetX2( )	Получить свойство (**)
Object.SetX2(X2)	Установить свойство (**)

Синтаксис COM:

Object.get_X2(&X2)	Получить свойство
Object.put_X2(X2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки стрелки по оси X.

Y1 – Координата начальной точки стрелки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1( )	Получить свойство (**)
Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство
Object.put_Y1(Y1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату начальной точки стрелки по оси Y.

Y2 – Координата конечной точки стрелки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2( )	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату конечной точки стрелки по оси Y.

Интерфейс IWaveLine

Справка системы КОМПАС...

`KOMPAS.chm::/CM_WAVELINE.htm`

Интерфейс волнистой линии.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IWaveLine

Описание:

Интерфейс позволяет задавать и получать параметры волнистой линии.

Примечание:

1. Интерфейс можно получить у коллекции волнистых линий, используя свойство IWaveLines::WaveLine или метод IWaveLines::Add.
2. После задания параметров объекта требуется вызвать метод IDrawingObject::Update.

IWaveLine – свойства

Angle – Угол между линией и осью OX текущей СК

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол между волнистой линией и осью OX текущей СК.

AutoWavesAmplitude – Использовать задание амплитуды волны из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoWavesAmplitude = Object.AutoWavesAmplitude	=	Получить свойство(*)
Object.AutoWavesAmplitude = AutoWavesAmplitude	=	Установить свойство (*)

```
AutoWavesAmplitude =                    Получить свойство (**)
Object.GetAutoWavesAmplitude(
)
Object.SetAutoWavesAmplitude(           Установить свойство (**)
AutoWavesAmplitude )
```

Синтаксис COM:

```
Object.get_AutoWavesAmplitude           Получить свойство
( &AutoWavesAmplitude )
Object.put_AutoWavesAmplitude           Установить свойство
(  AutoWavesAmplitude )
```

Примечание:

Свойство позволяет устанавливать и получать признак задания амплитуды волны из настроек документа.

Direction – Направление

Интерфейс...

Тип данных: BOOL

Значение свойства:

```
TRUE        - направление 1,
FALSE       - направление 2.
```

Синтаксис Automation:

```
Direction = Object.Direction            Получить свойство(* )
Object.Direction = Direction            Установить свойство (* )
Direction = Object.GetDirection()       Получить свойство (**)
Object.SetDirection( Direction )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction(  &Direction       Получить свойство
)
Object.put_Direction(  Direction )      Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать направление.

HalfWavesCount – Количество полуволн

Интерфейс...

Тип данных: long

Синтаксис Automation:

HalfWavesCount = Object.HalfWavesCount	=	Получить свойство(*)
Object.HalfWavesCount = HalfWavesCount	=	Установить свойство (*)
HalfWavesCount = Object.GetHalfWavesCount()	=	Получить свойство (**)
Object.SetHalfWavesCount(HalfWavesCount)		Установить свойство (**)

Синтаксис COM:

Object.get_HalfWavesCount(&HalfWavesCount)	Получить свойство
Object.put_HalfWavesCount(HalfWavesCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество полуволн.

Length – Длина

Интерфейс...

Тип данных: double

Синтаксис Automation:

Length = Object.Length	Получить свойство(*)
Object.Length = Length	Установить свойство (*)
Length = Object.GetLength()	Получить свойство (**)
Object.SetLength(Length)	Установить свойство (**)

Синтаксис COM:

Object.get_Length(&Length)	Получить свойство
Object.put_Length(Length)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину волнистой линии.

Style – Стиль линии

Интерфейс...

Тип данных: long.

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)

Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать стиль линии.
2. Системные стили линий берутся из перечисления ksCurveStyleEnum.

WaveLength – Длина волны

Интерфейс...

Тип данных: double

Синтаксис Automation:

WaveLength Object.WaveLength	=	Получить свойство(*)
Object.WaveLength WaveLength	=	Установить свойство (*)
WaveLength Object.GetWaveLength()	=	Получить свойство (**)
Object.SetWaveLength(WaveLength)		Установить свойство (**)

Синтаксис COM:

Object.get_WaveLength(&WaveLength)	Получить свойство
Object.put_WaveLength(WaveLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину волны.

WavesAmplitude – Амплитуда волн

Интерфейс...

Тип данных: double

Синтаксис Automation:

WavesAmplitude Object.WavesAmplitude	=	Получить свойство(*)

WavesAmplitude = Object.GetWavesAmplitude ()		Получить свойство (**)

Синтаксис COM:

Object.get_WavesAmplitude (&WavesAmplitude)		Получить свойство

Примечание:

1. Свойство позволяет получать значение амплитуды волн.
2. Свойство доступно только для чтения.

WavesAmplitudeRepresentation – Получить представление амплитуды волн

Интерфейс...

Тип данных: BOOL

Значение свойства:

TRUE	- амплитуда задана в абсолютных значениях,
FALSE	- в процентах.

Синтаксис Automation:

WavesAmplitudeRepresentation = Object.WavesAmplitudeRepresentation	Получить свойство(*)
WavesAmplitudeRepresentation = Object.GetWavesAmplitudeRepresentation()	Получить свойство (**)

Синтаксис COM:

Object.get_WavesAmplitudeRepresentation(&WavesAmplitudeRepresentation)	Получить свойство

Примечание:

1. Свойство позволяет получать представление амплитуды волн.
2. Свойство доступно только для чтения.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1()	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

Object.get_X1(&X1)	Получить свойство
Object.put_X1(X1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату первой точки по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X2 = Object.X2	Получить свойство(*)
Object.X2 = X2	Установить свойство (*)
X2 = Object.GetX2()	Получить свойство (**)
Object.SetX2(X2)	Установить свойство (**)

Синтаксис COM:

Object.get_X2(&X2)	Получить свойство
Object.put_X2(X2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату второй точки по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1()	Получить свойство (**)

Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство
Object.put_Y1(Y1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату первой точки по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство
Object.put_Y2(Y2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату второй точки по оси Y.

IWaveLine – методы

SetWavesAmplitude – Задать представление амплитуды волн

Интерфейс...

Синтаксис Automation:

BOOL SetWavesAmplitude(BOOL Representation,
double NewVal)

Синтаксис COM:

HRESULT SetWavesAmplitude(BOOL Representation,
double NewVal,
BOOL * Result);

Входные параметры:

Representation	- TRUE - амплитуда задана в абсолютных значениях, FALSE - в процентах,
NewVal	- значение амплитуды.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет задать представление амплитуды волн.

Размеры

Коллекции размеров

Интерфейс AngleDimensions

Справка системы КОМПАС: Команда Угловой размер (графический документ)

`kompas.chm::/CM_DIMA.htm`

Общие сведения

`kompas.chm::/204_26_1_Prostoj_uglovoj_razmer.htm`

Интерфейс коллекции угловых размеров.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IAngleDimensions

Описание:

Интерфейс позволяет получать и создавать угловые размеры.

Примечание:

1. Коллекция содержит угловые и угловые с обрывом размеры.
2. Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::AngleDimensions.

IAngleDimensions – свойства

AngleDimension – Угловой размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс углового размера IAngleDimension

Синтаксис Automation:

AngleDimension iObject.AngleDimension(Index);	=	Получить свойство(*)
AngleDimension iObject.GetAngleDimension(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AngleDimension(Index, &AngleDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IAngleDimensions – методы

Add – Добавить угловой размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(DrawingObjectTypeEnum DimType);

Синтаксис COM:

HRESULT Add(DrawingObjectTypeEnum DimType, [out, retval] IAngleDimension** Result);

Входной параметр:

DimType	- тип объекта.

Возвращаемое значение:

- указатель на интерфейс углового размера IAngleDimension.

Примечание:

Для добавления углового размера в качестве параметра DimType нужно передать константу ksDrADimension, для углового размера с обрывом нужно передать константу ksDrABreakDimension.

Интерфейс IArcDimensions

Справка системы КОМПАС: Команда Размер дуги окружности

kompas.chm::/CM_ARC_DIM.htm

Интерфейс коллекции размеров дуг окружностей.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IArcDimensions

Описание:

Интерфейс позволяет получать и создавать размеры дуг окружностей.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::ArcDimensions.

IArcDimensions – свойства

ArcDimension – Размер дуги окружности, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс размера дуги окружности IArcDimension

Синтаксис Automation:

ArcDimension = iObject.ArcDimension(Index);	Получить свойство(*)
ArcDimension = iObject.GetArcDimension(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_ArcDimension(Index, &ArcDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IArcDimensions – методы

Add – Добавить размер дуги окружности

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IArcDimension** Result);

Возвращаемое значение:

	- указатель на интерфейс размера дуги окружности IArcDimension.

Интерфейс IBreakLineDimensions

Справка системы КОМПАС: Команда Линейный размер с обрывом (графический документ)

`kompas.chm::/CM_CUT_DIML.htm`

Основные сведения

`kompas.chm::/193_24_2_Linejnyj_razmer_s_obry.htm`

Интерфейс коллекции линейных размеров с обрывом.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBreakLineDimensions

Описание:

Интерфейс позволяет получать и создавать линейные размеры с обрывом.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::BreakLineDimensions.

IBreakLineDimensions – свойства

BreakLineDimension – Линейный размер с обрывом, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс линейного размера с обрывом IBreakLineDimension

Синтаксис Automation:

LineDimension iObject.BreakLineDimension(Index);	=	Получить свойство(*)
LineDimension iObject.GetBreakLineDimension(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_BreakLineDimension(Index, &LineDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IBreakLineDimensions – методы

Add – Добавить линейный размер с обрывом в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IBreakLineDimension** Result);

Возвращаемое значение:

- указатель на интерфейс линейного размера с обрывом IBreakLineDimension.

Интерфейс IBreakRadialDimensions

Справка системы КОМПАС: Команда Радиальный размер с изломом

kompas.chm::/CM_DIMR_WITH_BREAK.htm

Основные сведения

kompas.chm::/202_25_3_Radialqnyj_razmer_s_iz.htm

Интерфейс коллекции радиальных размеров с изломом.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBreakRadialDimensions

Описание:

Интерфейс позволяет получать и создавать радиальные размеры с изломом.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::BreakRadialDimensions.

IBreakRadialDimensions – свойства

RadialDimension – Радиальный размер с изломом, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс радиального размера с изломом IBreakRadialDimension

Синтаксис Automation:

BreakRadialDimension = iObject.BreakRadialDimension(Index);	Получить свойство(*)
BreakRadialDimension = iObject.GetBreakRadialDimensio n(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_BreakRadialDimension(Index, &BreakRadialDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс узла.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IBreakRadialDimensions – методы

Add – Добавить радиальный размер с изломом в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IBreakRadialDimension** Result);

Возвращаемое значение:

	- указатель на интерфейс радиального размера с изломом IBreakRadialDimension.

Интерфейс IDiametralDimensions

Справка системы КОМПАС: Команда Диаметральный размер

`kompas.chm::/CM_DIMD.htm`

Основные сведения

`kompas.chm::/201_25_1_Diametralqnyj_razmer.htm`

Интерфейс коллекции диаметральных размеров.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IDiametralDimensions

Описание:

Интерфейс позволяет получать и создавать диаметральные размеры.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::DiametralDimensions.

IDiametralDimensions – свойства

DiametralDimension – Диаметральный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс диаметрального размера IDiametralDimension

Синтаксис Automation:

DiametralDimension = iObject.DiametralDimension(Index);		Получить свойство(*)
DiametralDimension = iObject.GetDiametralDimension(Index);		Получить свойство (**)

Синтаксис COM:

iObject->get_DiametralDimension(Index, &DiametralDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс узла.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IDiametralDimensions – методы

Add – Добавить диаметральный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IDiametralDimension** Result);

Возвращаемое значение:

- указатель на интерфейс диаметрального размера IDiametralDimension.

Интерфейс IHeightDimensions

Справка системы КОМПАС: Команда Размер высоты

kompas.chm::/CM_ORDINATE_DIM.htm

Общие сведения о простановке размера

kompas.chm::/198_24_8_Razmer_vysoty.htm

Диалог ввода размерной надписи размера высоты

kompas.chm::/DLG_DIM_TEXT_ORD_INPUT.htm

Интерфейс коллекции размеров высоты.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IHeightDimensions

Описание:

Интерфейс позволяет получать и создавать размеры высоты.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::HeightDimensions.

IHeightDimensions – свойства

HeightDimension – Размер высоты, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс размера высоты IHeightDimension

Синтаксис Automation:

HeightDimension = iObject.HeightDimension(Index);	Получить свойство(*)
HeightDimension = iObject.GetHeightDimension(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_HeightDimension(Index, &HeightDimension)		Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IHeightDimensions – методы

Add – Добавить размер высоты в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IHeightDimension** Result);

Возвращаемое значение:

	- указатель на интерфейс размера высоты IHeightDimension.

Интерфейс ILineDimensions

Справка системы КОМПАС: Команда Линейный размер (графический документ)

`kompas.chm::/CM_DIML.htm`

Основные сведения

`kompas.chm::/191_24_1_Prostoj_linejnyj_razme.htm`

Интерфейс коллекции линейных размеров.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ILineDimensions

Описание:

Интерфейс позволяет получать и создавать линейные размеры.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::LineDimensions.

ILineDimensions – свойства

LineDimension – Линейный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс линейного размера ILineDimension

Синтаксис Automation:

LineDimension = iObject.LineDimension(Index);	Получить свойство(*)
LineDimension = iObject.GetLineDimension(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_LineDimension(Index, &LineDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ILineDimensions – методы

Add – Добавить линейный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add( );

Синтаксис COM:

HRESULT Add([out, retval] ILineDimension** Result);

Возвращаемое значение:

- указатель на интерфейс линейного размера ILineDimension.

Интерфейс IRadialDimensions

Справка системы КОМПАС: Команда Радиальный размер

kompas.chm::/CM_DIMR.htm

Основные сведения

kompas.chm::/201_25_2_Prostoj_radialqnyj_raz.htm

Интерфейс коллекции радиальных размеров.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IRadialDimensions

Описание:

Интерфейс позволяет получать и создавать радиальные размеры.

Примечание:

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols2DContainer::RadialDimensions.

IRadialDimensions – свойства

RadialDimension – Радиальный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс радиального размера IRadialDimension

Синтаксис Automation:

RadialDimension = iObject.RadialDimension(Index);	=	Получить свойство(*)
RadialDimension = iObject.GetRadialDimension(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_RadialDimension(Index, &RadialDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IRadialDimensions – методы

Add – Добавить радиальный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IRadialDimension** Result);

Возвращаемое значение:

	- указатель на интерфейс радиального размера IRadialDimension.

Интерфейс IAngleDimension

Справка системы КОМПАС: Команда Угловой размер (графический документ)

`kompas.chm::/CM_DIMA.htm`

Команда Угловой размер с общей размерной линией

`kompas.chm::/CM_COMMON_LINE_DIMA.htm`

Интерфейс параметров углового размера.

Иерархия:

IKompasAPIObject

IDrawingObject

IAngleDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства углового размера.

Интерфейс можно получить у коллекции угловых размеров, используя свойство IAngleDimensions::AngleDimension или метод IAngleDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Если при создании размера задать опорные объекты IAngleDimension::BaseObject1, IAngleDimension::BaseObject2, то координаты центра IAngleDimension::Xc, IAngleDimension::Yc, радиус IAngleDimension::Radius, IAngleDimension::X1, IAngleDimension::Y1, IAngleDimension::X2, IAngleDimension::Y2, задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве базовых объектов могут использоваться отрезки. Для углового размера второй опорный объект задает направление второй выносной линии. Для углового размера с обрывом второй опорный объект задает ось симметрии (биссектрису угла).

Свойство IDimensionText::NominalValue возвращает угол.

IAngleDimension – свойства

Angle1 – Угол наклона первой размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle1 = Object.Angle1	Получить свойство(*)
Object.Angle1 = Angle1	Установить свойство (*)
Angle1 = Object.GetAngle1()	Получить свойство (**)
Object.SetAngle1(Angle1)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle1(&Angle1)	Получить свойство(*)
Object.put_Angle1(Angle1)	Установить свойство (*)

Angle2 – Угол наклона второй размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство(*)
Object.Angle2 = Angle2	Установить свойство (*)

```
Angle2 = Object.GetAngle2()        Получить свойство (**)
Object.SetAngle2( Angle2 )         Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle2( &Angle2 )       Получить свойство(* )
Object.put_Angle2( Angle2 )        Установить свойство (* )
```

BaseObject1 – Первый опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

```
BaseObject1 = Object.BaseObject1   Получить свойство(* )
Object.BaseObject1 = BaseObject1   Установить свойство (* )
BaseObject1 = Object.GetBaseObject1()   Получить свойство (**)
Object.SetBaseObject1( BaseObject1 )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BaseObject1( &BaseObject1 )   Получить свойство(* )
Object.put_BaseObject1( BaseObject1 )    Установить свойство (* )
```

Примечание:

Если при создании размера задать опорные объекты IAngleDimension::BaseObject1, IAngleDimension::BaseObject2, то координаты центра IAngleDimension::Xc, IAngleDimension::Yc, радиус IAngleDimension::Radius, IAngleDimension::X1, IAngleDimension::Y1, IAngleDimension::X2, IAngleDimension::Y2, задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве базовых объектов могут использоваться отрезки.

Чтобы разорвать связь с объектом, нужно установить значение свойства равным NULL.

BaseObject2 – Второй опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

```
BaseObject2 = Object.BaseObject2   Получить свойство(* )
Object.BaseObject2 = BaseObject2   Установить свойство (* )
```

BaseObject2 = Object.GetBaseObject2()	Получить свойство (**)
Object.SetBaseObject2(BaseObject2)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject2(&BaseObject2)	Получить свойство(*)
Object.put_BaseObject2(BaseObject2)	Установить свойство (*)

Примечание:

Если при создании размера задать опорные объекты IAngleDimension::BaseObject1, IAngleDimension::BaseObject2, то координаты центра IAngleDimension::Xc, IAngleDimension::Yc, радиус IAngleDimension::Radius, IAngleDimension::X1, IAngleDimension::Y1, IAngleDimension::X2, IAngleDimension::Y2, задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве базовых объектов могут использоваться отрезки.

Чтобы разорвать связь с объектом, нужно установить значение свойства равным NULL.

DimensionType – Тип углового размера

Интерфейс...

Тип данных: из перечисления ksAngleDimTypeEnum

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство(*)
Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство(*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

Direction – Направление размерной дуги

Интерфейс...

Тип данных: BOOL

Значения свойства:

true	- по часовой стрелке;
false	- против часовой стрелки.

Синтаксис Automation:

Direction = Object.Direction	Получить свойство(*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство(*)
Object.put_Direction(Direction)	Установить свойство (*)

Radius – Радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство(*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство(*)
Object.put_Radius(Radius)	Установить свойство (*)

ShelfX – Точка начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = Object.ShelfX	Получить свойство(*)
Object.ShelfX = ShelfX	Установить свойство (*)
ShelfX = Object.GetShelfX()	Получить свойство (**)
Object.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство(*)
Object.put_Y2(Y2)	Установить свойство (*)

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

ShelfY – Точка начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfY = Object.ShelfY	Получить свойство(*)
Object.ShelfY = ShelfY	Установить свойство (*)
ShelfY = Object.GetShelfY()	Получить свойство (**)
Object.SetShelfY(ShelfY)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfY(&ShelfY)	Получить свойство(*)
Object.put_ShelfY(ShelfY)	Установить свойство (*)

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)

```
Object.SetXc( Xc )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )              Получить свойство(* )
Object.put_Xc(  Xc )              Установить свойство (* )
```

X1 – Координата точки выхода первой выносной линии по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X1                    Получить свойство(* )
Object.X1 = X1                    Установить свойство (* )
X1 = Object.GetX1()               Получить свойство (**)
Object.SetX1( X1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )              Получить свойство(* )
Object.put_X1(  X1 )              Установить свойство (* )
```

X2 – Координата точки выхода первой выносной линии по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X2                    Получить свойство(* )
Object.X2 = X2                    Установить свойство (* )
X2 = Object.GetX2()               Получить свойство (**)
Object.SetX2( X2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )              Получить свойство(* )
Object.put_X2(  X2 )              Установить свойство (* )
```

X3 – Начало ножки или точка на дуге задающая положение текста по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = Object.X3                  Получить свойство(* )
Object.X3 = X3                  Установить свойство (* )
X3 = Object.GetX3()             Получить свойство (**)
Object.SetX3( X3 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X3( &X3 )            Получить свойство(* )
Object.put_X3(  X3 )            Установить свойство (* )
```

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                  Получить свойство(* )
Object.Yc = Yc                  Установить свойство (* )
Yc = Object.GetYc()             Получить свойство (**)
Object.SetYc( Yc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )            Получить свойство(* )
Object.put_Yc(  Yc )            Установить свойство (* )
```

Y1 – Координата точки выхода первой выносной линии по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1                  Получить свойство(* )
Object.Y1 = Y1                  Установить свойство (* )
Y1 = Object.GetY1()             Получить свойство (**)
Object.SetY1( Y1 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )            Получить свойство(* )
Object.put_Y1(  Y1 )            Установить свойство (* )
```

Y2 – Координата точки выхода первой выносной линии по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство(*)
Object.put_Y2(Y2)	Установить свойство (*)

Y3 – Начало ножки или точка на дуге, задающая положение текста по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y3 = Object.Y3	Получить свойство(*)
Object.Y3 = Y2	Установить свойство (*)
Y3 = Object.Get3()	Получить свойство (**)
Object.SetY3(Y3)	Установить свойство (**)

Синтаксис COM:

Object.get_Y3(&Y3)	Получить свойство(*)
Object.put_Y3(Y3)	Установить свойство (*)

Интерфейс IBreakAngleDimension

Справка системы КОМПАС: Команда Угловой размер с обрывом...

`kompas.chm::/CM_CUT_DIMA.htm`

Интерфейс параметров углового размера с обрывом.

Иерархия:

IKompasAPIObject

IDrawingObject

IAngleDimension

IBreakAngleDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства углового размера с обрывом.

Интерфейс можно получить у коллекции угловых размеров, используя свойство IAngleDimensions::AngleDimension или метод IAngleDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Интерфейс IArcDimension

Справка системы КОМПАС: Команда Размер дуги окружности

`kompas.chm::/CM_ARC_DIM.htm`

Интерфейс параметров размера дуги окружности.

Иерархия:

IKompasAPIObject

IDrawingObject

IArcDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства размера дуги окружности.

Интерфейс можно получить у коллекции размеров дуг окружностей, используя свойство IArcDimensions::IArcDimension или метод IArcDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

1. Свойство IDimensionText::NominalValue возвращает длину дуги.
2. Если при создании размера задать опорный объект IArcDimension::BaseObject, то значения свойств IArcDimension::Xc, IArcDimension::Yc, IArcDimension::X1, IArcDimension::Y1, IArcDimension::X2, IArcDimension::Y2, IArcDimension::Direction задавать не нужно. Данные свойства будут получены с исходных объектов. В качестве базового объекта может использоваться дуга окружности.

IArcDimension – свойства

BaseObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство(*)
Object.put_BaseObject(BaseObject)	Установить свойство (*)

Примечание:

Если при создании размера задать опорный объект IArcDimension::BaseObject, то значения свойств IArcDimension::Xc, IArcDimension::Yc, IArcDimension::X1, IArcDimension::Y1, IArcDimension::X2, IArcDimension::Y2, IArcDimension::Direction задавать не нужно. Данные свойства будут получены с исходных объектов. В качестве базового объекта может использоваться дуга окружности.

Чтобы разорвать связь с объектом, нужно установить значение свойства, равное NULL.

DimensionType – Тип размера (Выносные линии от центра\Параллельные выносные линии)

Интерфейс...

Тип данных: BOOL

Значения свойства:

true	- выносные линии от центра;
false	- параллельные выносные линии.

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство(*)

Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство(*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

Примечание:

Свойство позволяет задать направление выносных линий – от центра или параллельные выносные линии, проведенному в середину дуги. Если угол раствора дуги больше 180 градусов, возможно создание размера только с выносными линиями от центра.

Direction – Направление размерной дуги

Интерфейс...

Тип данных: BOOL

Значения свойства:

true	- по часовой стрелке;
false	- против часовой стрелки.

Синтаксис Automation:

Direction = Object.Direction	Получить свойство(*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство(*)
Object.put_Direction(Direction)	Установить свойство (*)

ShelfX – Значение координаты положения точки начала полки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfX = Object.ShelfX              Получить свойство(* )
Object.ShelfX = ShelfX              Установить свойство (* )
ShelfX = Object.GetShelfX()         Получить свойство (**)
Object.SetShelfX( ShelfX )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfX( &ShelfX )        Получить свойство(* )
Object.put_ShelfX( ShelfX )         Установить свойство (* )
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

ShelfY – Значение координаты положения точки начала полки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfY = Object.ShelfY              Получить свойство(* )
Object.ShelfY = ShelfY              Установить свойство (* )
ShelfY = Object.GetShelfY()         Получить свойство (**)
Object.SetShelfY( ShelfY )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfY( &ShelfY )        Получить свойство(* )
Object.put_ShelfY(  ShelfY )        Установить свойство (* )
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

TextPointer – Указатель от текста к дуге

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
true                                - отрисовать указатель;
false                               - не отрисовывать указатель.
```

Синтаксис Automation:

TextPointer = Object.TextPointer	Получить свойство(*)
Object.TextPointer = TextPointer	Установить свойство (*)
TextPointer = Object.GetTextPointer()	Получить свойство (**)
Object.SetTextPointer(TextPointer)	Установить свойство (**)

Синтаксис COM:

Object.get_TextPointer(&TextPointer)	Получить свойство(*)
Object.put_TextPointer(TextPointer)	Установить свойство (*)

Примечание:

Свойство позволяет включить соединить указателем дугу и текст, относящегося к ней размера. Это может потребоваться, например, при простановке размеров концентрических дуг с одинаковым раствором и начальным углом.

Xc – Значение координаты центра по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)
Object.Xc = Xc	Установить свойство (*)
Xc = Object.GetXc()	Получить свойство (**)
Object.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

Object.get_Xc(&Xc)	Получить свойство(*)
Object.put_Xc(Xc)	Установить свойство (*)

X1 – Значение координаты первой точки дуги по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1()	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

```
Object.get_X1( &X1 )          Получить свойство(* )
Object.put_X1(  X1 )          Установить свойство (* )
```

X2 – Значение координаты второй точки дуги по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                Получить свойство(* )
Object.X2 = X2                Установить свойство (* )
X2 = Object.GetX2()           Получить свойство (**)
Object.SetX2( X2 )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )          Получить свойство(* )
Object.put_X2(  X2 )          Установить свойство (* )
```

X3 – Значение координаты положения размерной линии и надписи по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = Object.X3                Получить свойство(* )
Object.X3 = X3                Установить свойство (* )
X2 = Object.GetX2()           Получить свойство (**)
Object.SetX3( X3 )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X3( &X3 )          Получить свойство(* )
Object.put_X3(  X3 )          Установить свойство (* )
```

Yc – Значение координаты центра по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                Получить свойство(* )
Object.Yc = Yc                Установить свойство (* )
```

```
Yc = Object.GetYc()          Получить свойство (**)
Object.SetYc( Yc )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )         Получить свойство(* )
Object.put_Yc(  Yc )         Установить свойство (* )
```

Y1 – Значение координаты первой точки дуги по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1               Получить свойство(* )
Object.Y1 = Y1               Установить свойство (* )
Y1 = Object.GetY1()          Получить свойство (**)
Object.SetY1( Y1 )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )         Получить свойство(* )
Object.put_Y1(  Y1 )         Установить свойство (* )
```

Y2 – Значение координаты второй точки дуги по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = Object.Y2               Получить свойство(* )
Object.Y2 = Y2               Установить свойство (* )
Y2 = Object.GetY2()          Получить свойство (**)
Object.SetY2( Y2 )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y2( &Y2 )         Получить свойство(* )
Object.put_Y2(  Y2 )         Установить свойство (* )
```

Y3 – Значение координаты положения размерной линии и надписи по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y3 = Object.Y3	Получить свойство(*)
Object.Y3 = Y3	Установить свойство (*)
Y3 = Object.GetY3()	Получить свойство (**)
Object.SetY3(Y3)	Установить свойство (**)

Синтаксис COM:

Object.get_Y3(&Y3)	Получить свойство(*)
Object.put_Y3(Y3)	Установить свойство (*)

Интерфейс IBreakLineDimension

Справка системы КОМПАС: Команда Линейный размер с обрывом

`kompas.chm::/CM_CUT_DIML.htm`

Интерфейс параметров линейного размера с обрывом.

Иерархия:

IKompasAPIObject

IDrawingObject

IBreakLineDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства линейного размера с обрывом.

Интерфейс можно получить у коллекции линейных размеров с обрывом, используя свойство IBreakLineDimensions::BreakLineDimension или метод IBreakLineDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

IBreakLineDimension – свойства

BaseObject – Опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)

BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство(*)
Object.put_BaseObject(BaseObject)	Установить свойство (*)

Примечание:

1. Если при создании размера задать опорный объект, то координаты центра IBreakLineDimension::X1, IBreakLineDimension::Y1, IBreakLineDimension::X2, IBreakLineDimension::Y2 задавать не нужно. Данные свойства будут получены с исходного объекта.

   В качестве опорного объекта может использоваться отрезок.

2. Чтобы разорвать связь с объектом, нужно установить значение свойства BaseObject, равное NULL.

ShelfX – Координата X точки начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = Object.ShelfX	Получить свойство(*)
Object.ShelfX = ShelfX	Установить свойство (*)
ShelfX = Object.GetShelfX()	Получить свойство (**)
Object.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfX(&ShelfX)	Получить свойство(*)
Object.put_ShelfX(ShelfX)	Установить свойство (*)

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

ShelfY – Координата Y точки начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfY = Object.ShelfY              Получить свойство(* )
Object.ShelfY = ShelfY              Установить свойство (* )
ShelfY = Object.GetShelfY()         Получить свойство (**)
Object.SetShelfY( ShelfY )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfY( &ShelfY )        Получить свойство(* )
Object.put_ShelfY(  ShelfY )        Установить свойство (* )
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

X1 – Координата X первой точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X1                      Получить свойство(* )
Object.X1 = X1                      Установить свойство (* )
X1 = Object.GetX1()                 Получить свойство (**)
Object.SetX1( X1 )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )                Получить свойство(* )
Object.put_X1(  X1 )                Установить свойство (* )
```

X2 – Координата X второй точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                      Получить свойство(* )
Object.X2 = X2                      Установить свойство (* )
X2 = Object.GetX2()                 Получить свойство (**)
Object.SetX2( X2 )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )                Получить свойство(* )
Object.put_X2(  X2 )                Установить свойство (* )
```

X3 – Координата X положения размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = Object.X3                  Получить свойство(* )
Object.X3 = X3                  Установить свойство (* )
X3 = Object.GetX3()             Получить свойство (**)
Object.SetX3( X3 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X3( &X3 )            Получить свойство(* )
Object.put_X3(  X3 )            Установить свойство (* )
```

Y1 – Координата Y первой точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1                  Получить свойство(* )
Object.Y1 = Y1                  Установить свойство (* )
Y1 = Object.GetY1()             Получить свойство (**)
Object.SetY1( Y1 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )            Получить свойство(* )
Object.put_Y1(  Y1 )            Установить свойство (* )
```

Y2 – Координата Y второй точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = Object.Y2                  Получить свойство(* )
Object.Y2 = Y2                  Установить свойство (* )
Y2 = Object.GetY2()             Получить свойство (**)
Object.SetY2( Y2 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y2( &Y2 )            Получить свойство(* )
Object.put_Y2(  Y2 )            Установить свойство (* )
```

Y3 – Координата Y положения размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y3 = Object.Y3	Получить свойство(*)
Object.Y3 = Y3	Установить свойство (*)
Y3 = Object.GetY3()	Получить свойство (**)
Object.SetY3(Y3)	Установить свойство (**)

Синтаксис COM:

Object.get_Y3(&Y3)	Получить свойство(*)
Object.put_Y3(Y3)	Установить свойство (*)

Интерфейс IDiametralDimension

Справка системы КОМПАС: Команда Диаметральный размер

`kompas.chm::/CM_DIMD.htm`

Интерфейс параметров диаметрального размера.

Иерархия:

IKompasAPIObject

IDrawingObject

IDiametralDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства диаметрального размера.

Интерфейс можно получить у коллекции диаметральных размеров, используя свойство IDiametralDimensions::DiametralDimension или метод IDiametralDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Если при создании размера задать опорный объект IDiametralDimension::BaseObject, то координаты центра IDiametralDimension::Xc, IDiametralDimension::Yc и радиус IDiametralDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта. Свойство IDimensionText::NominalValue возвращает диаметр.

В качестве опорного объекта может быть использована окружность или дуга окружности.

IDiametralDimension – свойства

Angle – Угол наклона размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство(*)
Object.put_Angle(Angle)	Установить свойство (*)

BaseObject – Опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство(*)
Object.put_BaseObject(BaseObject)	Установить свойство (*)

Примечание:

Если при создании размера задать опорный объект IDiametralDimension::BaseObject, то координаты центра IDiametralDimension::Xc, IDiametralDimension::Yc и радиус IDiametralDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве опорного объекта может быть использована окружность или дуга окружности.

Чтобы разорвать связь с объектом, нужно установить значение свойства равным NULL.

DimensionType – Тип диаметрального размера (полная размерная линия\размерная линия с обрывом)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство(*)
Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство(*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

Radius – Радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство(*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство(*)
Object.put_Radius(Radius)	Установить свойство (*)

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = Object.Xc	Получить свойство(*)

```
Object.Xc = Xc                  Установить свойство (* )
Xc = Object.GetXc()             Получить свойство (**)
Object.SetXc( Xc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )            Получить свойство(* )
Object.put_Xc(  Xc )            Установить свойство (* )
```

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                  Получить свойство(* )
Object.Yc = Yc                  Установить свойство (* )
Yc = Object.GetYc()             Получить свойство (**)
Object.SetYc( Yc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )            Получить свойство(* )
Object.put_Yc(  Yc )            Установить свойство (* )
```

Интерфейс IHeightDimension

Справка системы КОМПАС: Команда Размер высоты

`kompas.chm::/CM_ORDINATE_DIM.htm`

Диалог ввода размерной надписи размера высоты

`kompas.chm::/DLG_DIM_TEXT_ORD_INPUT.htm`

Интерфейс параметров размера высоты.

Иерархия:

IKompasAPIObject

IDrawingObject

IHeightDimension

IDimensionParams

IDimensionText

Описание:

Интерфейс позволяет получить и задать свойства размера высоты.

Интерфейс можно получить у коллекции размеров высоты, используя свойство IHeightDimensions::HeightDimension или метод IHeightDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Свойство IDimensionText::NominalValue возвращает значение высоты (Y1 - Y).

IHeightDimension – свойства

DimensionType – Тип размера

Интерфейс...

Тип данных: из перечисления ksHeightDimTypeEnum

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство(*)
Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство(*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

X – Значение координаты точки нулевого уровня по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство(*)
Object.put_X(X)	Установить свойство (*)

X1 – Значение координаты точки измеряемого уровня по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X1 = Object.X1	Получить свойство(*)
Object.X1 = X1	Установить свойство (*)
X1 = Object.GetX1()	Получить свойство (**)
Object.SetX1(X1)	Установить свойство (**)

Синтаксис COM:

Object.get_X1(&X1)	Получить свойство(*)
Object.put_X1(X1)	Установить свойство (*)

X2 – Значение координаты положения размерной надписи по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X2 = Object.X2	Получить свойство(*)
Object.X2 = X2	Установить свойство (*)
X2 = Object.GetX2()	Получить свойство (**)
Object.SetX2(X2)	Установить свойство (**)

Синтаксис COM:

Object.get_X2(&X2)	Получить свойство(*)
Object.put_X2(X2)	Установить свойство (*)

Y – Значение координаты точки нулевого уровня по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство(*)
Object.put_Y(Y)	Установить свойство (*)

Y1 – Значение координаты точки измеряемого уровня по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y1 = Object.Y1	Получить свойство(*)
Object.Y1 = Y1	Установить свойство (*)
Y1 = Object.GetY1()	Получить свойство (**)
Object.SetY1(Y1)	Установить свойство (**)

Синтаксис COM:

Object.get_Y1(&Y1)	Получить свойство(*)
Object.put_Y1(Y1)	Установить свойство (*)

Y2 – Значение координаты положения размерной надписи по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y2 = Object.Y2	Получить свойство(*)
Object.Y2 = Y2	Установить свойство (*)
Y2 = Object.GetY2()	Получить свойство (**)
Object.SetY2(Y2)	Установить свойство (**)

Синтаксис COM:

Object.get_Y2(&Y2)	Получить свойство(*)
Object.put_Y2(Y2)	Установить свойство (*)

Интерфейс ILineDimension

Справка системы КОМПАС: Команда Линейный размер (графический документ)

`kompas.chm::/CM_DIML.htm`

Интерфейс параметров линейного размера.

Иерархия:

IKompasAPIObject

IDrawingObject

ILineDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства линейного размера.

Интерфейс можно получить у коллекции линейных размеров, используя свойство ILineDimensions::LineDimension или метод ILineDimensions::Add.

После задания параметров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Свойство IDimensionText::NominalValue возвращает расстояние между точками (X1, Y1) и (X2, Y2).

ILineDimension – свойства

Angle – Угол наклона размера

Справка системы КОМПАС: Угол наклона выносных линий

`kompas.chm::/192_24_1_3_Razmer_s_naklonnymi_.htm`

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство(*)
Object.put_Angle(Angle)	Установить свойство (*)

Примечание:

1. Свойство позволяет задавать угол наклона выносных линий размера.
2. Свойство используется при создании размера, параллельного объекту. Свойство ILineDimension::Orientation должно быть равно ksLinDParallel.

Orientation – Тип ориентации линейного размера

Интерфейс...

Тип данных: из перечисления ksLineDimensionOrientationEnum

Синтаксис Automation:

```
Orientation = Object.Orientation      Получить свойство(* )
Object.Orientation = Orientation      Установить свойство (* )
Orientation =                         Получить свойство (**)
Object.GetOrientation()
Object.SetOrientation( Orientation    Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_Orientation(               Получить свойство(* )
&Orientation )
Object.put_Orientation(               Установить свойство (* )
Orientation )
```

ShelfX – Координата X точки начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfX = Object.ShelfX                Получить свойство(* )
Object.ShelfX = ShelfX                Установить свойство (* )
ShelfX = Object.GetShelfX()           Получить свойство (**)
Object.SetShelfX( ShelfX )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfX( &ShelfX )          Получить свойство(* )
Object.put_ShelfX( ShelfX )           Установить свойство (* )
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

ShelfY – Координата Y точки начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfY = Object.ShelfY                  Получить свойство(* )
Object.ShelfY = ShelfY                  Установить свойство (* )
ShelfY = Object.GetShelfY()             Получить свойство (**)
Object.SetShelfY( ShelfY )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfY( &ShelfY )        Получить свойство(* )
Object.put_ShelfY(  ShelfY )        Установить свойство (* )
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

X1 – Координата X первой точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = Object.X1                          Получить свойство(* )
Object.X1 = X1                          Установить свойство (* )
X1 = Object.GetX1()                     Получить свойство (**)
Object.SetX1( X1 )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X1( &X1 )                Получить свойство(* )
Object.put_X1(  X1 )                Установить свойство (* )
```

X2 – Координата X второй точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = Object.X2                          Получить свойство(* )
Object.X2 = X2                          Установить свойство (* )
X2 = Object.GetX2()                     Получить свойство (**)
Object.SetX2( X2 )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X2( &X2 )                Получить свойство(* )
Object.put_X2(  X2 )                Установить свойство (* )
```

X3 – Координата X положения размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = Object.X3                  Получить свойство(* )
Object.X3 = X3                  Установить свойство (* )
X3 = Object.GetX3()             Получить свойство (**)
Object.SetX3( X3 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_X3( &X3 )            Получить свойство(* )
Object.put_X3(  X3 )            Установить свойство (* )
```

Y1 – Координата Y первой точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = Object.Y1                  Получить свойство(* )
Object.Y1 = Y1                  Установить свойство (* )
Y1 = Object.GetY1()             Получить свойство (**)
Object.SetY1( Y1 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y1( &Y1 )            Получить свойство(* )
Object.put_Y1(  Y1 )            Установить свойство (* )
```

Y2 – Координата Y второй точки привязки размера

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = Object.Y2                  Получить свойство(* )
Object.Y2 = Y2                  Установить свойство (* )
Y2 = Object.GetY2()             Получить свойство (**)
Object.SetY2( Y2 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Y2( &Y2 )            Получить свойство(* )
Object.put_Y2(  Y2 )            Установить свойство (* )
```

Y3 – Координата Y положения размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y3 = Object.Y3	Получить свойство(*)
Object.Y3 = Y3	Установить свойство (*)
Y3 = Object.GetY3()	Получить свойство (**)
Object.SetY3(Y3)	Установить свойство (**)

Синтаксис COM:

Object.get_Y3(&Y3)	Получить свойство(*)
Object.put_Y3(Y3)	Установить свойство (*)

Интерфейс IRadialDimension

Справка системы КОМПАС: Команда Радиальный размер

`kompas.chm::/CM_DIMR.htm`

Интерфейс параметров радиального размера.

Иерархия:

IKompasAPIObject

IDrawingObject

IRadialDimension

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства радиального размера.

Интерфейс можно получить у коллекции радиальных размеров, используя свойство IRadialDimensions::RadialDimension или метод IRadialDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface

Примечание:

Если при создании размера задать опорный объект IRadialDimension::BaseObject, то координаты центра IRadialDimension::Xc IRadialDimension::Yc и радиус IRadialDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта. Свойство IDimensionText::NominalValue возвращает диаметр.

В качестве опорного объекта может быть использована окружность или дуга окружности.

IRadialDimension – свойства

Angle – Угол наклона размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство(*)
Object.put_Angle(Angle)	Установить свойство (*)

BaseObject – Опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство(*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство(*)
Object.put_BaseObject(BaseObject)	Установить свойство (*)

Примечание:

Если при создании размера задать опорный объект IRadialDimension::BaseObject, то координаты центра IRadialDimension::Xc IRadialDimension::Yc и радиус IRadialDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве опорного объекта может быть использована окружность или дуга окружности.

Чтобы разорвать связь с объектом, нужно установить значение свойства равным NULL.

BranchBegin – Начало ответвления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BranchBegin = Object.BranchBegin(Index)	Получить свойство(*)
Object.BranchBegin(Index) = BranchBegin	Установить свойство (*)
BranchBegin = Object.GetBranchBegin(Index)	Получить свойство (**)
BranchBegin = Object.GetBranchBegin(Index)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchBegin(Index, &BranchBegin)	Получить свойство
Object.put_BranchBegin(Index, BranchBegin)	Установить свойство

Входные параметры:

long Index - номер ответвления

Значения свойства:

TRUE	- от начала полки,
FALSE	- от конца полки.

BranchsCount – Количество ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchsCount = Object.BranchsCount	Получить свойство(*)
BranchsCount = Object.GetBranchsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_BranchsCount(&BranchsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BranchObject – Опорный объект для дополнительного ответвления

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

BranchObject = Object.BranchObject(Index)	Получить свойство(*)
Object.BranchObject(Index) = BranchObject	Установить свойство (*)
BranchObject = Object.GetBranchObject(Index)	Получить свойство (**)
Object.SetBranchObject(Index, BranchObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchObject(Index, &BranchObject)	Получить свойство
Object.put_BranchObject(Index, BranchObject)	Установить свойство

Входные параметры:

long Index - номер ответвления.

Свойство позволяет устанавливать и получать объект, на который указывает ответвление.

DimensionType – Тип радиального размера (от центра\не от центра)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство(*)
Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство(*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

Radius – Радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius                Получить свойство(* )
Object.Radius = Radius                Установить свойство (* )
Radius = Object.GetRadius()           Получить свойство (**)
Object.SetRadius( Radius )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius )          Получить свойство(* )
Object.put_Radius(  Radius )          Установить свойство (* )
```

ShelfX – Точка начала полки – координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfX = Object.ShelfX                Получить свойство(* )
Object.ShelfX = ShelfX                Установить свойство (* )
ShelfX = Object.GetShelfX()           Получить свойство (**)
Object.SetShelfX( ShelfX )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfX( &ShelfX )          Получить свойство
Object.put_ShelfX(  ShelfX )          Установить свойство
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

ShelfY – Точка начала полки – координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfY = Object.ShelfY                Получить свойство(* )
Object.ShelfY = ShelfY                Установить свойство (* )
ShelfY = Object.GetShelfY()           Получить свойство (**)
Object.SetShelfY( ShelfY )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShelfY( &ShelfY )        Получить свойство
Object.put_ShelfY(  ShelfY )        Установить свойство
```

Примечание:

Перед заданием свойства нужно установить признак размещения на полке TextType = ksDTPOnShelf.

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = Object.Xc                      Получить свойство(* )
Object.Xc = Xc                      Установить свойство (* )
Xc = Object.GetXc()                 Получить свойство (**)
Object.SetXc( Xc )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )                Получить свойство(* )
Object.put_Xc(  Xc )                Установить свойство (* )
```

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                      Получить свойство(* )
Object.Yc = Yc                      Установить свойство (* )
Yc = Object.GetYc()                 Получить свойство (**)
Object.SetYc( Yc )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )                Получить свойство(* )
Object.put_Yc(  Yc )                Установить свойство (* )
```

IRadialDimension – методы

AddBranch – Добавить ответвление для объекта

Интерфейс...

Синтаксис Automation:

BOOL AddBranch(BOOL BranchBegin,
IDrawingObject * BranchObject);

Синтаксис COM:

HRESULT AddBranch(BOOL BranchBegin,
IDrawingObject * BranchObject,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BranchBegin	- строить от начала полки,
BranchObject	- опорный объект.

Метод позволяет добавить ответвление, используя опорный объект.

AddBranchByArcParam – Добавить ответвление по параметрам дуги

Интерфейс...

Синтаксис Automation:

BOOL AddBranchByArcParam(BOOL BranchBegin,
double Xc,
double Yc,
double Radius,
double Angle1,
double Angle2,
BOOL Direction);

Синтаксис COM:

HRESULT AddBranchByArcParam(BOOL BranchBegin,
double Xc,
double Yc,
double Radius,
double Angle1,
double Angle2,
BOOL Direction,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

BranchBegin	- строить от начала полки,
Xc,Yc	- координаты центра дуги,
Radius	- радиус дуги,
Angle1	- 1-й угол дуги,
Angle2	- 2-й угол дуги,
Direction	- направление дуги.

Метод позволяет построить ответвление, используя параметры дуги.

DeleteBranch – Удалить ответвление

Интерфейс...

Синтаксис Automation:

BOOL DeleteBranch(long Index);

Синтаксис COM:

HRESULT DeleteBranch(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index - номер ответвления.

GetBranchParam – Получить параметры ответвления для объекта

Интерфейс...

Синтаксис Automation:

BOOL GetBranchParam(long Index,
double * Xc,
double * Yc,
double * Angle1,
double * Angle2,
BOOL * Direction);

Синтаксис COM:

HRESULT GetBranchParam(long Index,
double * Xc,
double * Yc,
double * Angle1,
double * Angle2,
BOOL * Direction,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index - номер ответвления.

Выходные параметры:

Xc,Yc	- координаты центра дуги,
Angle1	- 1-й угол дуги,
Angle2	- 2-й угол дуги,
Direction	- направление дуги.

SetBranchParam – Установить параметры ответвления для объекта

Интерфейс...

Синтаксис Automation:

BOOL SetBranchParam(long Index, double Xc, double Yc, double Angle1, double Angle2, BOOL Direction);

Синтаксис COM:

HRESULT SetBranchParam(long Index, double Xc, double Yc, double Angle1, double Angle2, BOOL Direction, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- номер ответвления,
Xc,Yc	- координаты центра дуги,
Radius	- радиус дуги,
Angle1	- 1-й угол дуги,
Angle2	- 2-й угол дуги,

Direction - направление дуги.

Интерфейс IBreakRadialDimension

Справка системы КОМПАС: Команда Радиальный размер с изломом

kompas.chm::/CM_DIMR_WITH_BREAK.htm

Интерфейс параметров радиального размера с изломом.

Иерархия:

IKompasAPIObject

IDrawingObject

IBreakRadialDimension

IDimensionText

Описание:

Интерфейс позволяет получить и задать свойства радиального размера с изломом.

Интерфейс можно получить у коллекции радиальных размеров с изломом, используя свойство IBreakRadialDimensions::BreakRadialDimension или метод IBreakRadialDimensions::Add.

После задания параметров размера требуется вызвать метод IDrawingObject::Update.

Интерфейс IDimensionText является дополнительным. Его можно получить с помощью метода IUnknown::QueryInterface. Интерфейс IDimensionParams для радиального размер с изломом не поддерживается.

Примечание:

Если при создании размера задать опорный объект IBreakRadialDimension::BaseObject, то координаты центра IBreakRadialDimension::Xc, IBreakRadialDimension::Yc и радиус IBreakRadialDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта. Свойство IDimensionText::NominalValue возвращает радиус.

В качестве опорного объекта может быть использована окружность или дуга окружности.

IBreakRadialDimension – свойства

Angle – Угол наклона размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Angle( &Angle )          Получить свойство(* )
Object.put_Angle(  Angle )          Установить свойство (* )
```

BaseObject – Опорный объект

Интерфейс...

Тип данных: интерфейс IDrawingObject

Синтаксис Automation:

```
BaseObject = Object.BaseObject      Получить свойство(* )
Object.BaseObject = BaseObject      Установить свойство (* )
BaseObject =                        Получить свойство (**)
Object.GetBaseObject()
Object.SetBaseObject( BaseObject    Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_BaseObject(              Получить свойство(* )
&BaseObject )
Object.put_BaseObject(              Установить свойство (* )
BaseObject )
```

Примечание:

Если при создании размера задать опорный объект IBreakRadialDimension::BaseObject, то координаты центра IBreakRadialDimension::Xc IBreakRadialDimension::Yc и радиус IBreakRadialDimension::Radius задавать не нужно. Данные свойства будут получены с опорного объекта.

В качестве опорного объекта может быть использована окружность или дуга окружности.

Чтобы разорвать связь с объектом, нужно установить значение свойства равным NULL.

BreakAngle – Угол излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BreakAngle = Object.BreakAngle      Получить свойство(* )
Object.BreakAngle = BreakAngle      Установить свойство (* )
BreakAngle =                        Получить свойство (**)
Object.GetBreakAngle()
Object.SetBreakAngle( BreakAngle    Установить свойство (**)
)
```

Синтаксис COM:

Object.get_BreakAngle (&BreakAngle)	Получить свойство
Object.put_BreakAngle (BreakAngle)	Установить свойство

BreakLength – Длина излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakLength = Object.BreakLength	Получить свойство(*)
Object.BreakLength = BreakLength	Установить свойство (*)
BreakLength = Object.GetBreakLength()	Получить свойство (**)
Object.SetBreakLength(BreakLength)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakLength(&BreakLength)	Получить свойство(*)
Object.put_BreakLength(BreakLength)	Установить свойство (*)

BreakX1 – Координата X начала излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakX1 = Object.BreakX1	Получить свойство(*)
Object.BreakX1 = BreakX1	Установить свойство (*)
BreakX1 = Object.GetBreakX1()	Получить свойство (**)
Object.SetBreakX1(BreakX1)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakX1(&BreakX1)	Получить свойство
Object.put_BreakX1(BreakX1)	Установить свойство

BreakX2 – Координата X конца излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakX2 = Object.BreakX2	Получить свойство(*)
Object.BreakX2 = BreakX2	Установить свойство (*)
BreakX2 = Object.GetBreakX2()	Получить свойство (**)
Object.SetBreakX2(BreakX2)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakX2(&BreakX2)	Получить свойство
Object.put_BreakX2(BreakX2)	Установить свойство

BreakY1 – Координата Y начала излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakY1 = Object.BreakY1	Получить свойство(*)
Object.BreakY1 = BreakY1	Установить свойство (*)
BreakY1 = Object.GetBreakY1()	Получить свойство (**)
Object.SetBreakY1(BreakY1)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakY1(&BreakY1)	Получить свойство
Object.put_BreakY1(BreakY1)	Установить свойство

BreakY2 – Координата Y конца излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

BreakY2 = Object.BreakY2	Получить свойство(*)
Object.BreakY2 = BreakY2	Установить свойство (*)
BreakY2 = Object.GetBreakY2()	Получить свойство (**)

Object.SetBreakY2(BreakY2)	Установить свойство (**)

Синтаксис COM:

Object.get_BreakY2(&BreakY2)	Получить свойство
Object.put_BreakY2(BreakY2)	Установить свойство

Radius – Радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство(*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство(*)
Object.put_Radius(Radius)	Установить свойство (*)

TextOnLine – Положение размерной надписи относительно размерной линии

Интерфейс...

Тип данных: из перечисления ksDimensionTextPosEnum

Синтаксис Automation:

TextOnLine = Object.TextOnLine	Получить свойство(*)
Object.TextOnLine = TextOnLine	Установить свойство (*)
TextOnLine = Object.GetTextOnLine()	Получить свойство (**)
Object.SetTextOnLine(TextOnLine)	Установить свойство (**)

Синтаксис COM:

Object.get_TextOnLine(&TextOnLine)	Получить свойство(*)
Object.put_TextOnLine(TextOnLine)	Установить свойство (*)

Примечание:

Позволяет задать положение размерной надписи относительно размерной линии. Умолчательное положение определяется в

диалоге настройки положения надписи

`kompas.chm::/CM_DIMR_WITH_BREAK.htm`
сделанной для текущего документа.

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = Object.Xc                  Получить свойство(* )
Object.Xc = Xc                  Установить свойство (* )
Xc = Object.GetXc()             Получить свойство (**)
Object.SetXc( Xc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )            Получить свойство(* )
Object.put_Xc(  Xc )            Установить свойство (* )
```

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                  Получить свойство(* )
Object.Yc = Yc                  Установить свойство (* )
Yc = Object.GetYc()             Получить свойство (**)
Object.SetYc( Yc )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )            Получить свойство(* )
Object.put_Yc(  Yc )            Установить свойство (* )
```

Обозначения для строительства

Интерфейс IBuildingContainer

Контейнер объектов СПДС.

Иерархия:

IDispatch

IBuildingContainer

Описание:

Позволяет получить доступ к коллекциям объектов СПДС.

Примечание:

Дополнительный интерфейс вида. Данный интерфейс можно получить у вида IView посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IBuildingContainer – свойства

Braces – Интерфейс коллекции фигурных скобок

Интерфейс...

Тип данных: указатель на интерфейс коллекции фигурных скобок IBraces

Синтаксис Automation:

IBraces = iObject.Braces;	Получить свойство(*)
IBraces = iObject.GetBraces();	Получить свойство (**)

Синтаксис COM:

iObject->get_Braces(&Braces);	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать фигурные скобки.

BuildingAxes – Интерфейс коллекции строительных осей

Интерфейс...

Тип данных: указатель на интерфейс IBuildingAxes коллекции строительных осей

Синтаксис Automation:

BuildingAxes = iObject.BuildingAxes	Получить свойство(*)
BuildingAxes = iObject.GetBuildingAxes()	Получить свойство (**)

Синтаксис COM:

iObject->get_BuildingAxes(&BuildingAxes)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать строительные оси.

BuildingCutLines – Коллекция линий разреза для СПДС

Интерфейс...

Тип данных: указатель на интерфейс ICutLines

Синтаксис Automation:

BuildingCutLines = Object.BuildingCutLines	Получить свойство(*)
BuildingCutLines = Object.GetBuildingCutLines()	Получить свойство (**)

Синтаксис COM:

Object.get_BuildingCutLines(&BuildingCutLines)	Получить свойство

Свойство позволяет получать интерфейс коллекции линий разреза/сечения для СПДС.

Примечание:

Свойство доступно только для чтения.

CutUnitMarkings – Интерфейс коллекции обозначений узла в сечении

Интерфейс...

Тип данных: указатель на интерфейс коллекции обозначений узла в сечении ICutUnitMarkings

Синтаксис Automation:

CutUnitMarkings = iObject.CutUnitMarkings	Получить свойство(*)
CutUnitMarkings = iObject.GetCutUnitMarkings()	Получить свойство (**)

Синтаксис COM:

iObject->get_CutUnitMarkings(&CutUnitMarkings)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать обозначения узла в сечении.

Marks – Интерфейс коллекции марок

Интерфейс...

Тип данных: указатель на интерфейс коллекции марок IMarks

Синтаксис Automation:

Marks = iObject.Marks	Получить свойство(*)
Marks = iObject.GetMarks()	Получить свойство (**)

Синтаксис COM:

iObject->get_Marks(&Marks)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать марки.

MultiTextLeaders – Интерфейс коллекции выносных надписей к многослойным конструкциям

Интерфейс...

Тип данных: указатель на интерфейс коллекции надписей к многослойным конструкциям IMultiTextLeaders

Синтаксис Automation:

MultiTextLeaders = iObject.MultiTextLeaders	Получить свойство(*)
MultiTextLeaders = iObject.GetMultiTextLeaders()	Получить свойство (**)

Синтаксис COM:

iObject->get_MultiTextLeaders(&MultiTextLeaders)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать надписи к многослойным конструкциям.

UnitMarkings – Интерфейс коллекции обозначений узлов

Интерфейс...

Тип данных: указатель на интерфейс коллекции обозначений узлов IUnitMarkings

Синтаксис Automation:

UnitMarkings = iObject.UnitMarkings	Получить свойство(*)
UnitMarkings = iObject.GetUnitMarkings()	Получить свойство (**)

Синтаксис COM:

iObject->get_UnitMarkings(&UnitMarkings)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать обозначения узлов.

UnitNumbers – Интерфейс коллекции номеров узла

Интерфейс...

Тип данных: указатель на интерфейс коллекции номеров узла IUnitNumbers

Синтаксис Automation:

IUnitNumbers = iObject.UnitNumbers;	Получить свойство(*)
IUnitNumbers = iObject.GetUnitNumbers();	Получить свойство (**)

Синтаксис COM:

iObject->get_UnitNumbers(&UnitNumbers);	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция позволяет получать и создавать номера узла.

Интерфейс IBraces

Справка системы КОМПАС...

KOMPAS.chm::/CM_BRACE.htm

Интерфейс коллекции фигурных скобок.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBraces

Описание:

Интерфейс позволяет получить доступ к фигурным скобкам на чертеже.

Примечание:

Получить интерфейс коллекции фигурных скобок можно, используя метод контейнера объектов СПДС IBuildingContainer::Braces.

IBraces – свойства

Brace – Фигурная скобка, заданная по индексу, ссылке или имени и номеру

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BRACE.htm

Тип данных: указатель на интерфейс фигурной скобки IBrace

Синтаксис Automation:

Brace = iObject.Brace(index)	Получить свойство(*)
Brace = iObject.GetBrace(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Brace(&Brace)	index,	Получить свойство

Входные параметры:

Index (Variant)	Индекс.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса могут использоваться следующие типы:

- индекс объекта в коллекции,
- ссылка на объект (reference),
- строковое значение текста IBrace::Text.

Пример:

Brace = iBraces.Brace("Text");

где "Brace" - текст на фигурной скобке.

IBraces – методы

Add – Создать фигурную скобку

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BRACE.htm

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add(IBrace** Result);

Возвращаемое значение:

Result	- Указатель на интерфейс фигурной скобки IBrace.

Примечание:

Метод позволяет создать новый интерфейс фигурной скобки. После получения нового интерфейса нужно задать параметры фигурной скобки и вызвать метод IDrawingObject::Update.

Интерфейс IBuildingAxes

Интерфейс коллекции строительных (координационных) осей.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IBuildingAxes

Описание:

Интерфейс позволяет получать и создавать объекты СПДС "строительные оси".

Примечание:

Получить интерфейс коллекции строительных осей можно, используя метод контейнера объектов СПДС IBuildingContainer::BuildingAxes.

IBuildingAxes – свойства

BuildingAxis – Ось, заданная по индексу, ссылке или имени

Интерфейс...

Тип данных: указатель на интерфейс оси IBuildingAxis

Синтаксис Automation:

Axis = iObject.BuildingAxis(index)	Получить свойство(*)
Axis = iObject.GetBuildingAxis(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_BuildingAxis(index, &Axis)	Получить свойство

Входные параметры:

Index (Variant)	Индекс.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса могут использоваться следующие типы:

- Индекс объекта в коллекции,
- Ссылка на объект (reference),
- Строковое значение текста строительной оси IBuildingAxis::Text.

Пример:

axis = iBuildingAxes.BuildingAxis("A1")

где "A1" - текст оси

IBuildingAxes – методы

Add – Создать ось

Интерфейс...

Синтаксис Automation:

LPDISPATH Add(DrawingObjectTypeEnum type);

Синтаксис COM:

HRESULT Add(DrawingObjectTypeEnum type, IBuildingAxis** Result);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum.

Возможные значения type:

ksDrStraightAxis	56	IStraightAxis Прямая ось,
ksDrCircleAxis	58	ICircleAxis Круговая ось,

ksDrArcAxis	59	IArcAxis Дуговая ось.

Возвращаемое значение:

Result	- Указатель на базовый интерфейс для строительных осей IBuildingAxis.

Примечание:

Метод позволяет создать новый интерфейс строительной оси. После получения нового интерфейса нужно задать параметры марки и вызвать метод IDrawingObject::Update.

Интерфейс IMarks

Интерфейс коллекции марок (содержит марки всех типов).

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IMarks

Описание:

Интерфейс позволяет получить доступ к маркам на чертеже.

Примечание:

Получить интерфейс коллекции марок можно, используя метод контейнера объектов СПДС IBuildingContainer::Marks.

IMarks – свойства

Mark – Марка, заданная по индексу, ссылке или имени и номеру

Интерфейс...

Тип данных: указатель на интерфейс марки IMark

Синтаксис Automation:

Mark = iObject.Mark(index) Mark = iObject.GetMark(index)	Получить свойство(*) Получить свойство (**)

Синтаксис COM:

iObject->get_Mark(index, &Mark)	Получить свойство

Входные параметры:

Index (Variant) - Индекс.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса могут использоваться следующие типы:

- Индекс объекта в коллекции,
- Ссылка на объект (reference),
- Суммарная строка – имя (IMark::Name) + номер марки (IMark::Number) без дополнительных разделителей.

Пример:

```
mark = iMarks.Mark("A1");
```

где "A" - имя марки

"1" - номер марки

IMarks – методы

Add – Создать марку

Интерфейс...

Синтаксис Automation:

LPDISPATH Add(DrawingObjectTypeEnum MarkType);

Синтаксис COM:

HRESULT Add([in] DrawingObjectTypeEnum MarkType,
[out, retval] IMark** Result);

Входные параметры:

MarkType - тип объекта из перечисления DrawingObjectTypeEnum.

Возможные значения MarkType

ksDrMarkOnLeader	52	IMarkOnLeader - Марка/позиционное обозначение с линией-выноской.
ksDrMarkOnLine	53	IMarkOnLine - Марка/позиционное обозначение на линии.
ksDrMarkInsideForm	54	IMarkInsideForm - Марка/позиционное обозначение без линии-выноски.

Возвращаемое значение:

Result - указатель на базовый интерфейс для марок IMark.

Примечание:

Метод позволяет создать новый интерфейс марки. После получения нового интерфейса нужно задать параметры марки и вызвать метод IDrawingObject::Update.

Интерфейс ICutUnitMarkings

Справка системы КОМПАС...

KOMPAS.chm::/CM_CUTNODE.htm

Интерфейс коллекции обозначений узла в сечении.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

ICutUnitMarkings

Описание:

Интерфейс позволяет получить коллекции обозначений узла в сечении.

Примечание:

Данный интерфейс можно получить у интерфейса контейнера объектов СПДС, используя метод IBuildingContainer::CutUnitMarkings.

ICutUnitMarkings – свойства

CutUnitMarking – Обозначение узла в сечении, заданное по индексу, ссылке или тексту

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CUTNODE.htm

Тип данных: указатель на интерфейс ICutUnitMarking

Синтаксис Automation:

CutUnitMarking = iObject.CutUnitMarking(index)	Получить свойство(*)
CutUnitMarking = iObject.GetCutUnitMarking(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_CutUnitMarking(index, &CutUnitMarking)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс обозначения узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта - строковое значение текста сверху ICutUnitMarking::TextUp.

Пример:

mark = iMarks.Mark("A1");

где "A1" - текст сверху

Примечание:

Свойство доступно только для чтения.

ICutUnitMarkings – методы

Add – Создать обозначение узла в сечении и добавить его в коллекцию

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_CUTNODE.htm`

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(ICutUnitMarking** Result);

Возвращаемое значение:

	- Указатель на интерфейс обозначения узла в сечении ICutUnitMarking.

Примечание:

Метод позволяет создать новый интерфейс обозначения узла в сечении. После получения нового интерфейса нужно задать параметры обозначения узла и вызвать метод IDrawingObject::Update.

Интерфейс IMultiTextLeaders

Интерфейс коллекции выносных надписей к многослойным конструкциям.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IMultiTextLeaders

Описание:

Интерфейс позволяет получить доступ к выносным надписям к многослойным конструкциям для данного вида, а также позволяет создавать новые выносные надписи к многослойным конструкциям.

Примечание:

Получить интерфейс можно, используя свойство IBuildingContainer::MultiTextLeaders.

IMultiTextLeaders – свойства

MultiTextLeader – Выносная надпись к многослойным конструкциям, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс IMultiTextLeader

Синтаксис Automation:

MultiTextLeader = iObject.MultiTextLeader(Index)	Получить свойство(*)
MultiTextLeader = iObject.GetMultiTextLeader(Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_MultiTextLeader(Index, &MultiTextLeader)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, reference объекта.

Значения свойства:

указатель на интерфейс IMultiTextLeader	- в случае успеха,
NULL	- в случае неудачи.

Примечание:

Свойство доступно только для чтения.

IMultiTextLeaders – методы

Add – Создать выносную надпись к многослойным конструкциям (Добавить объект в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add([out, retval] IMultiTextLeader** Result);

Возвращаемое значение:

- указатель на интерфейс IMultiTextLeader

Примечание:

После получения нового интерфейса нужно задать параметры объекта и вызвать метод IDrawingObject::Update. Объект в модели появится после вызова этого метода.

Интерфейс IUnitMarkings

Справка системы КОМПАС...

KOMPAS.chm::/CM_SIGNNODE.htm

Интерфейс коллекции обозначений узла.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IUnitMarkings

Описание:

Интерфейс позволяет получить коллекции обозначений узла.

Примечание:

Данный интерфейс можно получить у интерфейса контейнера объектов СПДС, используя метод IBuildingContainer::UnitMarkings.

IUnitMarkings – свойства

UnitMarking – Обозначение узла, заданное по индексу, ссылке или тексту

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_SIGNNODE.htm

Тип данных: указатель на интерфейс IUnitMarking

Синтаксис Automation:

UnitMarking = iObject.UnitMarking(index) UnitMarking = iObject.GetUnitMarking(index)	Получить свойство(*) Получить свойство (**)

Синтаксис COM:

iObject->get_UnitMarking(index, &UnitMarking)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс обозначения узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс объекта, - reference объекта - строковое значение текста сверху IUnitMarking::TextUp.

Пример:

UnitMarking = iUnitMarkings.UnitMarking("A1");

где "A1" - текст сверху

Примечание:

Свойство доступно только для чтения.

IUnitMarkings – методы

Add – Создать обозначение узла и добавить его в коллекцию

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_SIGNNODE.htm

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IUnitMarking** Result);

Возвращаемое значение:

- Указатель на интерфейс обозначения узла IUnitMarking.

Примечание:

Метод позволяет создать новый интерфейс обозначения узла. После получения нового интерфейса нужно задать параметры обозначения узла и вызвать метод IDrawingObject::Update.

Интерфейс IUnitNumbers

Справка системы КОМПАС...

KOMPAS.chm::/CM_KNOTNUMBER.htm

Интерфейс коллекции номеров узла.

Иерархия:

IKompasAPIObject

IKompasCollection

IDrawingObjects

IUnitNumbers

Описание:

Интерфейс позволяет получить доступ к номерам узла на чертеже.

Примечание:

Получить интерфейс коллекции номеров узла можно, используя метод контейнера объектов СПДС IBuildingContainer::UnitNumbers.

IUnitNumbers – свойства

UnitNumber – Номер узла, заданный по индексу, или ссылке

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_KNOTNUMBER.htm

Тип данных: указатель на интерфейс марки IUnitNumber

Синтаксис Automation:

UnitNumber = iObject.UnitNumber(index)	Получить свойство(*)
UnitNumber = iObject.GetUnitNumber(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_UnitNumber(index, &UnitNumber)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса могут использоваться следующие типы:

- индекс объекта в коллекции,
- ссылка на объект (reference),
- строковое значение текста сверху IUnitNumber::TextUp.

Пример:

UnitNumber = iUnitNumbers.UnitNumber("TextUp");

Где "TextUp" - верхний текст номера узла.

IUnitNumbers – методы

Add – Создать номер узла

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_KNOTNUMBER.htm`

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add(IUnitNumber** Result);

Возвращаемое значение:

Result	- Указатель на интерфейс номера узла IUnitNumber.

Примечание:

Метод позволяет создать новый интерфейс номера узла. После получения нового интерфейса нужно задать параметры номера узла и вызвать метод IDrawingObject::Update.

Интерфейс IMarkNodes

Интерфейс коллекции узлов для вставки дополнительных марок.

Иерархия:

IKompasAPIObject

IKompasCollection

IMarkNodes

Описание:

Интерфейс позволяет получать и создавать дополнительные узлы для вставки марок.

Примечание:

Получить интерфейс коллекции марок можно, используя методы IStraightAxis::MarkNodes, IArcAxis::MarkNodes, IMarkNode::MarkNodes.

IMarkNodes – свойства

Item – Узел, заданный по индексу или тексту марки

Интерфейс...

Тип данных: указатель на интерфейс узла IMarkNode

Синтаксис Automation:

Item = iObject.Item(Index)	Получить свойство(*)
Item = iObject.GetItem(Index)	Получить свойство (**)

Синтаксис COM:

Item = iObject->get_Item(Index)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс узла в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс узла, - VT_BSTR - имя узла.

Описание:

Индекс равный 0 позволяет получить указатель на узел, являющийся маркой оси.

Примечание:

Свойство доступно только для чтения.

IMarkNodes – методы

Add – Создать узел (добавить узел в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATH Add(ksMarkNodeEnum Type,
VARIANT Index);

Синтаксис COM:

HRESULT Add(ksMarkNodeEnum Type,
VARIANT Index, IMarkNode** Result);

Входные параметры:

Type	- тип марки из перечисления ksMarkNodeEnum,
Index	- индекс узла для вставки марки.
	Поддерживаются следующие типы:
	VT_I4 - добавление узла по индексу, -1 - добавление в конец списка,
	VT_BSTR - добавление узла перед элементом с заданным именем,
	VT_DISPATCH - добавление узла перед заданным.

Возвращаемое значение:

- Указатель на интерфейс узла IMarkNode.

Примечание:

Метод позволяет создать дополнительные узлы для марок.

Clear – Очистить коллекцию

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить дополнительные узлы для марок, при этом узлы удаляются из коллекции. Из объекта узлы будут удалены после вызова метода IDrawingObject::Update.

Интерфейс IBuildingAxis

Строительная ось (базовый интерфейс для строительных осей).

Иерархия:

IKompasAPIObject

IDrawingObject

IBuildingAxis

Описание:

Интерфейс позволяет получать и изменять общие свойства для строительных осей.

Примечание:

Получить интерфейс строительной оси можно, используя методы коллекции строительных осей IBuildingAxes::BuildingAxis, IBuildingAxes::Add.

IBuildingAxis – свойства

AutoStroke – Автоопределение длины штриха

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoStroke = iObject.AutoStroke	Получить свойство(*)
iObject.AutoStroke = AutoStroke	Установить свойство (*)
AutoStroke = iObject.GetAutoStroke()	Получить свойство (**)
iObject.SetAutoStroke(AutoStroke)	Установить свойство (**)

Синтаксис COM:

iObject->get_AutoStroke(&AutoStroke)	Получить свойство
iObject->put_AutoStroke(AutoStroke)	Установить свойство

DottedLength – Длина пунктира

Интерфейс...

Тип данных: double

Синтаксис Automation:

DottedLength = iObject.DottedLength	Получить свойство(*)
iObject.DottedLength = DottedLength	Установить свойство (*)
iObject.GetDottedLength()	Получить свойство (**)
iObject.SetDottedLength(DottedLength)	Установить свойство (**)

Синтаксис COM:

iObject->get_DottedLength(&DottedLength)	Получить свойство

iObject->put_DottedLength(DottedLength)	Установить свойство

DoubleMark – Тип марки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DoubleMark = iObject.DoubleMark	Получить свойство(*)
iObject.DoubleMark = DoubleMark	Установить свойство (*)
DoubleMark = iObject.GetDoubleMark()	Получить свойство (**)
iObject.SetDoubleMark(DoubleMark)	Установить свойство (**)

Синтаксис COM:

iObject->get_DoubleMark(&DoubleMark)	Получить свойство
iObject->put_DoubleMark(DoubleMark)	Установить свойство

Значения свойства:

TRUE	- двойная,
FALSE	- одинарная.

Примечание:

Свойство позволяет получить и установить тип марки.

Interval – Длина промежутка

Интерфейс...

Тип данных: double

Синтаксис Automation:

Interval = iObject.Interval	Получить свойство(*)
iObject.Interval = Interval	Установить свойство (*)
Interval = iObject.GetInterval()	Получить свойство (**)
iObject.SetInterval(Interval)	Установить свойство (**)

Синтаксис COM:

iObject->get_Interval(&Interval)	Получить свойство
iObject->put_Interval(Interval)	Установить свойство

MarkSize – Размер марки

Интерфейс...

Тип данных: double

Синтаксис Automation:

MarkSize = iObject.MarkSize	Получить свойство(*)
iObject.MarkSize = MarkSize	Установить свойство (*)
MarkSize = iObject.GetMarkSize()	Получить свойство (**)
iObject.SetMarkSize(MarkSize)	Установить свойство (**)

Синтаксис COM:

iObject->get_MarkSize(&MarkSize)	Получить свойство
iObject->put_MarkSize(MarkSize)	Установить свойство

Примечание:

Значения параметра пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Stroke – Длина штриха

Интерфейс...

Тип данных: double

Синтаксис Automation:

Stroke = iObject.Stroke	Получить свойство(*)
iObject.Stroke = Stroke	Установить свойство (*)
Stroke = iObject.GetStroke()	Получить свойство (**)
iObject.SetStroke(Stroke)	Установить свойство (**)

Синтаксис COM:

iObject->get_Stroke(&Stroke)	Получить свойство
iObject->put_Stroke(Stroke)	Установить свойство

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = iObject.Text	Получить свойство(*)

Text = iObject.GetText() Получить свойство (**)

Синтаксис COM:

iObject->get_Text(&Text) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Значение свойства можно использовать для поиска марки в коллекции строительных осей IBuildingAxes, а также для поиска в функции IDrawingObjects::Item или IBuildingAxes::BuildingAxis строкового значения текста.

Пример:

axis = iBuildingAxes.Axis("A1");

TextAfter – Текст после

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = iObject.TextAfter Получить свойство(*)
Text = iObject.GetTextAfter() Получить свойство (**)

Синтаксис COM:

iObject->get_TextAfter(&Text) Получить свойство

Примечание:

Свойство доступно только для чтения.

TextBefore – Текст до

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = iObject.TextBefore Получить свойство(*)
Text = iObject.GetTextBefore() Получить свойство (**)

Синтаксис COM:

iObject->get_TextBefore(&Text) Получить свойство

Примечание:

Свойство доступно только для чтения.

IBuildingAxis – методы

AddNodeByPoint – Добавить узел марки по координатам

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddNodeByPoint(ksMarkNodeEnum Type,
double X,
double Y);

Синтаксис COM:

HRESULT AddNodeByPoint(ksMarkNodeEnum Type,
double X,
double Y,
IMarkNode ** PVal);

Входные параметры:

Type	- тип узла для марки из ksMarkNodeEnum,
X, Y	- координаты для подключения узла.

Возвращаемое значение:

Указатель на интерфейс IMarkNode	- если узел подключен,
NULL	- в случае неудачи.

Примечание:

Значения координат и направлений пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

GetInsertionPoints – Получить координаты точек для вставки дополнительных элементов

Интерфейс...

Синтаксис Automation:

BOOL GetInsertionPoints(BOOL First,
VARIANT * Points,
VARIANT * Directions,
VARIANT * Nodes,
BOOL * RetVal);

Синтаксис COM:

HRESULT GetInsertionPoints(BOOL First,
VARIANT * Points,
VARIANT * Directions,

VARIANT * Nodes,
BOOL * RetVal);

Входные параметры:

First	- вернуть координаты точек TRUE - для первого ответвления, FALSE - для второго ответвления.

Выходные параметры:

Points	- массив SafeArray типа VT_ARRAY I VT_R8 координат точек для подключения дополнительных узлов марок,
Directions	- массив SafeArray типа VT_ARRAY I VT_R8 направлений (углы относительно центра родительского узла),
Nodes	- массив SafeArray типа VT_ARRAY I VT_DISPATCH родительские узлы для точек подключения.

Возвращаемое значение:

TRUE	- координаты получены,
FALSE	- в случае неудачи.

Примечание:

1. Параметры Points, Directions и Nodes не являются обязательными. В функцию достаточно передать один из указателей на VARIANT.
2. Массивы являются согласованными.

   Координаты точек в массиве Points лежат в следующей последовательности:

- x0, y0, x1, y1, ...xi, yi

  Направления в масиве Directions лежат в последовательности:

- angle0, angle1, ...anglei

  В массиве Nodes лежат указатели на интерфейсы узлов для вставок марок IMarkNode.

3. У узла может быть от 1 до 3 точек для подключения дополнительных узлов в зависимости от типа узла и его расположения. Так как массивы координат и узлов должны быть согласованы, то один и тот же указатель на интерфейс может встречаться в массиве более одного раза.
4. Значения координат и направлений пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
5. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY I VT_DISPATCH.

GetNodeByPoint – Получить узел марки по координатам

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetNodeByPoint(double X, double Y, double Limit);

Синтаксис COM:

HRESULT GetNodeByPoint(double X, double Y, double Limit, IMarkNode ** PVal);

Входные параметры:

X, Y	- координаты точки,
Limit	- максимальное расстояние от точки до узла при поиске.

Возвращаемое значение:

Указатель на интерфейс IMarkNode	- если узел найден,
NULL	- в случае неудачи.

Примечание:

1. Значения координат и направлений пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType
2. Подключить узел можно только к точкам подключения, координаты которых возвращаются функциями IBuildingAxis::GetInsertionPoints и IMarkNode::GetInsertionPoints.
3. После добавления узла координаты и список доступных точек для подключения новых узлов может измениться.

Интерфейс IArcAxis

Справка системы КОМПАС...

KOMPAS.chm::/CM_ARC_AXIS.htm

Интерфейс дуговой координационной оси.

Иерархия:

IKompasAPIObject
- IDrawingObject
  - IBuildingAxis
    - IArcAxis

Описание:

Интерфейс позволяет получать и изменять свойства дуговой строительной оси.

Примечание:

1. Получить интерфейс строительной оси можно, используя методы коллекции строительных осей IBuildingAxes::BuildingAxis, IBuildingAxes::Add.
2. При создании дуговой оси можно использовать несколько способов построения.

▼ По координатам,

- По трем точкам,
- По углам,
- По точке и углу,
- По углу, точке и радиусу,
- По конечным точкам и углу,
- Плавающий центр, вариант 1,
- Плавающий центр, вариант 2.

3. После задания параметров оси требуется вызвать метод IDrawingObject::Update.
4. При изменении параметров зависимые параметры пересчитываются после вызова метода IDrawingObject::Update.

IArcAxis – свойства

Angle1 – Угол между осью 0X и отрезком, соединяющим первую точку дуги с началом координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle1 = iObject.Angle1;                Получить свойство(* )
iObject.Angle1 = Angle1;                Установить свойство (* )
Angle1 = iObject.GetAngle1();           Получить свойство (**)
iObject.SetAngle1( Angle1 );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle1( &Angle1 )          Получить свойство
iObject->put_Angle1( Angle1 )           Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол между осью 0X и отрезком, соединяющим первую точку дуги с началом координат.

Angle2 – Угол между осью 0X и отрезком, соединяющим вторую точку дуги с началом координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle2 = iObject.Angle2;                Получить свойство(* )
iObject.Angle2 = Angle2;                Установить свойство (* )
Angle2 = iObject.GetAngle2();           Получить свойство (**)
```

```
iObject.SetAngle2( Angle2 );                    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle2( &Angle2 )                  Получить свойство
iObject->put_Angle2( Angle2 )                   Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол между осью 0X и отрезком, соединяющим вторую точку дуги с началом координат.

Direction – Направление построения дуги

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = iObject.Direction;                  Получить свойство(* )
iObject.Direction = Direction;                  Установить свойство (* )
Direction = iObject.GetDirection();             Получить свойство (**)
iObject.SetDirection( Direction );              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Direction( &Direction );           Получить свойство
iObject->put_Direction( Direction );            Установить свойство
```

Значения свойства:

```
TRUE            - по часовой стрелке,
FALSE           - против часовой стрелки.
```

Примечание:

Свойство позволяет получить и установить направление построения дуги.

Jut – Получить выступ

Интерфейс...

Тип данных: указатель на интерфейс выступа IAxisJut

Синтаксис Automation:

```
Jut = iObject.Jut( First );                     Получить свойство(* )
Jut = iObject.GetJut( First );                  Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Jut(First, &Jut);          Получить свойство
```

Входные параметры:

First	- признак порядка выступа на оси, BOOL, TRUE- первый выступ, FALSE - второй.

Примечание:

Свойство доступно только для чтения.

MarkNodes – Коллекция узлов марок

Интерфейс...

Тип данных: указатель на интерфейс коллекции IMarkNodes

Синтаксис Automation:

```
MarkNodes = iObject.MarkNodes;          Получить свойство(* )
MarkNodes = iObject.GetMarkNodes();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_MarkNodes(                 Получить свойство
&MarkNodes );
```

Примечание:

1. Свойство доступно только для чтения.
2. Узлы правого и левого выступа имеют одинаковые параметры, поэтому редактирование осуществляется через одну коллекцию.

Radius – Радиус дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = iObject.Radius;                Получить свойство(* )
iObject.Radius = Radius;                Установить свойство (* )
Radius = iObject.GetRadius();           Получить свойство (**)
iObject.SetRadius( Radius );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Radius( &Radius );         Получить свойство
iObject->put_Radius( Radius );          Установить свойство
```

Примечание:

Свойство позволяет получить и установить радиус дуги.

Xc – Координата X центра дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = iObject.Xc;                    Получить свойство(* )
iObject.Xc = Xc;                    Установить свойство (* )
Xc = iObject.GetXc();               Получить свойство (**)
iObject.SetXc( Xc );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Xc( &Xc );             Получить свойство
iObject->put_Xc( Xc );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Xc.

X1 – Координата первой точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1;                    Получить свойство(* )
iObject.X1 = X1;                    Установить свойство (* )
X1 = iObject.GetX1();               Получить свойство (**)
iObject.SetX1( X1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1( &X1 );             Получить свойство
iObject->put_X1( X1 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги X1.

X2 – Координата второй точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2;                    Получить свойство(* )
iObject.X2 = X2;                    Установить свойство (* )
X2 = iObject.GetX2();               Получить свойство (**)
iObject.SetX2( X2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2( &X2 );             Получить свойство
iObject->put_X2( X2 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги X2.

X3 – Координата средней точки, лежащей на дуге по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X3 = iObject.X3;                    Получить свойство(* )
iObject.X3 = X3;                    Установить свойство (* )
X3 = iObject.GetX3();               Получить свойство (**)
iObject.SetX3( X3 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X3( &X3 );             Получить свойство
iObject->put_X3( X3 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату средней точки, лежащей на дуге по X.

Yc – Координата Y центра дуги

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = iObject.Yc;                    Получить свойство(* )
iObject.Yc = Yc;                    Установить свойство (* )
Yc = iObject.GetYc();               Получить свойство (**)
iObject.SetYc( Yc );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Yc( &Yc );                    Получить свойство
iObject->put_Yc( Yc );                     Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Yc.

Y1 – Координата первой точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1;                           Получить свойство(* )
iObject.Y1 = Y1;                           Установить свойство (* )
Y1 = iObject.GetY1();                      Получить свойство (**)
iObject.SetY1( Y1 );                       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1( &Y1 );                    Получить свойство
iObject->put_Y1( Y1 );                     Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки дуги Y1.

Y2 – Координата второй точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2;                           Получить свойство(* )
iObject.Y2 = Y2;                           Установить свойство (* )
Y2 = iObject.GetY2();                      Получить свойство (**)
iObject.SetY2( Y2 );                       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2( &Y2 );                    Получить свойство
iObject->put_Y2( Y2 );                     Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки дуги Y2.

Y3 – Координата средней точки, лежащей на дуге по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y3 = iObject.Y3;                    Получить свойство(* )
iObject.Y3 = Y3;                    Установить свойство (* )
Y3 = iObject.GetY3();               Получить свойство (**)
iObject.SetY3( Y3 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y3( &Y3 );             Получить свойство
iObject->put_Y3( Y3 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату средней точки, лежащей на дуге по Y.

Интерфейс ICircleAxis

Справка системы КОМПАС...

`KOMPAS.chm::/CM_CIRCLE_AXIS.htm`

Интерфейс круговой координационной оси.

Иерархия:

IKompasAPIObject
- IDrawingObject
  - IBuildingAxis
    - ICircleAxis

Описание:

Интерфейс позволяет получать и изменять свойства круговой строительной оси.

Примечание:

1. Получить интерфейс строительной оси можно, используя методы коллекции строительных осей IBuildingAxes::BuildingAxis, IBuildingAxes::Add.
2. При создании дуговой оси необходимо задать следующие параметры:

- координаты центра XC; YC,
- радиус Radius.

3. После задания параметров оси требуется вызвать метод IDrawingObject::Update.
4. При изменении параметров зависимые параметры пересчитываются после вызова метода IDrawingObject::Update.

ICircleAxis – свойства

BaseMarkNode – Базовый узел для марки

Интерфейс...

Тип данных: указатель на интерфейс IMarkNode

Синтаксис Automation:

MarkNode = iObject.BaseMarkNode;	Получить свойство(*)
MarkNode = iObject.GetBaseMarkNode();	Получить свойство (**)

Синтаксис COM:

iObject->get_BaseMarkNode(&MarkNode);	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Интерфейс IMarkNode позволяет задать параметры основного узла марки оси. Позволяет добавлять и удалять дополнительные узлы.

MarkAngle – Угол привязки марки

Интерфейс...

Тип данных: double

Синтаксис Automation:

MarkAngle = iObject.MarkAngle	Получить свойство(*)
iObject.MarkAngle = MarkAngle	Установить свойство (*)
MarkAngle = iObject.GetMarkAngle()	Получить свойство (**)
iObject.SetMarkAngle(MarkAngle)	Установить свойство (**)

Синтаксис COM:

iObject->get_MarkAngle(&MarkAngle)	Получить свойство
iObject->put_MarkAngle(MarkAngle)	Установить свойство

Примечание:

Свойство позволяет получить и установить угол между осью 0X и отрезком, соединяющим центр оси с маркой.

MarkOn – Марка вкючена\выключена

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
MarkOn = iObject.MarkOn                 Получить свойство(* )
iObject.MarkOn = MarkOn                 Установить свойство (* )
MarkOn = iObject.GetMarkOn()            Получить свойство (**)
iObject.SetMarkOn( MarkOn )             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_MarkOn(&MarkOn)            Получить свойство
iObject->put_MarkOn( MarkOn)            Установить свойство
```

Примечание:

Свойство позволяет управлять отображением марки на координационной оси.

Radius – Радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = iObject.Radius;                Получить свойство(* )
iObject.Radius = Radius;                Установить свойство (* )
Radius = iObject.GetRadius();           Получить свойство (**)
iObject.SetRadius( Radius );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Radius( &Radius );         Получить свойство
iObject->put_Radius( Radius );          Установить свойство
```

Примечание:

Свойство позволяет получить и установить радиус дуги.

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = iObject.Xc;                        Получить свойство(* )
iObject.Xc = Xc;                        Установить свойство (* )
Xc = iObject.GetXc();                   Получить свойство (**)
iObject.SetXc( Xc );                    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Xc( &Xc );                    Получить свойство
iObject->put_Xc( Xc );                     Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Xc.

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = iObject.Yc;                           Получить свойство(* )
iObject.Yc = Yc;                           Установить свойство (* )
Yc = iObject.GetYc();                      Получить свойство (**)
iObject.SetYc( Yc );                       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Yc( &Yc );                    Получить свойство
iObject->put_Yc( Yc );                     Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Yc.

Интерфейс IStraightAxis

Справка системы КОМПАС...

`KOMPAS.chm::/CM_DIRECT_AXIS.htm`

Интерфейс прямой координационной оси.

Иерархия:

IKompasAPIObject

IDrawingObject

IBuildingAxis

IStraightAxis

Описание:

Интерфейс позволяет получать и изменять свойства прямой строительной оси.

Примечание:

1. Получить интерфейс строительной оси можно, используя методы коллекции строительных осей IBuildingAxes::BuildingAxis, IBuildingAxes::Add.
2. При создании прямой оси можно использовать несколько типов построения:
   2.1. по координатам (требуется задать координаты точек X1; Y1, X2; Y2).

2.2. По точке 1 (X1; Y1), длине Length и углу Angle,

2.3. По точке 2 (X2; Y2), длине Length и углу Angle относительно точки 1.

3. После задания параметров оси требуется вызвать метод IDrawingObject::Update.
4. При изменении параметров зависимые параметры пересчитываются после вызова метода IDrawingObject::Update.

IStraightAxis – свойства

Angle – Угол наклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = iObject.Angle;                  Получить свойство(* )
iObject.Angle = Angle;                  Установить свойство (* )
Angle = iObject.GetAngle();             Получить свойство (**)
iObject.SetAngle( Angle );              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle(&Angle)              Получить свойство
iObject->put_Angle( Angle );            Установить свойство
```

Примечание:

Свойство позволяет получить и установить угол наклона отрезка.

Jut – Выступ

Интерфейс...

Тип данных: указатель на интерфейс IAxisJut

Синтаксис Automation:

```
Jut = iObject.Jut( First );             Получить свойство(* )
Jut = iObject.GetJut( First );          Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Jut(First, &Jut);          Получить свойство
```

Входные параметры:

First	-тип данных BOOL (TRUE - первый выступ, FALSE - второй).

Примечание:

Свойство доступно только для чтения.

Length – Длина отрезка

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = iObject.Length;                Получить свойство(* )
iObject.Length = Length;                Установить свойство (* )
Length = iObject.GetLength();           Получить свойство (**)
iObject.SetLength( Length );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Length(&Length);           Получить свойство
iObject->put_Length( Length );          Установить свойство
```

Примечание:

Свойство позволяет получить и установить длину отрезка.

MarkNodes – Коллекция узлов марок

Интерфейс...

Тип данных: указатель на интерфейс IMarkNodes

Синтаксис Automation:

```
MarkNodes = iObject.MarkNodes;          Получить свойство(* )
MarkNodes = iObject.GetMarkNodes();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_MarkNodes(                 Получить свойство
&MarkNodes );
```

Примечание:

1. Свойство доступно только для чтения.
2. Узлы правого и левого выступа имеют одинаковые параметры, поэтому редактирование осуществляется через одну коллекцию.

X1 – Координата первой точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1;                    Получить свойство(* )
iObject.X1 = X1;                    Установить свойство (* )
X1 = iObject.GetX1();               Получить свойство (**)
iObject.SetX1( X1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1(&X1);               Получить свойство
iObject->put_X1( X1 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки отрезка по оси X.

X2 – Координата второй точки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2;                    Получить свойство(* )
iObject.X2 = X2;                    Установить свойство (* )
X2 = iObject.GetX2();               Получить свойство (**)
iObject.SetX2( X2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2(&X2);               Получить свойство
iObject->put_X2( X2 );              Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки отрезка по оси X.

Y1 – Координата первой точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1;                    Получить свойство(* )
iObject.Y1 = Y1;                    Установить свойство (* )
Y1 = iObject.GetY1();               Получить свойство (**)
iObject.SetY1( Y1 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1(&Y1);          Получить свойство
iObject->put_Y1( Y1 );         Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату первой точки отрезка по оси Y.

Y2 – Координата второй точки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2;               Получить свойство(* )
iObject.Y2 = Y2;               Установить свойство (* )
Y2 = iObject.GetY2();          Получить свойство (**)
iObject.SetY2( Y2 );           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y2(&Y2);          Получить свойство
iObject->put_Y2( Y2 );         Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату второй точки отрезка по оси Y.

Интерфейс IBrace

Справка системы КОМПАС...

`KOMPAS.chm::/CM_BRACE.htm`

Интерфейс фигурной скобки.

Иерархия:

IKompasAPIObject

IDrawingObject

IBrace

Описание:

Интерфейс позволяет получать и изменять свойства фигурных скобок.

Примечание:

Получить интерфейс марки можно, используя свойство интерфейса коллекции фигурных скобок IBraces::Brace или метод IBraces::Add.

IBrace – свойства

Alignment – Ориентация фигурной скобки

Интерфейс...

Тип данных: тип ориентации из перечисления ksAlignmentTypeEnum

Синтаксис Automation:

```
Alignment = iObject.Alignment          Получить свойство(* )
iObject.Alignment = Alignment          Установить свойство (* )
Alignment = iObject.GetAlignment()     Получить свойство (**)
iObject.SetAlignment( Alignment )      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Alignment(                Получить свойство
&Alignment )
iObject->put_Alignment(                Установить свойство
Alignment )
```

Angle – Угол наклона фигурной скобки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = iObject.Angle                  Получить свойство(* )
iObject.Angle = Angle                  Установить свойство (* )
Angle = iObject.GetAngle()             Получить свойство (**)
iObject.SetAngle( Angle )              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle( &Angle )           Получить свойство
iObject->put_Angle( Angle )            Установить свойство
```

Direction – Ориентация горлышка скобки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = iObject.Direction          Получить свойство(* )
iObject.Direction = Direction          Установить свойство (* )
Direction = iObject.GetDirection()     Получить свойство (**)
```

iObject.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

iObject->get_Direction(&Direction)	Получить свойство
iObject->put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- нормальная ориентация,
FALSE	- перевернутая.

Length – Длина фигурной скобки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Length = iObject.Length	Получить свойство(*)
iObject.Length = Length	Установить свойство (*)
Length = iObject.GetLength()	Получить свойство (**)
iObject.SetLength(Length)	Установить свойство (**)

Синтаксис COM:

iObject->get_Length(&Length)	Получить свойство
iObject->put_Length(Length)	Установить свойство

Radius – Радиус скругления скобки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = iObject.Radius	Получить свойство(*)
iObject.Radius = Radius	Установить свойство (*)
Radius = iObject.GetRadius()	Получить свойство (**)
iObject.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

iObject->get_Radius(&Radius)	Получить свойство
iObject->get_Radius(&Radius)	Установить свойство

ShelfDirection – Положение полки

Интерфейс...

Тип данных: Тип положения полки из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = iObject.ShelfDirection	Получить свойство(*)
iObject.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = iObject.GetShelfDirection()	Получить свойство (**)
iObject.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfDirection(&ShelfDirection)	Получить свойство
iObject->put_ShelfDirection(ShelfDirection)	Установить свойство

ShelfPoints – Массив SAFEARRAY координат точек полки

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

ShelfPoints = Object.ShelfPoints	Получить свойство(*)
Object.ShelfPoints = ShelfPoints	Установить свойство (*)
ShelfPoints = Object.GetShelfPoints()	Получить свойство (**)
Object.SetShelfPoints(ShelfPoints)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfPoints(&ShelfPoints)	Получить свойство
Object.put_ShelfPoints(ShelfPoints)	Установить свойство

ShelfX – Координата X начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = iObject.ShelfX	Получить свойство(*)
iObject.ShelfX = ShelfX	Установить свойство (*)
ShelfX = iObject.GetShelfX()	Получить свойство (**)
iObject.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

```
iObject->get_ShelfX( &ShelfX )          Получить свойство
iObject->put_ShelfX( ShelfX )           Установить свойство
```

Примечание:

При попытке получить координату полки, которой нет, свойство вернёт 0.0.

ShelfY – Координата Y начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfY = iObject.ShelfY                 Получить свойство(* )
iObject.ShelfY = ShelfY                 Установить свойство (* )
ShelfY = iObject.GetShelfY()            Получить свойство (**)
iObject.SetShelfY( ShelfY )             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_ShelfY( &ShelfY )          Получить свойство
iObject->put_ShelfY( ShelfY )           Установить свойство
```

Примечание:

При попытке получить координату полки, которой нет, свойство вернёт 0.0.

Style – Стиль линии фигурной скобки

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = iObject.Style                   Получить свойство(* )
iObject.Style = Style                   Установить свойство (* )
Style = iObject.GetStyle()              Получить свойство (**)
iObject.SetStyle( Style )               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Style( &Style )            Получить свойство
iObject->put_Style( Style )             Установить свойство
```

Text – Текст фигурной скобки

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

```
Text = iObject.Text                    Получить свойство(* )
Text = iObject.GetText()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Text( &Text )             Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Текст можно использовать для поиска фигурной скобки в коллекции IBraces. Для поиска в функции IDrawingObjects::Item или IBraces::Brace нужно передать строковое значение текста.

Пример:

Brace = iBraces.Brace("Text");

где "Text" - текст на фигурной скобке;

X1 – Координата первой точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X1 = iObject.X1;                       Получить свойство(* )
iObject.X1 = X1;                       Установить свойство (* )
X1 = iObject.GetX1();                  Получить свойство (**)
iObject.SetX1( X1 );                   Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X1(&X1);                  Получить свойство
iObject->put_X1( X1 );                 Установить свойство
```

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

X2 – Координата второй точки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X2 = iObject.X2;                    Получить свойство(* )
iObject.X2 = X2;                    Установить свойство (* )
X2 = iObject.GetX2();               Получить свойство (**)
iObject.SetX2( X2 );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X2(&X2);               Получить свойство
iObject->put_X2( X2 );              Установить свойство
```

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Y1 – Координата первой точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y1 = iObject.Y1;                    Получить свойство(* )
iObject.Y1 = Y1;                    Установить свойство (* )
Y1 = iObject.GetY1();               Получить свойство (**)
iObject.SetY1(Y1);                  Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y1(&Y1);               Получить свойство
iObject->put_Y1(Y1);                Установить свойство
```

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Y2 – Координата второй точки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y2 = iObject.Y2;                    Получить свойство(* )
iObject.Y2 = Y2;                    Установить свойство (* )
Y2 = iObject.GetY2();               Получить свойство (**)
iObject.SetY2(Y2);                  Установить свойство (**)
```

Синтаксис COM:

iObject->get_Y2(&Y2);	Получить свойство
iObject->put_Y2(Y2);	Установить свойство

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Интерфейс ICutUnitMarking

Справка системы КОМПАС...

KOMPAS.chm::/CM_cutnode.htm

Интерфейс обозначения узла в сечении.

Иерархия:

IKompasAPIObject

IDrawingObject

ICutUnitMarking

Примечание:

Получить интерфейс можно, используя свойство интерфейса коллекции ICutUnitMarkings::CutUnitMarking или метод ICutUnitMarkings::Add.

ICutUnitMarking – свойства

Angle – Угол наклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle	Получить свойство(*)
iObject.Angle = Angle	Установить свойство (*)
Angle = iObject.GetAngle()	Получить свойство (**)
iObject.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle(&Angle)	Получить свойство
iObject->put_Angle(Angle)	Установить свойство

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = iObject.ShelfDirection	Получить свойство(*)
iObject.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = iObject.GetShelfDirection()	Получить свойство (**)
iObject.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfDirection(&ShelfDirection)	Получить свойство
iObject->put_ShelfDirection(ShelfDirection)	Установить свойство

ShelfX – Координата начала полки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = iObject.ShelfX	Получить свойство(*)
iObject.ShelfX = ShelfX	Установить свойство (*)
ShelfX = iObject.GetShelfX()	Получить свойство (**)
iObject.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfX(&ShelfX)	Получить свойство
iObject->put_ShelfX(ShelfX)	Установить свойство

ShelfY – Координата начала полки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfY = iObject.StrokeY	Получить свойство(*)
iObject.ShelfY = StrokeY	Установить свойство (*)
ShelfY = iObject.GetShelfY()	Получить свойство (**)
iObject.SetShelfY(ShelfY)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfY(&ShelfY)	Получить свойство
iObject->put_ShelfY(ShelfY)	Установить свойство

Stroke – Длина штриха

Интерфейс...

Тип данных: double

Синтаксис Automation:

Stroke = iObject.Stroke(Index)	Получить свойство(*)
iObject.Stroke(Index) = Stroke	Установить свойство (*)
Stroke = iObject.GetStroke(Index)	Получить свойство (**)
iObject.SetStroke(Index, Stroke)	Установить свойство (**)

Синтаксис COM:

iObject->get_Stroke(&Stroke)	Index,	Получить свойство
iObject->put_Stroke(Stroke)	Index,	Установить свойство

Входные параметры:

Index (long)	- индекс штриха.

StrokeCount – Число штрихов

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = iObject.StrokeCount	Получить свойство(*)
iObject.StrokeCount = Count	Установить свойство (*)
Count = iObject.GetStrokeCount()	Получить свойство (**)
iObject.SetStrokeCount(Count)	Установить свойство (**)

Синтаксис COM:

iObject->get_StrokeCount(&Count)	Получить свойство
iObject->put_StrokeCount(Count)	Установить свойство

StrokeX – Координата точки привязки штриха по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

StrokeX = iObject.StrokeX(Index)	Получить свойство(*)
iObject.StrokeX(Index) = StrokeX	Установить свойство (*)
StrokeX = iObject.GetStrokeX(Index)	Получить свойство (**)
iObject.SetStrokeX(Index, StrokeX)	Установить свойство (**)

Синтаксис COM:

iObject->get_StrokeX(&StrokeX)	Index,	Получить свойство
iObject->put_StrokeX(StrokeX)	Index,	Установить свойство

Входные параметры:

Index (long)	- индекс штриха.

StrokeY – Координата точки привязки штриха по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

StrokeY = iObject.StrokeY(Index)	Получить свойство(*)
iObject.StrokeY(Index) = StrokeY	Установить свойство (*)
StrokeY = iObject.GetStrokeY(Index)	Получить свойство (**)
iObject.SetStrokeY(Index, StrokeY)	Установить свойство (**)

Синтаксис COM:

iObject->get_StrokeY(&StrokeY)	Index,	Получить свойство
iObject->put_StrokeY(StrokeY)	Index,	Установить свойство

Входные параметры:

Index (long)	- индекс штриха.

TextDown – Текст внизу

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextDown = iObject.TextDown	Получить свойство(*)
TextDown = iObject.GetTextDown()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextDown(&TextDown)	Получить свойство

Примечание.

Свойство доступно только для чтения.

TextUp – Текст вверху

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUp = iObject.TextUp	Получить свойство(*)
TextUp = iObject.GetTextUp()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextUp(&TextUp)	Получить свойство

Описание.

Свойство **Текст вверху** можно использовать для поиска обозначения узла в сечении в коллекции ICutUnitMarkings. Для поиска в функции IDrawingObjects::Item или ICutUnitMarkings::CutUnitMarking нужно передать строковое значение текста.

Пример:

cutUnitMarking = iCutUnitMarkings.CutUnitMarking("A1");

где "A1" - Текст вверху

Примечание.

Свойство доступно только для чтения.

ICutUnitMarking – методы

AddStroke – Добавить штрих

Интерфейс...

Синтаксис Automation:

BOOL AddStoke(double X,
double Y,
double Length);

Синтаксис COM:

HRESULT AddStoke(double X,
double Y,

double Length,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки привязки штриха,
Length	- длина.

Возвращаемое значение:

TRUE	- штрих добавлен,
FALSE	- в случае неудачи.

Примечание:

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType

DeleteStroke – Удалить штрих

Интерфейс...

Синтаксис Automation:

BOOL DeleteStroke(long Index);

Синтаксис COM:

HRESULT AddStroke(long Index, BOOL * Result);

Входные параметры:

index	- номер удаляемого штриха.

Возвращаемое значение:

TRUE	- штрих добавлено,
FALSE	- в случае неудачи.

Интерфейс IMark

Справка системы КОМПАС...

`KOMPAS.chm::/CM_MARK_WOLDR_POSNUM.htm`

Марка – базовый интерфейс для марок.

Иерархия:

IKompasAPIObject

IDrawingObject

IMark

Описание:

Интерфейс позволяет получать и изменять общие свойства для марок.

Примечание:

Получить интерфейс марки можно, используя методы интерфейса коллекции марок IMarks::Mark и IMarks::Add.

IMark – свойства

AutoNumber – Автонумерация марок и позиционных обозначений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AutoNumber = iObject.AutoNumber                 Получить свойство(* )
iObject.AutoNumber = AutoNumber                 Установить свойство (* )
AutoNumber = iObject.GetAutoNumber()            Получить свойство (**)
iObject.SetAutoNumber(AutoNumber)               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_AutoNumber(&AutoNumber)            Получить свойство
iObject->put_AutoNumber( AutoNumber)            Установить свойство
```

Comment – Текст комментария

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Comment = iObject.Comment                       Получить свойство(* )
iObject.Comment = Comment                       Установить свойство (* )
Comment = iObject.GetComment()                  Получить свойство (**)
iObject.SetComment(Comment)                     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Comment(&Comment)                  Получить свойство
iObject->put_Comment( Comment)                  Установить свойство
```

Примечание:

Свойство позволяет получить и установить текст комментария.

Name – Имя марки

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

```
Name = iObject.Name                    Получить свойство(* )
Name = iObject.GetName()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name(&Name)               Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Имя марки и номер марки IMark::Number можно использовать для поиска марки в коллекции марок IMarks.
3. Для поиска в функции IDrawingObjects::Item или IMarks::Mark нужно передать суммарную строку, включающую в себя имя марки + номер марки без дополнительных разделителей.

Пример:

mark = iMarks.Mark("A1");

где "A" - имя марки

"1" - номер марки

Number – Номер марки

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

```
Number = iObject.Number                Получить свойство(* )
Number = iObject.GetNumber()           Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Number(&Number)           Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Имя марки IMark::Name и номер марки можно использовать для поиска марки в коллекции марок IMarks.
3. Для поиска в функции IDrawingObjects::Item или IMarks::Mark нужно передать суммарную строку, включающую в себя имя марки + номер марки без дополнительных разделителей.

Пример:

mark = iMarks.Mark("A1");

где "A" - имя марки

"1" - номер марки

TextAfter – Текст после

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextAfter = iObject.TextAfter	Получить свойство(*)
TextAfter = iObject.GetTextAfter()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextAfter(&TextAfter)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TextBefore – Текст до

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextBefore = iObject.TextBefore	Получить свойство(*)
TextBefore = iObject.GetTextBefore()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextBefore(&TextBefore)	Получить свойство

Примечание:

Свосто доступно только для чтения.

TextUnder – Текст под

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUnder = iObject.TextUnder	Получить свойство(*)
TextUnder = iObject.GetTextUnder()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextUnder(&TextUnder)	Получить свойство

Примечание:

Свойство доступно только для чтения.

X – Координата точки привязки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = iObject.X;	Получить свойство(*)
iObject.X = X;	Установить свойство (*)
X = iObject.GetX();	Получить свойство (**)
iObject.SetX(X);	Установить свойство (**)

Синтаксис COM:

iObject->get_X(&X);	Получить свойство
iObject->put_X(X);	Установить свойство

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Y – Координата точки привязки марки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = iObject.Y;	Получить свойство(*)
iObject.Y = Y;	Установить свойство (*)
Y = iObject.GetY();	Получить свойство (**)
iObject.SetY(Y);	Установить свойство (**)

Синтаксис COM:

iObject->get_Y(&Y);	Получить свойство
iObject->put_Y(Y);	Установить свойство

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Интерфейс IMarkOnLeader

Справка системы КОМПАС...

KOMPAS.chm::/280_29_4_Marka_pozicionnoe_oboz.htm

Интерфейс марки и позиционного обозначения (с линией-выноской).

Иерархия:

IKompasAPIObject

IDrawingObject

IMark

IMarkOnLeader

Примечание:

Получить интерфейс марки можно, используя методы интерфейса коллекции марок IMarks::Mark и IMarks::Add.

IMarkOnLeader – свойства

ArrowType – Тип стрелки линии-выноски

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = iObject.ArrowType	Получить свойство(*)
iObject.ArrowType = ArrowType	Установить свойство (*)
ArrowType = iObject.GetArrowType()	Получить свойство (**)
iObject.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

iObject->get_ArrowType(&ArrowType)	Получить свойство
iObject->put_ArrowType(ArrowType)	Установить свойство

BranchBegin – Начало ответвления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
BranchBegin = iObject.BranchBegin                  Получить свойство(* )
iObject.BranchBegin = BranchBegin                  Установить свойство (* )
BranchBegin = iObject.GetBranchBegin()             Получить свойство (**)
iObject.SetBranchBegin(BranchBegin)                Установить свойство (**)
```

Синтаксис COM:

```
iObject-                                           Получить свойство
>get_BranchBegin(&BranchBegin)
iObject->put_BranchBegin( BranchBegin)             Установить свойство
```

Значения свойства:

```
TRUE                                               - от начала полки,
FALSE                                              - от конца полки.
```

Примечание.

Свойство задает признак начала ответвления.

BranchCount – Число ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
BranchCount = iObject.BranchCount                  Получить свойство(* )
BranchCount = iObject.GetBranchCount()             Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_BranchCount(&BranchCount)             Получить свойство
```

Примечание.

Свойство доступно только для чтения.

BranchPoints – Массив SAFEARRAY координат точек ответвления

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

BranchPoints = iObject.BranchPoints(index)	Получить свойство(*)
iObject.BranchPoints(index) = BranchPoints	Установить свойство (*)
BranchPoints = iObject.GetBranchPoints(index)	Получить свойство (**)
iObject.SetBranchPoints(index, BranchPoints)	Установить свойство (**)

Синтаксис COM:

iObject->get_BranchPoints(index, &BranchPoints)	Получить свойство
iObject->put_BranchPoints(index, BranchPoints)	Установить свойство

Примечание.

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, x1, y1, ...xi, yi.
2. Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

BranchPointsCount – Число точек в ответвлении

Интерфейс...

Тип данных: double

Синтаксис Automation:

Count = iObject.BranchPointsCount	Получить свойство(*)
Count = iObject.GetBranchPointsCount()	Получить свойство (**)

Синтаксис COM:

iObject->get_BranchPointsCount(&Count)	Получить свойство

Примечание.

Свойство доступно только для чтения.

BranchX – Координата конечной точки ответвления по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchX = iObject.BranchX	Получить свойство(*)
iObject.BranchX = BranchX	Установить свойство (*)
BranchX = iObject.GetBranchX()	Получить свойство (**)
iObject.SetBranchX(BranchX)	Установить свойство (**)

Синтаксис COM:

iObject->get_BranchX(&BranchX)	Получить свойство
iObject->put_BranchX(BranchX)	Установить свойство

Примечание.

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

BranchY – Координата конечной точки ответвления по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchY = iObject.BranchY	Получить свойство(*)
iObject.BranchY = BranchY	Установить свойство (*)
BranchY = iObject.GetBranchY()	Получить свойство (**)
iObject.SetBranchY(BranchY)	Установить свойство (**)

Синтаксис COM:

iObject->get_BranchY(&BranchY)	Получить свойство
iObject->put_BranchY(BranchY)	Установить свойство

Примечание.

Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

ParallelBranch – Параллельное ответвление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ParallelBranch = iObject.ParallelBranch	Получить свойство(*)
iObject.ParallelBranch = ParallelBranch	Установить свойство (*)
ParallelBranch = iObject.GetParallelBranch()	Получить свойство (**)
iObject.SetParallelBranch(ParallelBranch)	Установить свойство (**)

Синтаксис COM:

iObject->get_ParallelBranch(&ParallelBranch)	Получить свойство
iObject->put_ParallelBranch(ParallelBranch)	Установить свойство

Значения свойства:

TRUE	- строятся параллельные ответвления,
FALSE	- строятся непараллельные ответвления.

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = iObject.ShelfDirection	Получить свойство(*)
iObject.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = iObject.GetShelfDirection()	Получить свойство (**)
iObject.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfDirection(&ShelfDirection)	Получить свойство
iObject->put_ShelfDirection(ShelfDirection)	Установить свойство

IMarkOnLeader – методы

AddBranch – Добавить ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranch(long Index,
BOOL Begin,
VARIANT Points);

Синтаксис COM:

HRESULT AddBranch(long Index,
BOOL Begin,
VARIANT Points,
BOOL * Result);

Входные параметры:

index	- индекс, с которым добавляется ответвление,
Begin	- признак начала ответвления TRUE - от начала полки, FALSE - от конца полки,
Points	- массив SafeArray координат точек.

Возвращаемое значение:

TRUE	- ответвление добавлено,
FALSE	- в случае неудачи.

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, x1, y1, ...xi, yi.
2. Значения координат зависят от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType

AddBranchByPoint – Добавить прямолинейное ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranchByPoint(long Index,
BOOL Begin,
double x, double y);

Синтаксис COM:

HRESULT AddBranchByPoint(long Index,
BOOL Begin,
double x,
double y,
BOOL * Result);

Входные параметры:

index	- индекс, с которым добавляется ответвление,
Begin	- признак начала ответвления TRUE - от начала полки, FALSE - от конца полки,
x, y	- координаты конца ответвления.

Возвращаемое значение:

TRUE	- ответвление добавлено,
FALSE	- в случае неудачи.

Примечание:

Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType

DeleteBranch – Удалить ответвление

Интерфейс...

Синтаксис Automation:

BOOL DeleteBranch(long Index)

Синтаксис COM:

HRESULT DeleteBranch([in]long Index, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

index	- индекс удаляемого ответвление.

Возвращаемое значение:

TRUE	- ответвление удалено,
FALSE	- в случае неудачи.

Интерфейс IMarkInsideForm

Справка системы КОМПАС...

KOMPAS.chm::/273_29_3_Marka_pozicionnoe_oboz.htm

Интерфейс марки и позиционного обозначения (без линии-выноски).

Иерархия:

IKompasAPIObject

IDrawingObject

IMark

IMarkInsideForm

Примечание:

Получить интерфейс марки можно, используя методы интерфейса коллекции марок IMarks::Mark и IMarks::Add.

IMarkInsideForm – свойства

Angle – Угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle	Получить свойство(*)
iObject.Angle = Angle	Установить свойство (*)
Angle = iObject.GetAngle()	Получить свойство (**)
iObject.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle(&Angle)	Получить свойство
iObject->put_Angle(Angle)	Установить свойство

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Form – Форма

Интерфейс...

Тип данных: из перечисления ksMarkInsideFormEnum

Синтаксис Automation:

Form = iObject.Form	Получить свойство(*)
iObject.Form = Form	Установить свойство (*)
Form = iObject.GetForm()	Получить свойство (**)
iObject.SetForm(Form)	Установить свойство (**)

Синтаксис COM:

iObject->get_Form(&Form)	Получить свойство
iObject->put_Form(Form)	Установить свойство

FormGabarit – Габарит формы (радиус, ширина)

Интерфейс...

Тип данных: double

Синтаксис Automation:

FormGabarit = iObject.FormGabarit	Получить свойство(*)
iObject.FormGabarit = FormGabarit	Установить свойство (*)
FormGabarit = iObject.GetFormGabarit()	Получить свойство (**)
iObject.SetFormGabarit(FormGabarit)	Установить свойство (**)

Синтаксис COM:

iObject->get_FormGabarit(&FormGabarit)	Получить свойство
iObject->put_FormGabarit(FormGabarit)	Установить свойство

Примечание:

Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

FormHeight – Высота формы

Интерфейс...

Тип данных: double

Синтаксис Automation:

FormHeight = iObject.FormHeight	Получить свойство(*)

iObject.FormHeight = FormHeight	Установить свойство (*)
FormHeight = iObject.GetFormHeight()	Получить свойство (**)
iObject.SetFormHeight(FormHeight)	Установить свойство (**)

Синтаксис COM:

iObject->get_FormHeight(&FormHeight)	Получить свойство
iObject->put_FormHeight(FormHeight)	Установить свойство

FormStyle – Стиль линий формы

Интерфейс...

Тип данных: long

Синтаксис Automation:

FormStyle = iObject.FormStyle	Получить свойство(*)
iObject.FormStyle = FormStyle	Установить свойство (*)
FormStyle = iObject.GetFormStyle()	Получить свойство (**)
iObject.SetFormStyle(FormStyle)	Установить свойство (**)

Синтаксис COM:

iObject->get_FormStyle(&FormStyle)	Получить свойство
iObject->put_FormStyle(FormStyle)	Установить свойство

Примечание:

Стиль формы позволяет задать стиль линий, используемых для отрисовки формы:

Стандартные стили ksCurveStyleEnum, используемые для отрисовки формы марки:

ksCSNormal	1	Основная,
ksCSThin	2	Тонкая,
ksCSAxial	3	Осевая.

Допустимо задавать и другие стили, в том числе и пользовательские.

Интерфейс IMarkOnLine

Справка системы КОМПАС...

KOMPAS.chm::/284_29_5_Marka_pozicionnoe_oboz.htm

Интерфейс марки и позиционного обозначения (на линии).

Иерархия:

IKompasAPIObject

IDrawingObject

IMark

IMarkOnLine

Примечание:

Получить интерфейс марки можно, используя методы интерфейса коллекции марок IMarks::Mark и IMarks::Add.

IMarkOnLine – свойства

Line – Линия, с которой связывается марка

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

Line = iObject.Line	Получить свойство(*)
iObject.Line = Line	Установить свойство (*)
Line = iObject.GetLine()	Получить свойство (**)
iObject.SetLine(Line)	Установить свойство (**)

Синтаксис COM:

iObject->get_Line(&Line)	Получить свойство
iObject->put_Line(Line)	Установить свойство

Position – Положение марки относительно линии

Интерфейс...

Тип данных: из перечисления ksMarkOnLinePosTypeEnum

Синтаксис Automation:

Position = iObject.Position	Получить свойство(*)
iObject.Position = Position	Установить свойство (*)
Position = iObject.GetPosition()	Получить свойство (**)
iObject.SetPosition(Position)	Установить свойство (**)

Синтаксис COM:

iObject->get_Position(&Position)	Получить свойство
iObject->put_Position(Position)	Установить свойство

Интерфейс IMultiTextLeader

Интерфейс выносной надписи к многослойным конструкциям.

Иерархия:

IKompasAPIObject

IDrawingObject

IMultiTextLeader

Описание:

Интерфейс позволяет получить доступ к выносной надписи к многослойным конструкциям. С помощью интерфейса можно получить и отредактировать параметры объекта.

Примечание:

1. Получить интерфейс можно, используя свойство интерфейса коллекции IMultiTextLeaders::MultiTextLeader или метод IMultiTextLeaders::Add.
2. Новые параметры вступят в силу после вызова метода IDrawingObject::Update.

IMultiTextLeader – свойства

Align – Выравнивание

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

VFormat = iObject.VFormat;	Получить свойство (*)
iObject.VFormat = VFormat;	Установить свойство (*)
VFormat = iObject.GetVFormat( );	Получить свойство (**)
iObject.SetVFormat(VFormat);	Установить свойство (**)

Синтаксис COM:

iObject->get_VFormat(&VFormat);	Получить свойство
iObject->put_VFormat(VFormat);	Установить свойство

Значения свойства:

TRUE	- полки одинаковой длины,
FALSE	- полки по длине текста.

Примечание:

Свойство позволяет установить и получить выравнивание полок.

ArrowType – Тип стрелки линии-выноски

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = iObject.ArrowType	Получить свойство(*)
iObject.ArrowType = ArrowType	Установить свойство (*)
ArrowType = iObject.GetArrowType( )	Получить свойство (**)
iObject.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

iObject->get_ArrowType(&ArrowType)	Получить свойство
iObject->put_ArrowType(ArrowType)	Установить свойство

Примечание:

Свойство позволяет установить и получить тип стрелки линии-выноски.

BranchCount – Количество ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchCount = iObject.BranchCount	Получить свойство (*)
BranchCount = iObject.GetBranchCount( )	Получить свойство (**)

Синтаксис COM:

iObject->get_BranchCount(&BranchCount)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить количество ответвлений.

BranchPoints – Массив SAFEARRAY координат точек ответвления

Интерфейс...

Тип данных: VARIANT (VT_ARRAY | VT_R8)

Синтаксис Automation:

BranchPoints = iObject.BranchPoints	Получить свойство (*)
BranchPoints = iObject.GetBranchPoints( )	Получить свойство (**)

Синтаксис COM:

iObject->get_BranchPoints(&BranchPoints)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить SAFEARRAY координат точек ответвления. В массиве последовательно собраны координаты X и Y для каждой точки ответвления: x0, y0, x1, y1, ...xi, yi.

BranchPointsCount – Количество точек в ответвлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchPointsCount = iObject.BranchPointsCount	Получить свойство (*)
iObject.GetBranchPointsCount( )	Получить свойство (**)

Синтаксис COM:

iObject->get_BranchPointsCount(&BranchPointsCount)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить количество точек в ответвлении.

BranchX – Координата конечной точки ответвления по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchX = iObject.BranchX(Index)	Получить свойство (*)
iObject.BranchX = BranchX(Index)	Установить свойство (*)
BranchX = iObject.GetBranchX(Index)	Получить свойство (**)
iObject.SetBranchX(Index, BranchX)	Установить свойство (**)

Синтаксис COM:

iObject->get_BranchX(Index, &BranchX)	Получить свойство
iObject->put_BranchX(Index, BranchX)	Установить свойство

Входные параметры:

Index	- индекс объекта (long).

Примечание:

1. Свойство позволяет установить и получить координату конечной точки ответвления по оси X.
2. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

BranchY – Координата конечной точки ответвления по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

BranchY = iObject.BranchY(Index)	Получить свойство (*)
iObject.BranchY = BranchY(Index)	Установить свойство (*)
BranchY = iObject.GetBranchY(Index)	Получить свойство (**)
iObject.SetBranchY(Index, BranchY)	Установить свойство (**)

Синтаксис COM:

iObject->get_BranchY(Index, &BranchY)	Получить свойство
iObject->put_BranchY(Index, BranchY)	Установить свойство

Входные параметры:

Index	- индекс объекта (long).

Примечание:

1. Свойство позволяет установить и получить координату конечной точки ответвления по оси Y.
2. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

Form – Тип формы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Form = iObject.Form	Получить свойство (*)
iObject.Form = Form	Установить свойство (*)
Form = iObject.GetForm( )	Получить свойство (**)
iObject.SetForm(Form)	Установить свойство (**)

Синтаксис COM:

iObject->get_Form(&Form)	Получить свойство
iObject->put_Form(Form)	Установить свойство

Значения свойства:

TRUE	- с выступом,
FALSE	- без выступа.

Примечание:

Свойство позволяет установить и получить тип формы.

ShelfX – Координата начала полки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = iObject.ShelfX	Получить свойство (*)
iObject.ShelfX = ShelfX	Установить свойство (*)
ShelfX = iObject.GetShelfX( )	Получить свойство (**)
iObject.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfX(&ShelfX)	Получить свойство
iObject->put_ShelfX(ShelfX)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить координату начала полки по X.
2. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

ShelfY – Координата начала полки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfY = iObject.ShelfY	Получить свойство (*)
iObject.ShelfY = ShelfY	Установить свойство (*)
ShelfY = iObject.GetShelfY( )	Получить свойство (**)
iObject.SetShelfY(ShelfY)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfY(&ShelfY)	Получить свойство
iObject->put_ShelfY(ShelfY)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить координату начала полки по Y.
2. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = iObject.ShelfDirection	Получить свойство (*)
iObject.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = iObject.GetShelfDirection( )	Получить свойство (**)
iObject.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfDirection(&ShelfDirection)	Получить свойство
iObject->put_ShelfDirection(ShelfDirection)	Установить свойство

Значения свойства:

ksLSLeft	-1	- влево,
ksLSRight	1	- вправо.

Примечание:

Свойство позволяет установить и получить направление полки.

TextDirection – Направление текста

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TextDirection = iObject.TextDirection	Получить свойство (*)
iObject.TextDirection = TextDirection	Установить свойство (*)
TextDirection = iObject.GetTextDirection( )	Получить свойство (**)
iObject.SetTextDirection(TextDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_TextDirection(&TextDirection)	Получить свойство
iObject->put_TextDirection(TextDirection)	Установить свойство

Значения свойства:

TRUE	- вверх,
FALSE	- вниз.

Примечание:

Свойство позволяет установить и получить направление текста.

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = iObject.Text	Получить свойство (*)
Text = iObject.GetText( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Text(&Text)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить текст объекта.

IMultiTextLeader – методы

AddBranch – Добавить ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranch(long index,
VARIANT Points);

Синтаксис COM:

HRESULT AddBranch(long index,
VARIANT Points,
BOOL * Result);

Входные параметры:

index	- индекс, с которым добавляется ответвление,
Points	- массив SafeArray координат точек.

Возвращаемое значение:

TRUE	- ответвление добавлено,
FALSE	- в случае неудачи.

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, x1, y1, ...xi, yi.
2. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
3. Если index = -1, ответвление добавляется в конец массива ответвлений.

AddBranchByPoint – Добавить прямолинейное ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranchByPoint(long index,
double x, double y);

Синтаксис COM:

HRESULT AddBranchByPoint(long index,
double x,
double y,
BOOL * Result);

Входные параметры:

index	- индекс, с которым добавляется ответвление,
x, y	- координаты конца ответвления.

Возвращаемое значение:

TRUE	- ответвление добавлено,
FALSE	- в случае неудачи.

Примечание:

1. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
2. Если index = -1, ответвление добавляется в конец массива ответвлений.

DeleteBranch – Удалить ответвление

Интерфейс...

Синтаксис Automation:

BOOL DeleteBranch(long index);

Синтаксис COM:

HRESULT DeleteBranch(long Index,
VARIANT_BOOL * Result);

Входные параметры:

index	- индекс удаляемого ответвления.

Возвращаемое значение:

TRUE	- ответвление удалено,

FALSE	- в случае неудачи.

Примечание:

1. Значения координат пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
2. Если index = -1, ответвление добавляется в конец массива ответвлений.

Интерфейс IUnitMarking

Справка системы КОМПАС...

KOMPAS.chm::/CM_SIGNNODE.htm

Интерфейс обозначения узла.

Иерархия:

IKompasAPIObject

IDrawingObject

IUnitMarking

Примечание:

Получить интерфейс можно, используя свойство IUnitMarkings::UnitMarking или метод IUnitMarkings::Add интерфейса коллекции обозначений узла.

IUnitMarking – свойства

Height – Высота контура

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = iObject.Height	Получить свойство(*)
iObject.Height = Height	Установить свойство (*)
Height = iObject.GetHeight()	Получить свойство (**)
iObject.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

iObject->get_Height(&Height)	Получить свойство
iObject->put_Height(Height)	Установить свойство

FilletRadius – Радиус скругления прямоугольного контура

Интерфейс...

Тип данных: double

Синтаксис Automation:

FilletRadius = iObject.FilletRadius	Получить свойство(*)
iObject.FilletRadius = FilletRadius	Установить свойство (*)
FilletRadius = iObject.GetFilletRadius()	Получить свойство (**)
iObject.SetFilletRadius(FilletRadius)	Установить свойство (**)

Синтаксис COM:

iObject->get_FilletRadius(&FilletRadius)	Получить свойство
iObject->put_FilletRadius(FilletRadius)	Установить свойство

Form – Форма контура

Интерфейс...

Тип данных: тип формы из перечисления ksContourFormEnum

Синтаксис Automation:

Form = iObject.Form	Получить свойство(*)
iObject.Form = Form	Установить свойство (*)
Form = iObject.GetForm()	Получить свойство (**)
iObject.SetForm(Form)	Установить свойство (**)

Синтаксис COM:

iObject->get_Form(&Form)	Получить свойство
iObject->put_Form(Form)	Установить свойство

Radius – Радиус контура обозначения узла

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = iObject.Radius	Получить свойство(*)
iObject.Radius = Radius	Установить свойство (*)
Radius = iObject.GetRadius()	Получить свойство (**)
iObject.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

iObject->get_Radius(&Radius)	Получить свойство
iObject->put_Radius(Radius)	Установить свойство

ShelfDirection – Направление полки

Интерфейс...

Тип данных: тип направления полки из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = iObject.ShelfDirection	Получить свойство(*)
iObject.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = iObject.GetShelfDirection()	Получить свойство (**)
iObject.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfDirection(&ShelfDirection)	Получить свойство
iObject->put_ShelfDirection(ShelfDirection)	Установить свойство

ShelfX – Координата начала полки по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfX = iObject.ShelfX	Получить свойство(*)
iObject.ShelfX = ShelfX	Установить свойство (*)
ShelfX = iObject.GetShelfX()	Получить свойство (**)
iObject.SetShelfX(ShelfX)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfX(&ShelfX)	Получить свойство
iObject->put_ShelfX(ShelfX)	Установить свойство

ShelfY – Координата начала полки по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfY = iObject.ShelfY	Получить свойство(*)
iObject.ShelfY = ShelfY	Установить свойство (*)
ShelfY = iObject.GetShelfY()	Получить свойство (**)
iObject.SetShelfY(ShelfY)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShelfY(&ShelfY)	Получить свойство
iObject->put_ShelfY(ShelfY)	Установить свойство

TextDown – Текст внизу

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextDown = iObject.TextDown	Получить свойство(*)
TextDown = iObject.GetTextDown()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextDown(&TextDown)	Получить свойство

Примечание.

Свойство доступно только для чтения.

TextUp – Текст вверху

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TextUp = iObject.TextUp	Получить свойство(*)
TextUp = iObject.GetTextUp()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextUp(&TextUp)	Получить свойство

Описание.

Свойство **Текст вверху** можно использовать для поиска обозначения узла в коллекции IUnitMarkings. Для поиска в функции IDrawingObjects::Item или IUnitMarkings::UnitMarking нужно передать строковое значение текста.

Пример:

UnitMarking = IUnitMarkings.UnitMarking("A1");

где "A1" - Текст вверху

Примечание.

Свойство доступно только для чтения.

Width – Ширина контура

Интерфейс...

Тип данных: double

Синтаксис Automation:

Width = iObject.Width	Получить свойство(*)
iObject.Width = Width	Установить свойство (*)
Width = iObject.GetWidth()	Получить свойство (**)
iObject.SetWidth(Width)	Установить свойство (**)

Синтаксис COM:

iObject->get_Width(&Width)	Получить свойство
iObject->put_Width(Width)	Установить свойство

Xc – Координата X центра контура

Интерфейс...

Тип данных: double

Синтаксис Automation:

Xc = iObject.Xc	Получить свойство(*)
iObject.Xc = Xc	Установить свойство (*)
Xc = iObject.GetXc()	Получить свойство (**)
iObject.SetXc(Xc)	Установить свойство (**)

Синтаксис COM:

iObject->get_Xc(&Xc)	Получить свойство
iObject->put_Xc(Xc)	Установить свойство

Yc – Координата Y центра контура

Интерфейс...

Тип данных: double

Синтаксис Automation:

Yc = iObject.Yc	Получить свойство(*)
iObject.Yc = Yc	Установить свойство (*)
Yc = iObject.GetYc()	Получить свойство (**)
iObject.SetYc(Yc)	Установить свойство (**)

Синтаксис COM:

iObject->get_Yc(&Yc)	Получить свойство
iObject->put_Yc(Yc)	Установить свойство

Интерфейс IRemoteElement

Справка системы КОМПАС...

KOMPAS.chm::/CM_REMOTE_ELEMENT.htm

Интерфейс выносного элемента.

Иерархия:

IDispatch

IKompasAPIObject

IDrawingObject

IUnitMarking

IRemoteElement

Описание:

Интерфейс позволяет задавать параметры выносного элемента.

Примечание:

1. Интерфейс можно получить у коллекции выносных элементов, используя свойство IRemoteElements::RemoteElement или метод IRemoteElements::Add.
2. После задания параметров выносного элемента требуется вызвать метод IDrawingObject::Update.

IRemoteElement – свойства

AdditionalText – Дополнительный текст

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

AdditionalText = iObject.AdditionalText (Index);	Получить свойство(*)
AdditionalText = iObject.GetAdditionalText(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalText(Index, &AdditionalText)	Получить свойство

Примечание:

1. Свойство позволяет получить дополнительный текст выносного элемента.
2. Свойство доступно только для чтения.

AutoSheet – Лист

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSheet = Object.AutoSheet	Получить свойство(*)
Object.AutoSheet = AutoSheet	Установить свойство (*)
AutoSheet = Object.GetAutoSheet()	Получить свойство (**)
Object.SetAutoSheet(AutoSheet)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSheet(&AutoSheet)	Получить свойство
Object.put_AutoSheet(AutoSheet)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак включения номера листа в обозначение выносного элемента.

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство(*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автосортировки.

AutoZone – Зона

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoZone = Object.AutoZone	Получить свойство(*)
Object.AutoZone = AutoZone	Установить свойство (*)
AutoZone = Object.GetAutoZone()	Получить свойство (**)
Object.SetAutoZone(AutoZone)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoZone(&AutoZone)	Получить свойство
Object.put_AutoZone(AutoZone)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак включения обозначения зоны в обозначение выносного элемента.

Интерфейс IUnitNumber

Справка системы КОМПАС...

`KOMPAS.chm::/CM_KNOTNUMBER.htm`

Интерфейс номера узла.

Иерархия:

IKompasAPIObject

IDrawingObject

IUnitNumber

Описание:

Интерфейс позволяет получать и изменять свойства номеров узлов на чертеже.

Примечание:

Интерфейс можно получить с помощью метода коллекции номеров узла IUnitNumbers::Add или свойства IUnitNumbers::UnitNumber.

IUnitNumber – свойства

TextDown – Текст снизу

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

TextDown = iObject.TextDown	Получить свойство(*)
TextDown = iObject.TextDown()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextDown(&TextDown)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TextUp – Текст сверху

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

TextUp = iObject.TextUp	Получить свойство(*)
TextUp = iObject.TextUp()	Получить свойство (**)

Синтаксис COM:

iObject->get_TextUp(&TextUp)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Текст можно использовать для поиска номера узла в коллекции IUnitNumbers. Для поиска в функции IDrawingObjects::Item или IUnitNumbers::UnitNumber нужно передать строковое значение текста.

Пример:

UnitNumber = iUnitNumbers.UnitNumber("TextUp"),

где "TextUp" - текст сверху номера узла.

X – Координата номера узла по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = iObject.X;	Получить свойство(*)
iObject.X = X;	Установить свойство (*)
X = iObject.GetX();	Получить свойство (**)

iObject.SetX(X);	Установить свойство (**)

Синтаксис COM:

iObject->get_X(&X);	Получить свойство
iObject->put_X(X);	Установить свойство

Примечание:

1. Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
2. Свойство позволяет считывать и устанавливать координату номера узла по X.

Y – Координата номера узла по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = iObject.Y;	Получить свойство(*)
iObject.Y = Y;	Установить свойство (*)
Y = iObject.GetY();	Получить свойство (**)
iObject.SetY(Y);	Установить свойство (**)

Синтаксис COM:

iObject->get_Y(&Y);	Получить свойство
iObject->put_Y(Y);	Установить свойство

Примечание:

1. Значение параметра зависит от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType.
2. Свойство позволяет считывать и устанавливать координату номера узла по Y.

Интерфейс IAxisJut

Справка системы КОМПАС...

`KOMPAS.chm::/304_29_11_4_Vystupy.htm`

Справка системы КОМПАС...

`KOMPAS.chm::/CM_DIRECT_AXIS.htm`

Интерфейс выступа строительной оси.

Иерархия:

IDispatch

IKompasAPIObject

IAxisJut

Описание:

Интерфейс позволяет задавать параметры выступов для прямой и дуговой строительных осей.

Примечание:

Интерфейс можно получить с помощью свойств IStraightAxis::Jut и IArcAxis::Jut.

IAxisJut – свойства

BreakDirection – Направление смещения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BreakDirection = iObject.BreakDirection	Получить свойство(*)
iObject.BreakDirection = BreakDirection	Установить свойство (*)
BreakDirection = iObject.GetBreakDirection ()	Получить свойство (**)
iObject.SetBreakDirection (BreakDirection)	Установить свойство (**)

Синтаксис COM:

iObject->get_BreakDirection(&BreakDirection)	Получить свойство
iObject->put_BreakDirection(BreakDirection)	Установить свойство

Значения свойства:

TRUE	- смещение вправо,
FALSE	- смещение влево.

BreakOffset – Смещение излома

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BreakOffset = iObject.BreakOffset          Получить свойство(* )
iObject.BreakOffset = BreakOffset          Установить свойство (* )
BreakOffset = iObject.GetBreakOffset()     Получить свойство (**)
iObject.SetBreakOffset( BreakOffset )      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_BreakOffset(&BreakOffset)     Получить свойство
iObject->put_BreakOffset( BreakOffset)     Установить свойство
```

Примечание:

Значение свойства рассчитывается в процентах от длины выступа.

Length – Длина выступа

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = iObject.Length                    Получить свойство(* )
iObject.Length = Length                    Установить свойство (* )
Length = iObject.GetLength()               Получить свойство (**)
iObject.SetLength( Length )                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Length(&Length)               Получить свойство
iObject->put_Length( Length)               Установить свойство
```

Примечание:

Свойство позволяет получить и установить координату центра дуги Xc.

MarkOffset – Смещение марки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MarkOffset = iObject.MarkOffset            Получить свойство(* )
iObject.MarkOffset = MarkOffset            Установить свойство (* )
MarkOffset = iObject.GetMarkOffset()       Получить свойство (**)
iObject.SetMarkOffset( MarkOffset )        Установить свойство (**)
```

Синтаксис COM:

iObject->get_MarkOffset(&MarkOffset)	Получить свойство
iObject->put_MarkOffset(MarkOffset)	Установить свойство

Примечание:

Свойство позволяет получить и установить расстояние от марки до оси.

MarkOn – Марка включена\выключена

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MarkOn = iObject.MarkOn	Получить свойство(*)
iObject.MarkOn = MarkOn	Установить свойство (*)
MarkOn = iObject.GetMarkOn()	Получить свойство (**)
iObject.SetMarkOn(MarkOn)	Установить свойство (**)

Синтаксис COM:

iObject->get_MarkOn(&MarkOn)	Получить свойство
iObject->put_MarkOn(MarkOn)	Установить свойство

Примечание:

Свойство позволяет управлять отображением марки на координационной оси.

Интерфейс IMarkNode

Справка системы КОМПАС...

KOMPAS.chm::/306_29_11_5_Dopolnitelqnye_oboz.htm#Rfv60991

Интерфейс узла для вставки дополнительных марок.

Примечание:

Интерфейс можно получить с помощью методов IMarkNodes::Item, IMarkNodes::Add, ICircleAxis::BaseMarkNode.

IMarkNode – свойства

DoubleMark – Признак марки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DoubleMark = iObject.DoubleMark	Получить свойство(*)

iObject.DoubleMark = DoubleMark — Установить свойство (*)
DoubleMark = iObject.GetDoubleMark() — Получить свойство (**)
iObject.SetDoubleMark(DoubleMark) — Установить свойство (**)

Синтаксис COM:

iObject->get_DoubleMark(&DoubleMark) — Получить свойство
iObject->put_DoubleMark(DoubleMark) — Установить свойство

Значения свойства:

TRUE - двойная окружность,
FALSE - одинарная.

Примечание:

Свойство используется для узлов типов ksMarkCirle и ksMarkRefCirle из перечисления ksMarkNodeEnum.

MarkNodes – Коллекция узлов марок

Интерфейс...

Тип данных: указатель на интерфейс IMarkNodes

Синтаксис Automation:

MarkNodes = iObject.MarkNodes(Left) — Получить свойство(*)
MarkNodes = iObject.GetMarkNodes(Left) — Получить свойство (**)

Синтаксис COM:

iObject->get_MarkNodes(Left, &MarkNodes) — Получить свойство

Входные параметры:

Left (BOOL) — TRUE - возвращается коллекция узлов слева по направлению оси, FALSE - справа.

Примечание:

Свойство доступно только для чтения.

MarkType – Тип марки

Интерфейс...

Тип данных: тип марки из перечисления ksMarkNodeEnum

Синтаксис Automation:

MarkType = iObject.MarkType — Получить свойство(*)
iObject.MarkType = MarkType — Установить свойство (*)

MarkType = iObject.GetMarkType()	Получить свойство (**)
iObject.SetMarkType(MarkType)	Установить свойство (**)

Синтаксис COM:

iObject->get_MarkType(&MarkType)	Получить свойство
iObject->put_MarkType(MarkType)	Установить свойство

Примечание:

Свойство позволяет получить и установить тип марки.

RefLength – Длина указателя

Интерфейс...

Тип данных: double

Синтаксис Automation:

RefLength = iObject.RefLength	Получить свойство(*)
iObject.RefLength = RefLength	Установить свойство (*)
RefLength = iObject.GetRefLength()	Получить свойство (**)
iObject.SetRefLength(RefLength)	Установить свойство (**)

Синтаксис COM:

iObject->get_RefLength(&RefLength)	Получить свойство
iObject->put_RefLength(RefLength)	Установить свойство

Примечание:

Свойство используется для узлов типа ksMarkRefCirle из перечисления ksMarkNodeEnum.

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

Text = iObject.Text	Получить свойство(*)
Text = iObject.GetText()	Получить свойство (**)

Синтаксис COM:

iObject->get_Text(&Text)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Текст можно использовать для поиска узла марки в коллекции IMarkNodes.
3. Для поиска в функцию IMarkNodes::Item нужно передать строковое значение текста.

Пример:

```
node = iMarkNodes.Item("A1");
```

IMarkNode – методы

Delete – Удалить узел

Интерфейс...

Синтаксис Automation:

```
BOOL Delete();
```

Синтаксис COM:

```
HRESULT Delete( BOOL * Result );
```

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет удалить дополнительные узлы для марок.
2. Из объекта узлы удалятся после вызова метода IDrawingObject::Update.

GetInsertionPoints – Получить координаты точек для вставки дополнительных элементов

Интерфейс...

Синтаксис Automation:

```
BOOL GetInsertionPoints( BOOL LeftJut,
VARIANT * Points,
VARIANT * Directions );
```

Синтаксис COM:

```
HRESULT GetInsertionPoints( BOOL LeftJut,
VARIANT * Points,
VARIANT * Directions, BOOL * RetVal );
```

Входные параметры:

LeftJut	- TRUE координаты точек для левого ответвления, - FALSE - для правого. Для круглой оси нужно передавать LeftJut = TRUE.

Выходные параметры:

Points	- массив SafeArray типа VT_ARRAY I VT_R8 координат точек для подключения дополнительных узлов марок,
Directions	- массив SafeArray типа VT_ARRAY I VT_R8 направлений (углы относительно центра родительского узла).

Возвращаемое значение:

TRUE	- координаты получены,
FALSE	- в случае неудачи.

Примечание:

1. Параметры Points, Directions не являются обязательными.
2. В функцию достаточно передать один из указателей на VARIANT.
3. Массивы являются согласованными.
4. Координаты точек в массиве Points лежат в следующей последовательности:

▼ x0, y0, x1, y1, ...xi, yi.

5. Направления в массиве Directions лежат в последовательности:

▼ angle0, angle1, ...anglei.

6. Значения координат и направлений пересчитываются в зависимости от текущей системы координат для объекта, задаваемой параметром IDrawingObject::DrawingObjectParamType

Ограничения

Интерфейс IParametriticConstraint

Справка системы КОМПАС...

`kompas.chm::/471_54_7_Parametricheskij_rezhi.htm`

Интерфейс параметрического ограничения.

Иерархия:

IKompasAPIObject

IParametriticConstraint

Описание:

Интерфейс позволяет получить доступ к параметрическому ограничению в 2D документе.

Изменение параметров ограничения возможно на этапе создания ограничения. Создать ограничение можно при помощи метода IDrawingObject1::NewConstraint.

Если необходимо отредактировать параметрическое ограничение, то его нужно удалить и создать новое с нужными параметрами.

Примечание:

1. Аналоги интерфейса в API5:

   API Экспортных функций:

- Структура параметров: ConstraintParam.

  Функции:

- ksSetObjConstraint - Установить параметрическое ограничение.
- ksDestroyObjConstraint - Разрушить параметрическое ограничение.

  Automation:

- ksConstraintParam - Интерфейс параметров для параметрических ограничений.

  Методы:

- ksDocument2D::ksSetObjConstraint - Установить параметрическое ограничение.
- ksDocument2D::ksDestroyObjConstraint - Разрушить параметрическое ограничение.

2. В API5 нельзя создать или получить ограничения следующих типов:

- Ассоциативная связь.
- Размер с переменной.
- Фиксированный размер.

IParametriticConstraint – свойства

Axis – Ось симметрии для ограничения "симметрия точек"

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

```
Axis = Object.Axis          Получить свойство(* )
Object.Axis = Axis          Установить свойство (* )
Axis = Object.GetAxis()     Получить свойство (**)
Object.SetAxis( Axis )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Axis( &Axis )    Получить свойство
Object.put_Axis(  Axis )    Установить свойство
```

Примечание:

Свойство позволяет задать ось симметрии для ограничения типа "симметрия точек". В качестве оси симметрии может использоваться отрезок или объект "прямая ось".

AxisSegmentIndex – Индекс сегмента для оси

Интерфейс...

Тип данных: long

Синтаксис Automation:

AxisSegmentIndex = Object. AxisSegmentIndex	Получить свойство(*)
Object. AxisSegmentIndex = AxisSegmentIndex	Установить свойство (*)
AxisSegmentIndex = Object.Get AxisSegmentIndex()	Получить свойство (**)
Object.Set AxisSegmentIndex(AxisSegmentIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_ AxisSegmentIndex(& AxisSegmentIndex)	Получить свойство
Object.put_ AxisSegmentIndex(AxisSegmentIndex)	Установить свойство

BisectorVariant – Вариант решения биссектрисы

Интерфейс...

Тип данных: из перечисления ksBisectorVariant

Синтаксис Automation:

BisectorVariant = Object.BisectorVariant	Получить свойство(*)
Object.BisectorVariant = BisectorVariant	Установить свойство (*)
BisectorVariant = Object.GetBisectorVariant( )	Получить свойство (**)
Object.SetBisectorVariant(BisectorVariant)	Установить свойство (**)

Синтаксис COM:

Object.get_BisectorVariant(&BisectorVariant)	Получить свойство

Object.put_BisectorVariant(BisectorVariant)	Установить свойство

Примечание:

Свойство имеет смысл только для ограничения с типом ksCBisector.

Comment – Комментарий

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Comment = iObject.Comment	Получить свойство(*)
iObject.Comment = Comment	Установить свойство (*)
Comment = iObject.GetComment()	Получить свойство (**)
iObject.SetComment(Comment)	Установить свойство (**)

Синтаксис COM:

Comment = iObject->get_Comment()	Получить свойство
iObject->put_Comment(Comment)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить комментарий. Используется только для ограничения "Размер с переменной" (ksCDimWithVariable). Комментарий можно установить для размеров при редактировании эскизов.
2. Изменение параметра возможно на этапе создания ограничения.

ConstraintType – Тип ограничения

Интерфейс...

Тип данных: из перечисления ksConstraintTypeEnum

Синтаксис Automation:

ConstraintType = iObject.ConstraintType	Получить свойство(*)
iObject.ConstraintType = ConstraintType	Установить свойство (*)
ConstraintType = iObject.GetConstraintType( )	Получить свойство (**)
iObject.SetConstraintType(ConstraintType)	Установить свойство (**)

Синтаксис COM:

ConstraintType = iObject->get_ConstraintType()	Получить свойство
iObject->put_ConstraintType(ConstraintType)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить тип ограничения.
2. Изменение параметра возможно на этапе создания ограничения.

Degrees – Градусы

Интерфейс...

Тип данных: long

Синтаксис Automation:

Degrees = iObject.Degrees	Получить свойство(*)
iObject.Degrees = Degrees	Установить свойство (*)
Degrees = iObject.GetDegrees()	Получить свойство (**)
iObject.SetDegrees(Degrees)	Установить свойство (**)

Синтаксис COM:

Degrees = iObject->get_Degrees()	Получить свойство
iObject->put_Degrees(Degrees)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить составляющую "Градусы" для значения углового размера. Используется только для ограничения "Размер с переменной" (ksCDimWithVariable).
2. Изменение параметра возможно на этапе создания ограничения.

Expression – Выражение

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Expression = iObject.Expression	Получить свойство(*)
iObject.Expression = Expression	Установить свойство (*)
Expression = iObject.GetExpression()	Получить свойство (**)
iObject.SetExpression(Expression)	Установить свойство (**)

Синтаксис COM:

Expression = iObject->get_Expression()	Получить свойство
iObject->put_Expression(Expression)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить выражение. Используется только для ограничения "Размер с переменной" (ksCDimWithVariable).
2. Изменение параметра возможно на этапе создания ограничения.

Index – Индекс точки на объекте (начинается с 0, у дуги и окружности 0–центр)

Интерфейс...

Тип данных: long

Синтаксис Automation:

Index = iObject.Index	Получить свойство(*)
iObject.Index = Index	Установить свойство (*)
Index = iObject.GetIndex()	Получить свойство (**)
iObject.SetIndex(Index)	Установить свойство (**)

Синтаксис COM:

Index = iObject->get_Index()	Получить свойство
iObject->put_Index(Index)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить индекс точки на данном графическом объекте. Применяется, например, для ограничения "Совпадение двух точек"(ksCMergePoints).
2. Изменение параметра возможно на этапе создания ограничения.

Minutes – Минуты

Интерфейс...

Тип данных: long

Синтаксис Automation:

Minutes = iObject.Minutes	Получить свойство(*)
iObject.Minutes = Minutes	Установить свойство (*)
Minutes = iObject.GetMinutes()	Получить свойство (**)

iObject.SetMinutes(Minutes)	Установить свойство (**)

Синтаксис COM:

Minutes = iObject->get_Minutes()	Получить свойство
iObject->put_Minutes(Minutes)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить составляющую "Минуты" для значения углового размера. Используется только для ограничения "Размер с переменной"(ksCDimWithVariable).
2. Изменение параметра возможно на этапе создания ограничения.

Partner – Второй объект или массив SAFEARRAY объектов для установки ограничения

Интерфейс...

Тип данных: VARIANT (VT_ARRAY | VT_DISPATCH) или VT_DISPATCH

Синтаксис Automation:

Partner = iObject.Partner	Получить свойство(*)
iObject.Partner = Partner	Установить свойство (*)
Partner = iObject.GetPartner()	Получить свойство (**)
iObject.SetPartner(Partner)	Установить свойство (**)

Синтаксис COM:

Partner = iObject->get_Partner()	Получить свойство
iObject->put_Partner(Partner)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить второй объект. SAFEARRAY используется только для ограничения "Ассоциативная связь"(ksCAssociation). Например, для углового размера ассоциативная связь может быть установлена на два отрезка, между которыми установлен размер.
2. Изменение параметра возможно на этапе создания ограничения.

PartnerIndex – Индекс точки на втором объекте (начинается с 0, у дуги и окружности 0–центр)

Интерфейс...

Тип данных: long

Синтаксис Automation:

PartnerIndex = iObject.PartnerIndex	Получить свойство(*)
iObject.PartnerIndex = PartnerIndex	Установить свойство (*)
PartnerIndex = iObject.GetPartnerIndex()	Получить свойство (**)
iObject.SetPartnerIndex(PartnerIndex)	Установить свойство (**)

Синтаксис COM:

PartnerIndex = iObject->get_PartnerIndex()	Получить свойство
iObject->put_PartnerIndex(PartnerIndex)	Установить свойство

Значения свойства:

TRUE	- описание активно,
FALSE	- описание неактивно.

Примечание:

Свойство позволяет установить и получить индекс точки на втором графическом объекте, который используется для создания ограничения.

Применяется, например, для ограничения "Совпадение двух точек"(ksCMergePoints).

Изменение параметра возможно на этапе создания ограничения.

PartnerSegmentIndex – Индекс сегмента партнера

Интерфейс...

Тип данных: long

Синтаксис Automation:

PartnerSegmentIndex = Object.PartnerSegmentIndex	Получить свойство(*)
Object.PartnerSegmentIndex = PartnerSegmentIndex	Установить свойство (*)
PartnerSegmentIndex = Object.GetPartnerSegmentIndex()	Получить свойство (**)
Object.SetPartnerSegmentIndex(PartnerSegmentIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_PartnerSegmentIndex(&PartnerSegmentIndex)	Получить свойство
Object.put_PartnerSegmentIndex(PartnerSegmentIndex)	Установить свойство

Seconds – Секунды

Интерфейс...

Тип данных: double

Синтаксис Automation:

Seconds = iObject.Seconds	Получить свойство(*)
iObject.Seconds = Seconds	Установить свойство (*)
Seconds = iObject.GetSeconds()	Получить свойство (**)
iObject.SetSeconds(Seconds)	Установить свойство (**)

Синтаксис COM:

Seconds = iObject->get_Seconds()	Получить свойство
iObject->put_Seconds(Seconds)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить составляющую "Секунды" для значения углового размера. Используется только для ограничения "Размер с переменной"(ksCDimWithVariable).
2. Изменение параметра возможно на этапе создания ограничения.

SegmentIndex – Индекс сегмента объекта

Интерфейс...

Тип данных: long

Синтаксис Automation:

SegmentIndex = Object.SegmentIndex	Получить свойство(*)
Object.SegmentIndex = SegmentIndex	Установить свойство (*)
SegmentIndex = Object.GetSegmentIndex()	Получить свойство (**)
Object.SetSegmentIndex(SegmentIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_SegmentIndex(&SegmentIndex)	Получить свойство
Object.put_SegmentIndex(SegmentIndex)	Установить свойство

Valid – Действие ограничения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Valid = iObject.Valid                    Получить свойство(* )
Valid = iObject.GetValid()               Получить свойство (**)
```

Синтаксис COM:

```
Valid = iObject->get_Valid()             Получить свойство
```

Значения свойства:

```
TRUE      - параметрическое ограничение действительно,
FALSE     - параметрическое ограничение недействительно.
```

Примечание:

Свойство позволяет определить действительно ограничение или нет.

Value – Значение

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Value = iObject.Value                    Получить свойство(* )
iObject.Value = Value                    Установить свойство (* )
Value = iObject.GetValue()               Получить свойство (**)
iObject.SetValue(Value)                  Установить свойство (**)
```

Синтаксис COM:

```
Value = iObject->get_Value()             Получить свойство
iObject->put_Value(Value)                Установить свойство
```

Примечание:

1. Свойство позволяет установить и получить значение. Используется только для ограничения "Размер с переменной" (ksCDimWithVariable).

   Установить значение можно, если на объекте есть ограничение "Фиксированный размер"(ksCFixedDim).
2. Размер не является информационным.
3. Изменение параметра возможно на этапе создания ограничения.

Variable – Имя переменной

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Variable = iObject.Variable	Получить свойство(*)
iObject.Variable = Variable	Установить свойство (*)
Variable = iObject.GetVariable()	Получить свойство (**)
iObject.SetVariable(Variable)	Установить свойство (**)

Синтаксис COM:

Variable = iObject->get_Variable()	Получить свойство
iObject->put_Variable(Variable)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить имя переменной. Используется только для ограничения "Размер с переменной" (ksCDimWithVariable).
2. При создании ограничения ksCDimWithVariable автоматически присваивается уникальное имя переменной. При установке своего имени нужно следить за уникальностью имени самостоятельно. Иначе ограничение не создастся и взведется ошибка "Ограничение создать нельзя. Недопустимое имя переменной."
3. Изменение параметра возможно на этапе создания ограничения.

IParametriticConstraint – методы

Create – Создать ограничение в модели

Интерфейс...

Синтаксис Automation:

BOOL Create();

Синтаксис COM:

HRESULT Create([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- параметрическое ограничение создано,
FALSE	- в случае неудачи.

Примечание:

1. Если метод выполнился успешно, ограничение появится в модели и вступит в силу. Значение свойства IParametriticConstraint::Valid становится равным TRUE (ограничение недействительно). Если ограничение в модели создать нельзя, метод Create вернет FALSE.
2. В процессе создания ограничения могут быть взведены ошибки из перечисления ErrorType:

- Значение выходит за границы диапазона.
- Значение размера выходит за границы диапазона 30'' - 359°59'30''.
- Значение размера выходит за границы диапазона 0.5'' - 359°59'59.5''.

- Значение размера выходит за границы диапазона 30' - 359°30'.
- Ограничение создать нельзя. Данный тип размера не параметризуется.
- Ограничение создать нельзя. Нет информации о привязке.
- Ограничение создать нельзя.
- Ограничение такого типа уже существует.
- Ограничение создать нельзя. Недопустимое имя переменной.
- Ограничение создать нельзя. Недопустимое значение размера.
- Ограничение создать нельзя. Размер может быть только информационным.

Delete – Удалить ограничение

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- параметрическое ограничение удалено,
FALS E	- в случае неудачи.

Примечание:

Метод удаляет описание ограничение. Значение свойства IParametriticConstraint::Valid становится равным FALSE (ограничение недействительно).

Интерфейс IDrawingObject1

Дополнительный интерфейс для графических объектов.

Иерархия:

IDispatch

IDrawingObject1

Примечание:

1. Дополнительный интерфейс для графических объектов. Данный интерфейс можно получить у интерфейса графического объекта IDrawingObject посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).
2. Интерфейс позволяет выполнить следующие действия:

- получить массив ограничений для данного графического объекта,
- добавить новое ограничение,
- ассоциировать данный графический объект с другими объектами,
- удалить все ограничения.

IDrawingObject1 – свойства

AutoTransparentBackground – Включение/отключение управления очисткой фона под объектом через настройку документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoTransparentBackground = Object.AutoTransparentBackground	Получить свойство(*)
Object.AutoTransparentBackground = AutoTransparentBackground	Установить свойство (*)
AutoTransparentBackground = Object.GetAutoTransparentBackground()	Получить свойство (**)
Object.SetAutoTransparentBackground(AutoTransparentBackground)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoTransparentBackground(&AutoTransparentBackground)	Получить свойство
Object.put_AutoTransparentBackground(AutoTransparentBackground)	Установить свойство

Значение свойства:

TRUE	- включено управление очисткой фона через настройку документа,
FALSE	- отключено управление очисткой фона через настройку документа.

Примечание:

Позволяет включить/отключить признак подчинения очистки фона под аннотационными объектами DrawingObject1::TransparentBackground настройкам документа.

Constraints – Массив SAFEARRAY ограничений для данного объекта

Интерфейс...

Аналоги метода в API5: экспортная функция ksGetObjConstraints и метод ksDocument2D::ksGetObjConstraints. Тип данных:VARIANT (VT_ARRAY|VT_DISPATCH).

Синтаксис Automation:

Constraints = iObject.Constraints	Получить свойство(*)
Constraints = iObject.GetConstraints()	Получить свойство (**)

Синтаксис COM:

Constraints = iObject->get_Constraints(&v)	Получить свойство,

Примечание:

Свойство позволяет получить массив SAFEARRAY ограничений для данного объекта. Это массив LPDISPATCH, который можно преобразовать в интерфейсы IParametriticConstraint. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Id – Идентификатор объекта

Интерфейс...

Тип данных: int64

Синтаксис Automation:

Id = Object.Id	Получить свойство(*)
Id = Object.GetId()	Получить свойство (**)

Синтаксис COM:

Object.get_Id(&Id)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IsAnnotativeObject – Признак аннотационного объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsAnnotativeObject = Object.IsAnnotativeObject	Получить свойство(*)
IsAnnotativeObject = Object.GetIsAnnotativeObject()	Получить свойство (**)

Синтаксис COM:

Object.get_IsAnnotativeObject(&IsAnnotativeObject)	Получить свойство

Примечание:

Позволяет получить признак аннотационного объекта.

Различаются геометрические, аннотационные и составные объекты.

См. также IDrawingObject1::IsGeometryObject.

IsCurve – Объект является кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCurve = Object.IsCurve	Получить свойство(*)
IsCurve = Object.GetIsCurve()	Получить свойство (**)

Синтаксис COM:

Object.get_IsCurve(&IsCurve)	Получить свойство

Значение свойства:

TRUE	- объект является кривой.

Свойство позволяет получить признак того, что объект является кривой.

Примечание:

Свойство доступно только для чтения.

IsGeometryObject – Признак геометрического объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsGeometryObject = Object.IsGeometryObject	Получить свойство(*)
IsGeometryObject = Object.GetIsGeometryObject()	Получить свойство (**)

Синтаксис COM:

Object.get_IsGeometryObject(&IsGeometryObject)	Получить свойство

Примечание:

Позволяет получить признак геометрического объекта.

Различаются геометрические, аннотационные и составные объекты.

См. также IDrawingObject1::IsAnnotativeObject.

TransparentBackground – Включение/отключение очистки фона под аннотационным объектом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShelfPoints = Object.ShelfPoints	Получить свойство(*)
Object.ShelfPoints = ShelfPoints	Установить свойство (*)
ShelfPoints = Object.GetShelfPoints()	Получить свойство (**)
Object.SetShelfPoints(ShelfPoints)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfPoints(&ShelfPoints)	Получить свойство
Object.put_ShelfPoints(ShelfPoints)	Установить свойство

Значение свойства:

TRUE	- включение очистки фона,
FALSE	- отключение очистки фона.

Примечание:

Позволяет включить или отключить очистку фона под аннотационными объектами.

IDrawingObject1 – методы

Associate – Ассоциировать данный объект с другими объектами

Интерфейс...

Синтаксис Automation:

BOOL Associate();

Синтаксис COM:

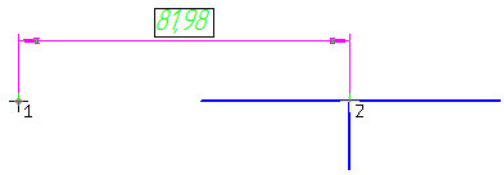
HRESULT Associate([out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Данный метод позволяет автоматизировать процесс наложения ассоциативных связей, если они возможны.
2. Существует также альтернативный вариант наложения ассоциативных связей - ручной, с использованием метода IDrawingObject1::NewConstraint. В этом случае пользователь сам несет ответственность за правильность заполнения параметров ограничения и за полноту ассоциативных связей.
3. В некоторых случаях, как например, на рисунке ниже, в результате использования данного метода при создании ассоциативного размера, с т.2 размера может быть ассоциирована как конечная точка вертикального отрезка, так и средняя точка горизонтального отрезка. В подобных ситуациях для обеспечения ассоциации точек размера с конкретными точками нужных объектов, требуется явно создавать новое ограничение через метод IDrawingObject1::NewConstraint.



Создание ограничения

Пример создания ограничения через метод IDrawingObject1::NewConstraint...

DeleteConstraints – Удалить все ограничения

Интерфейс...

Синтаксис Automation:

BOOL DeleteConstraints();

Синтаксис COM:

HRESULT DeleteConstraints([out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все ограничения для данного графического объекта.

DeleteHyperLink – Удалить гиперссылку

Интерфейс...

Синтаксис Automation:

BOOL DeleteHyperLink();

Синтаксис COM:

HRESULT DeleteHyperLink(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetCurve2D – Получить математическую кривую

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetCurve2D()

Синтаксис COM:

HRESULT GetCurve2D(ICurve2D ** Result);

Возвращаемое значение:

- указатель на интерфейс ICurve2D

GetHyperLinkParam – Получить параметры гиперссылки

Интерфейс...

Синтаксис Automation:

BSTR GetHyperLinkParam(ksHyperLinkTypeEnum * Type,
IDrawingObject ** LinkObject,
long * Level);

Синтаксис COM:

HRESULT GetHyperLinkParam(ksHyperLinkTypeEnum * Type,
IDrawingObject ** LinkObject,
long * Level,
BSTR * Result);

Возвращаемое значение:

текст гиперссылки.

Возвращаемое значение:

Type - тип ссылки,
LinkObject - объект, на который сделана ссылка,
Level - уровень.

NewConstraint – Создать новое ограничение

Интерфейс...

Синтаксис Automation:

LPDISPATCH NewConstraint();

Синтаксис COM:

HRESULT NewConstraint([out, retval] IParametriticConstraint** Result);

Возвращаемое значение:

указатель на интерфейс IParametriticConstraint параметрического ограничения

Примечание:

Метод создает новое параметрическое ограничение. Для изменения параметров ограничения, необходимо задать его свойства. Ограничение появится в модели и вступит в силу после вызова метода IParametriticConstraint::Create.

Параметры

Интерфейс IAxisLineParam

Справка системы КОМПАС...

KOMPAS.chm::/CM_AXEDLINESEG.htm

Параметры осевой линии или обозначения центра.

Иерархия:

IDispatch

IKompasAPIObject

IAxisLineParam

Описание:

Интерфейс позволяет задавать и получать параметры осевой линии.

Примечание:

Интерфейс является дополнительным к интерфейсу осевой линии IAxisLine и интерфейсу обозначения центра ICentreMarker.

IAxisLineParam – свойства

AutoDetectedDash – Ручное/автоматическое задание длины штриха

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoDetectedDash = Object.AutoDetectedDash	Получить свойство(*)
Object.AutoDetectedDash = AutoDetectedDash	Установить свойство (*)
AutoDetectedDash = Object.GetAutoDetectedDash()	Получить свойство (**)
Object.SetAutoDetectedDash(AutoDetectedDash)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoDetectedDash(&AutoDetectedDash)	Получить свойство
Object.put_AutoDetectedDash(AutoDetectedDash)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автоматического задания длины штриха осевой линии.

AutoDetectedDashModify – Использовать параметр Задание длины штриха из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoDetectedDashModify = Object.AutoDetectedDashModify	Получить свойство(*)
Object.AutoDetectedDashModify = AutoDetectedDashModify	Установить свойство (*)
AutoDetectedDashModify = Object.GetAutoDetectedDashModify()	Получить свойство (**)
Object.SetAutoDetectedDashModify(AutoDetectedDashModify)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoDetectedDashModify(&AutoDetectedDashModify)	Получить свойство

Object.put_AutoDetectedDashModify(AutoDetectedDashModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования параметра Задание длины штриха из настроек документа.

DashMaxLength – Максимальная длина штриха

Интерфейс...

Тип данных: double

Синтаксис Automation:

DashMaxLength = Object.DashMaxLength	Получить свойство(*)
Object.DashMaxLength = DashMaxLength	Установить свойство (*)
DashMaxLength = Object.GetDashMaxLength()	Получить свойство (**)
Object.SetDashMaxLength(DashMaxLength)	Установить свойство (**)

Синтаксис COM:

Object.get_DashMaxLength(&DashMaxLength)	Получить свойство
Object.put_DashMaxLength(DashMaxLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать максимальную длину штриха осевой линии.

DashMaxLengthModify – Использовать параметр Максимальная длина штриха из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DashMaxLengthModify = Object.DashMaxLengthModify	Получить свойство(*)
Object.DashMaxLengthModify = DashMaxLengthModify	Установить свойство (*)
DashMaxLengthModify = Object.GetDashMaxLengthModify()	Получить свойство (**)
Object.SetDashMaxLengthModify(DashMaxLengthModify)	Установить свойство (**)

Синтаксис COM:

Object.get_DashMaxLengthModify(&DashMaxLengthModify)	Получить свойство
Object.put_DashMaxLengthModify(DashMaxLengthModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования параметра Максимальная длина штриха из настроек документа.

DottedLength – Длина пунктира

Интерфейс...

Тип данных: double

Синтаксис Automation:

DottedLength = Object.DottedLength	Получить свойство(*)
Object.DottedLength = DottedLength	Установить свойство (*)
DottedLength = Object.GetDottedLength()	Получить свойство (**)
Object.SetDottedLength(DottedLength)	Установить свойство (**)

Синтаксис COM:

Object.get_DottedLength(&DottedLength)	Получить свойство
Object.put_DottedLength(DottedLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину пунктира осевой линии.

DottedLengthModify – Использовать параметр Длина пунктира из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DottedLengthModify = Object.DottedLengthModify	Получить свойство(*)
Object.DottedLengthModify = DottedLengthModify	Установить свойство (*)
DottedLengthModify = Object.GetDottedLengthModify()	Получить свойство (**)
Object.SetDottedLengthModify(DottedLengthModify)	Установить свойство (**)

Синтаксис COM:

Object.get_DottedLengthModify(&DottedLengthModify)	Получить свойство
Object.put_DottedLengthModify(DottedLengthModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования параметра Длина пунктира из настроек документа.

Interval – Промежуток

Интерфейс...

Тип данных: double

Синтаксис Automation:

Interval = Object.Interval	Получить свойство(*)
Object.Interval = Interval	Установить свойство (*)
Interval = Object.GetInterval()	Получить свойство (**)
Object.SetInterval(Interval)	Установить свойство (**)

Синтаксис COM:

Object.get_Interval(&Interval)	Получить свойство
Object.put_Interval(Interval)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать длину промежутка между штрихами пунктира осевой линии.

IntervalModify – Использовать параметр Промежуток из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IntervalModify = Object.IntervalModify	Получить свойство(*)
Object.IntervalModify = IntervalModify	Установить свойство (*)
IntervalModify = Object.GetIntervalModify()	Получить свойство (**)
Object.SetIntervalModify(IntervalModify)	Установить свойство (**)

Синтаксис COM:

Object.get_IntervalModify(&IntervalModify)	Получить свойство
Object.put_IntervalModify(IntervalModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования параметра Промежуток из настроек документа.

JutLength – Выступание осевой линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

JutLength = Object.JutLength	Получить свойство(*)
Object.JutLength = JutLength	Установить свойство (*)
JutLength = Object.GetJutLength()	Получить свойство (**)
Object.SetJutLength(JutLength)	Установить свойство (**)

Синтаксис COM:

Object.get_JutLength(&JutLength)	Получить свойство
Object.put_JutLength(JutLength)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать выступание осевой линии.

JutLength1 – Выступ осевой 1

Интерфейс...

Тип данных: double

Синтаксис Automation:

JutLength1 = Object.JutLength1	Получить свойство(*)
Object.JutLength1 = JutLength1	Установить свойство (*)
JutLength1 = Object.GetJutLength1()	Получить свойство (**)
Object.SetJutLength1(JutLength1)	Установить свойство (**)

Синтаксис COM:

Object.get_JutLength1(&JutLength1)	Получить свойство

Object.put_JutLength1(JutLength1)	Установить свойство

JutLength2 – Выступ осевой 2

Интерфейс...

Тип данных: double

Синтаксис Automation:

JutLength2 = Object.JutLength2	Получить свойство(*)
Object.JutLength2 = JutLength2	Установить свойство (*)
JutLength2 = Object.GetJutLength2()	Получить свойство (**)
Object.SetJutLength2(JutLength2)	Установить свойство (**)

Синтаксис COM:

Object.get_JutLength2(&JutLength2)	Получить свойство
Object.put_JutLength2(JutLength2)	Установить свойство

JutLengthModify – Использовать параметр Выступание осевой из настроек документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

JutLengthModify = Object.JutLengthModify	Получить свойство(*)
Object.JutLengthModify = JutLengthModify	Установить свойство (*)
JutLengthModify = Object.GetJutLengthModify()	Получить свойство (**)
Object.SetJutLengthModify(JutLengthModify)	Установить свойство (**)

Синтаксис COM:

Object.get_JutLengthModify(&JutLengthModify)	Получить свойство
Object.put_JutLengthModify(JutLengthModify)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак использования параметра Выступание осевой из настроек документа.

Интерфейс IBreakViewParam

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_BROKEN_VIEW.htm

Интерфейс параметров разрыва вида.

Иерархия:

IDispatch

IBreakViewParam

Описание:

Интерфейс позволяет задавать и получать параметры разрыва вида.

Примечание:

Интерфейс является дополнительным к интерфейсу вида IView для видов всех типов, кроме системного.

IBreakViewParam – свойства

BreaksCount – Количество линий разрыва

Интерфейс...

Тип данных: long

Синтаксис Automation:

BreaksCount = Object.BreaksCount	Получить свойство(*)
BreaksCount = Object.GetBreaksCount()	Получить свойство (**)

Синтаксис COM:

Object.get_BreaksCount(&BreaksCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество линий разрыва.
2. Свойство доступно только для чтения.

Breaksvisible – Отображение линий разрыва

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BreaksVisible = Object.BreaksVisible	Получить свойство(*)
Object.BreaksVisible = BreaksVisible	Установить свойство (*)
BreaksVisible = Object.GetBreaksVisible()	Получить свойство (**)
Object.SetBreaksVisible(BreaksVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_BreaksVisible(&BreaksVisible)	Получить свойство
Object.put_BreaksVisible(BreaksVisible)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак отображения разрывов.

IBreakViewParam – методы

AddBreakLine – Добавить линию разрыва

Интерфейс...

Синтаксис Automation:

long AddBreakLine(double X1,
double Y1,
double X2,
double Y2,
double angle)

Синтаксис COM:

HRESULT AddBreakLine(double X1,
double Y1,
double X2,
double Y2,
double angle
long * Index);

Входные параметры:

X1, Y1, X2, Y2	- координаты границы разрыва,
angle	- угол.

Возвращаемое значение:

- индекс линии разрыва.

Примечание:

Метод позволяет добавить линию разрыва вида.

Угол первого разрыва может быть произвольным. Последующие разрывы должны быть параллельны или перпендикулярны первому.

Например, в виде создан разрыв, направление сдвига которого расположено под углом 30 градусов к оси X. Впоследствии в этом виде можно будет создать разрывы, направления сдвига которых располагаются либо под таким же углом, либо под углом 120 градусов к оси X.

DeleteBreakLine – Удалить линию разрыва

Интерфейс...

Синтаксис Automation:

BOOL DeleteBreakLine(long Index)

Синтаксис COM:

HRESULT DeleteBreakLine(long Index, BOOL * Result);

Входные параметры:

Index - индекс удаляемой линии разрыва.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет удалить линию разрыва вида с заданным индексом.

DeleteAllBreakLines – Удалить все линии разрыва

Интерфейс...

Синтаксис Automation:

BOOL DeleteAllBreakLines()

Синтаксис COM:

HRESULT DeleteAllBreakLines(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет удалить все линии разрыва вида.

GetBreakLineParams – Получить параметры линии разрыва

Интерфейс...

Синтаксис Automation:

BOOL SetBreakLineParams(long Index,
double Angle,
double Clearance,
long BreakLineType,
double Amplitude,
double MaxAmplitude)

Синтаксис COM:

HRESULT SetBreakLineParams(long Index,
double Angle,
double Clearance,
ksBreakLineTypeEnum BreakLineType,
double Amplitude,
double MaxAmplitude,
BOOL * Result);

Входные параметры:

Index	- индекс линии разрыва,

Выходные параметры:

Angle	- угол, кратный 90 градусам,
Clearance	- зазор,
BreakLineType	- тип линии разрыва,
Amplitude	- амплитуда,
MaxAmplitude	- максимальная амплитуда.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получать параметры границ разрыва.

GetBreakLinePosition – Получить координаты границ разрыва

Интерфейс...

Синтаксис Automation:

BOOL GetBreakLinePosition(long Index,
double * X1,
double * Y1,
double * X2,
double * Y2)

Синтаксис COM:

HRESULT GetBreakLinePosition(long Index,
double * X1,
double * Y1,
double * X2,
double * Y2,
BOOL * Result);

Входные параметры:

Index	- индекс линии разрыва.

Выходные параметры:

X1, Y1, X2, Y2	- координаты границы разрыва.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получать координаты границ разрыва.

SetBreakLineParams – Установить параметры линии разрыва

Интерфейс...

Синтаксис Automation:

```
BOOL SetBreakLineParams( long Index,
double  Angle,
double  Clearance,
long BreakLineType,
double Amplitude,
double MaxAmplitude )
```

Синтаксис COM:

```
HRESULT SetBreakLineParams( long Index,
double Angle,
double Clearance,
ksBreakLineTypeEnum BreakLineType,
double Amplitude,
double MaxAmplitude,
BOOL * Result );
```

Входные параметры:

Index	- индекс линии разрыва,
Angle	- угол, кратный 90 градусам,
Clearance	- зазор,
BreakLineType	- тип линии разрыва,
Amplitude	- амплитуда,
MaxAmplitude	- максимальная амплитуда.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет установить параметры границ разрыва.

SetBreakLinePosition – Установить координаты границ разрыва

Интерфейс...

Синтаксис Automation:

BOOL SetBreakLinePosition(long Index,
double X1,
double Y1,
double X2,
double Y2)

Синтаксис COM:

HRESULT SetBreakLinePosition(long Index,
double X1,
double Y1,
double X2,
double Y2,
BOOL * Result);

Входные параметры:

Index - индекс линии разрыва,
X1, Y1, X2, Y2 - координаты границы разрыва.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет установить координаты границ разрыва.

Интерфейс ICopyObjectParam1

Интерфейс дополнительных параметров копирования объектов

Иерархия:

IDispatch

ICopyObjectParam1

Примечание:

Интерфейс является дополнительным для ICopyObjectParam.

ICopyObjectParam1 – свойства

HyperLinksCopy – Копировать ссылки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HyperLinksCopy = Object.HyperLinksCopy	Получить свойство(*)
Object.HyperLinksCopy = HyperLinksCopy	Установить свойство (*)
HyperLinksCopy = Object.GetHyperLinksCopy()	Получить свойство (**)
Object.SetHyperLinksCopy(HyperLinksCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_HyperLinks Copy(&HyperLinksCopy)	Получить свойство
Object.put_HyperLinks Copy(HyperLinksCopy)	Установить свойство

Значения свойства:

TRUE	- копировать ссылки,
FALSE	- не копировать ссылки.

StoragesCopy – TRUE – копировать пользовательские данные и свойства

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StoragesCopy = Object.StoragesCopy	Получить свойство(*)
Object.StoragesCopy = StoragesCopy	Установить свойство (*)

StoragesCopy = Object.GetStoragesCopy()	Получить свойство (**)
Object.SetStoragesCopy(StoragesCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_StoragesCopy(&StoragesCopy)	Получить свойство
Object.put_StoragesCopy(StoragesCopy)	Установить свойство

Интерфейс ICutViewParam

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_BROKEN_VIEW.htm

Интерфейс параметров разреза вида.

Иерархия:

IDispatch

ICutViewParam

Описание:

Интерфейс позволяет задавать и получать параметры разреза вида.

Примечание:

Интерфейс является дополнительным к интерфейсу вида IView для видов всех типов, кроме системного.

ICutViewParam – свойства

CutsCount – Количество разрезов

Интерфейс...

Тип данных: long

Синтаксис Automation:

CutsCount = Object.CutsCount	Получить свойство(*)
CutsCount = Object.GetCutsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_CutsCount(&CutsCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество разрезов.
2. Свойство доступно только для чтения.

HatchParam – Параметры штриховки

Интерфейс...

Тип данных: указатель на интерфейс параметров штриховки IHatchParam

Синтаксис Automation:

HatchParam = Object.HatchParam	Получить свойство(*)
HatchParam = Object.GetHatchParam()	Получить свойство (**)

Синтаксис COM:

Object.get_HatchParam(&HatchParam)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс параметров штриховки разреза.
2. Свойство доступно только для чтения.

LocalCut – Местный разрез

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LocalCut = Object.LocalCut	Получить свойство(*)
Object.LocalCut = LocalCut	Установить свойство (*)
LocalCut = Object.GetLocalCut()	Получить свойство (**)
Object.SetBreaksVisible(BreaksVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_LocalCut(&LocalCut)	Получить свойство
Object.put_LocalCut(LocalCut)	Установить свойство

Примечание:

Свойство позволяет включать и выключать местный разрез.

ICutViewParam – методы

AddCut – Добавить разрез

Интерфейс...

Синтаксис Automation:

long AddCut(BSTR Name,
long Number,
double X,
double Y,
BOOL ModelCut,
LPDISPATCH * Contour,
LPDISPATCH * View)

Синтаксис COM:

HRESULT AddCut(BSTR Name,
long umber,
double X,
double Y,
BOOL ModelCut,
IDrawingObject * Contour,
IView * View,
long * Index);

Входные параметры:

Name	- имя разреза,
Number	- номер разреза,
X, Y	- координаты точки, через которую пройдет секущая плоскость,
ModelCut	- признак разреза, TRUE - разрез, FALSE - сечение,
Contour	- контур, ограничивающий разрез,
View	- опорный вид для разреза (проекционный вид для базового вида, в котором создается вырез).

Возвращаемое значение:

- Индекс разреза.

Примечание:

Метод позволяет добавить разрез.

DeleteCut – Удалить разрез

Интерфейс...

Синтаксис Automation:

BOOL DeleteCut(long Index)

Синтаксис COM:

HRESULT DeleteCut(long Index, BOOL * Result);

Входные параметры:

Index	- индекс удаляемого разреза.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить разрез с заданным индексом.

DeleteAllCuts – Удалить все разрезы

Интерфейс...

Синтаксис Automation:

BOOL DeleteAllCuts()

Синтаксис COM:

HRESULT DeleteAllCuts(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить все разрезы.

GetCutParams – Получить параметры разреза

Интерфейс...

Синтаксис Automation:

LPDISPATCH SetCutParams(long Index,
BSTR * Name,
long * Number,
BOOL * ModelCut)

Синтаксис COM:

HRESULT SetCutParams(long Index,
BSTR * Name,
long * Number,
BOOL * ModelCut,
IDrawingObject ** Contour);

Входные параметры:

Index	- индекс разреза.

Выходные параметры:

Name	- имя разреза,
Number	- номер разреза,
ModelCut	- признак разреза, TRUE - разрез, FALSE - сечение.

Возвращаемое значение:

- Указатель на контур, ограничивающий разрез.

Примечание:

Метод позволяет установить параметры разреза.

GetCutPlanePosition – Получить координаты точки, через которую пройдет секущая плоскость

Интерфейс...

Синтаксис Automation:

BOOL GetCutPlanePosition(long Index,
double * X,
double * Y)

Синтаксис COM:

HRESULT GetCutPlanePosition(long Index,
double * X,
double * Y,
BOOL * Result);

Входные параметры:

Index	- индекс разреза.

Выходные параметры:

X, Y	- координаты точки, через которую пройдет секущая плоскость.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

Метод позволяет получить координаты точки, через которую пройдет секущая плоскость.

SetCutParams – Установить параметры разреза

Интерфейс...

Синтаксис Automation:

BOOL SetCutParams(long Index,
BSTR Name,
long Number,
BOOL ModelCut)

Синтаксис COM:

HRESULT SetCutParams(long Index,
BSTR Name,
long Number,
BOOL ModelCut,
BOOL * Result);

Входные параметры:

Index	- индекс разреза,
Name	- имя разреза,
Number	- номер разреза,
ModelCut	- признак разреза, TRUE - разрез, FALSE - сечение.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры разреза.

SetCutPlanePosition – Установить координаты точки, через которую пройдет секущая плоскость

Интерфейс...

Синтаксис Automation:

BOOL SetCutPlanePosition(long Index,
double X,
double Y)

Синтаксис COM:

HRESULT SetCutPlanePosition(long Index,
double X,
double Y,
BOOL * Result);

Входные параметры:

Index	- индекс разреза,
X, Y	- координаты точки, через которую пройдет секущая плоскость.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить координаты точки, через которую пройдет секущая плоскость.

Интерфейс IDimensionParams

Справка системы КОМПАС...

kompas.chm::/176_Glava23_Obshchie_svedenija_.htm

Интерфейс параметров отрисовки размера.

Иерархия:

IDispatch

IDimensionParams

Примечание:

Является дополнительным интерфейсом для 2D и 3D размеров (интерфейсы IAngleDimension3D, IAngleDimension, IArcDimension, IBaseLineDimension3D, IBreakAngleDimension, IBreakLineDimension, IDiametralDimension3D, IDiametralDimension, IHeightDimension, ILineDimension3D, ILineDimension, IRadialDimension3D, IRadialDimension). Позволяет сформировать нужный текст размерной надписи.

IDimensionParams – свойства

ArrowPos – Размещение стрелок

Интерфейс...

Тип данных: Из перечисления ksDimensionArrowPosEnum

Синтаксис Automation:

ArrowPos = Object.ArrowPos	Получить свойство(*)
Object.ArrowPos = ArrowPos	Установить свойство (*)

```
ArrowPos = Object.GetArrowPos()   Получить свойство (**)
Object.SetArrowPos( ArrowPos )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ArrowPos(              Получить свойство(*
&ArrowPos )                       )
Object.put_ArrowPos(              Установить свойство
ArrowPos )                        (* )
```

Примечание:

Свойство позволяет выбрать способ размещения стрелок относительно выносных линий: внутри, снаружи или автоматическое определение положения. Вариант автоматического определения положения означает, что система будет для каждого размера автоматически определять, ставить ли стрелки изнутри или снаружи. Умолчательное размещение определяется настройкой стрелок, сделанной для текущего документа.

ArrowType1 – Тип стрелки у первой выносной линии

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

```
ArrowType1 = Object.ArrowType1    Получить свойство(* )
Object.ArrowType1 = ArrowType1    Установить свойство (* )
ArrowType1 =                      Получить свойство (**)
Object.GetArrowType1()
Object.SetArrowType1(             Установить свойство (**)
ArrowType1 )
```

Синтаксис COM:

```
Object.get_ArrowType1(            Получить свойство(* )
&ArrowType1 )
Object.put_ArrowType1(            Установить свойство (* )
ArrowType1 )
```

Примечание:

Позволяют задать вид стрелки у первой выносной линии.

ArrowType2 – Тип стрелки у второй выносной линии

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

```
ArrowType2 = Object.ArrowType2      Получить свойство(* )
Object.ArrowType2 = ArrowType2      Установить свойство (* )
ArrowType2 =                        Получить свойство (**)
Object.GetArrowType2()
Object.SetArrowType2(               Установить свойство (**)
ArrowType2 )
```

Синтаксис COM:

```
Object.get_ArrowType2(              Получить свойство(* )
&ArrowType2 )
Object.put_ArrowType2(              Установить свойство (* )
ArrowType2 )
```

Примечание:

Позволяют задать вид стрелки у второй выносной линии.

Gap – Зазор или длина выносной линии

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/193_24_1_4_Formirovanie_zazora_.htm`

Тип данных: BOOL

Синтаксис Automation:

```
Gap = Object.Gap                    Получить свойство(* )
Object.Gap = Gap                    Установить свойство (* )
Gap = Object.GetGap()               Получить свойство (**)
Object.SetGap( Gap )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Gap( &Gap )              Получить свойство(* )
Object.put_Gap(  Gap )              Установить свойство (* )
```

Примечание:

1. Позволяет выбрать способа формирования зазора выносных линий. Если требуется, чтобы заданное число IDimensionParams::GapValue определяло зазор между началом выносной линии и точкой привязки размера, установите свойство в TRUE. В этом случае при изменении положения размерной линии будет изменяться длина выносной линии. Если значение должно определять длину выносной линии, установите значение свойства, равное FALSE. В этом случае при изменении положения размерной линии зазор будет

изменяться, а длина выносной линии оставаться постоянной и равной IDimensionParams::GapValue.

Если значение IDimensionParams::GapValue равно нулю, то выносные линии начинаются в точках привязки размера и могут иметь любую длину.

2. Свойство используется только в линейных размерах.

GapValue – Значение зазора или длины

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/193_24_1_4_Formirovanie_zazora_.htm

Тип данных: double

Синтаксис Automation:

GapValue = Object.GapValue	Получить свойство(*)
Object.GapValue = GapValue	Установить свойство (*)
GapValue = Object.GetGapValue()	Получить свойство (**)
Object.SetGapValue(GapValue)	Установить свойство (**)

Синтаксис COM:

Object.get_GapValue(&GapValue)	Получить свойство(*)
Object.put_GapValue(GapValue)	Установить свойство (*)

Примечание:

1. Позволяет задать значение зазора или длины выносных линий. Если требуется, чтобы заданное число определяло зазор между началом выносной линии и точкой привязки размера, установите свойство IDimensionParams::Gap в TRUE. В этом случае при изменении положения размерной линии будет изменяться длина выносной линии, а зазор - оставаться постоянным и равным заданному значению.

   Если значение должно определять длину выносной линии, установите свойство IDimensionParams::Gap в FALSE. В этом случае при изменении положения размерной линии зазор будет изменяться, а длина выносной линии оставаться постоянной и равной заданному значению.

   Если значение равно нулю, то выносные линии начинаются в точках привязки размера и могут иметь любую длину.

2. Свойство используется только в линейных размерах.

RemoteLine1 – Признак отрисовки первой выносной линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

RemoteLine1 = Object.RemoteLine1	Получить свойство(*)
Object.RemoteLine1 = RemoteLine1	Установить свойство (*)
RemoteLine1 = Object.GetRemoteLine1()	Получить свойство (**)
Object.SetRemoteLine1(RemoteLine1)	Установить свойство (**)

Синтаксис COM:

Object.get_RemoteLine1(&RemoteLine1)	Получить свойство(*)
Object.put_RemoteLine1(RemoteLine1)	Установить свойство (*)

Примечание:

Позволяет управлять отрисовкой первой выносной линии.

RemoteLine2 – Признак отрисовки второй выносной линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

RemoteLine2 = Object.RemoteLine2	Получить свойство(*)
Object.RemoteLine2 = RemoteLine2	Установить свойство (*)
RemoteLine2 = Object.GetRemoteLine2()	Получить свойство (**)
Object.SetRemoteLine2(RemoteLine2)	Установить свойство (**)

Синтаксис COM:

Object.get_RemoteLine2(&RemoteLine2)	Получить свойство(*)
Object.put_RemoteLine2(RemoteLine2)	Установить свойство (*)

Примечание:

Позволяет управлять отрисовкой второй выносной линии.

ShelfAngle – Угол наклона выносной полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ShelfAngle = Object.ShelfAngle	Получить свойство(*)
Object.ShelfAngle = ShelfAngle	Установить свойство (*)
ShelfAngle = Object.GetShelfAngle()	Получить свойство (**)
Object.SetShelfAngle(ShelfAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfAngle(&ShelfAngle)	Получить свойство(*)
Object.put_ShelfAngle(ShelfAngle)	Установить свойство (*)

Примечание:

Позволяет задать угол наклона выносной полки к оси абсцисс текущей системы координат.

При использовании свойств ShelfX и ShelfY задание данного свойства не требуется.

ShelfDirection – Направление выносной полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = Object.ShelfDirection	Получить свойство(*)
Object.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = Object.GetShelfDirection()	Получить свойство (**)
Object.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfDirection(&ShelfDirection)	Получить свойство(*)
Object.put_ShelfDirection(ShelfDirection)	Установить свойство (*)

Примечание:

1. Позволяет задать направление выносной полки.
2. Свойство используется совместно со свойством IDimensionParams::TextType.

ShelfLength – Длина ножки выносной полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ShelfLength = Object.ShelfLength     Получить свойство(* )
Object.ShelfLength = ShelfLength     Установить свойство (* )
ShelfLength =                        Получить свойство (**)
Object.GetShelfLength()
Object.SetShelfLength(               Установить свойство (**)
ShelfLength )
```

Синтаксис COM:

```
Object.get_ShelfLength(              Получить свойство(* )
&ShelfLength )
Object.put_ShelfLength(              Установить свойство (* )
ShelfLength )
```

Примечание:

Позволяет задать длину ножки выносной полки.

При использовании свойств ShelfX и ShelfY задание данного свойства не требуется.

TextBase – Параметр отрисовки текста

Интерфейс...

Тип данных: из перечисления ksDimensionBaseEnum

Синтаксис Automation:

```
TextBase = Object.TextBase           Получить свойство(* )
Object.TextBase = TextBase           Установить свойство (* )
TextBase = Object.GetTextBase()      Получить свойство (**)
Object.SetTextBase( TextBase )       Установить свойство (**)
```

Синтаксис COM:

```
Object.get_TextBase( &TextBase )     Получить свойство(* )
Object.put_TextBase(  TextBase )     Установить свойство (* )
```

Примечание:

Позволяет задать положение размерной надписи.

TextOnLine – Положение размерной надписи относительно размерной линии

Интерфейс...

Тип данных: из перечисления ksDimensionTextPosEnum

Синтаксис Automation:

TextOnLine = Object.TextOnLine	Получить свойство(*)
Object.TextOnLine = TextOnLine	Установить свойство (*)
TextOnLine = Object.GetTextOnLine()	Получить свойство (**)
Object.SetTextOnLine(TextOnLine)	Установить свойство (**)

Синтаксис COM:

Object.get_TextOnLine(&TextOnLine)	Получить свойство(*)
Object.put_TextOnLine(TextOnLine)	Установить свойство (*)

Примечание:

Позволяет задать положение размерной надписи относительно размерной линии. Умолчательное положение определяется

Настройкой положения надписи

`kompas.chm::/DLG_SETDIMTEXTPOSITION.htm`
сделанной для текущего документа.

TextPos – Условное расстояние текста от выносной линии

Интерфейс...

Тип данных: long

Значения свойства:

0	- автоматическое размещение текста,
> 0	- размещение текста на указанном расстоянии в направлении от первой точки ко второй,
< 0	- размещение текста на указанном расстоянии в направлении от второй точки к первой.

Синтаксис Automation:

```
TextPos = Object.TextPos              Получить свойство(* )
Object.TextPos = TextPos              Установить свойство (* )
TextPos = Object.GetTextPos()         Получить свойство (**)
Object.SetTextPos( TextPos )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_TextPos( &TextPos )        Получить свойство(* )
Object.put_TextPos(  TextPos )        Установить свойство (* )
```

Примечание:

1. Значение свойства задается в миллиметрах для линейных размеров и в градусах для угловых.
2. Параметр является аннотационным (не зависит от масштаба, измеряется "по бумаге").
3. Для радиального и диаметрального размера задает длину "ножки":

0	- автоматическое размещение в центре,
1	- автоматическое размещение за дугой по направлению угла,
-1	- автоматическое размещение за дугой против направления угла, откладывается от центра дуги.

TextType – Тип размещения размерной надписи

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/CM_BRACE_SHELF_COMBO.htm`

Тип данных: из перечисления ksDimensionTextTypeEnum

Синтаксис Automation:

```
TextType = Object.TextType            Получить свойство(* )
Object.TextType = TextType            Установить свойство (* )
TextType = Object.GetTextType()       Получить свойство (**)
Object.SetTextType( TextType )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_TextType( &TextType )      Получить свойство(* )
Object.put_TextType(  TextType )      Установить свойство (* )
```

Примечание:

Позволяет задать тип размещения размерной надписи: Автоматический, ручной или на полке. При размещении текста на полке можно задать также направление полки см. ShelfDirection.

IDimensionParams – методы

InitDefaultValues – Инициализация параметров умолчательными значениями

Интерфейс...

Синтаксис Automation:

BOOL InitDefaultValues()

Синтаксис COM:

HRESULT InitDefaultValues(BOOL * Result)

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	в случае неудачи.

Интерфейс IHatchParam

Справка системы КОМПАС...

KOMPAS.chm::/220_Glava_26_Shtrihovka_i_zalivka.htm

Интерфейс параметров штриховки.

Иерархия:

IDispatch

IHatchParam

Описание:

Интерфейс позволяет задавать параметры штриховки.

Примечание:

1. Интерфейс является дополнительным для IHatch и IAssociationView.
2. Интерфейс можно получить с помощью метода ICutViewParam::HatchParam.

IHatchParam – свойства

HatchColor – Цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Color = Object.Color                    Получить свойство(* )
Object.Color = Color                    Установить свойство (* )
Color = Object.GetColor()               Получить свойство (**)
Object.SetColor( Color )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Color( &Color )              Получить свойство
Object.put_Color(  Color )              Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать цвет штриховки.

HatchAngle – Угол наклона штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                    Получить свойство(* )
Object.Style = Style                    Установить свойство (* )
Style = Object.GetStyle()               Получить свойство (**)
Object.SetStyle( Style )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Style( &Style )              Получить свойство
Object.put_Style( Style )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать стиль штриховки.

HatchFileName – Имя файла для стиля штриховки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
HatchFileName = Object.HatchFileName            Получить свойство(* )
HatchFileName = Object.GetHatchFileName()       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_HatchFileName( &HatchFileName )      Получить свойство
```

Примечание:

1. Свойство позволяет получить имя файла библиотеки стилей штриховки (*.lhs).
2. Свойство доступно только для чтения.

HatchType – Тип штриховки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- область,
FALSE	- полоса.

Синтаксис Automation:

HatchType = Object.HatchType	Получить свойство(*)
Object.HatchType = HatchType	Установить свойство (*)
HatchType = Object.GetHatchType()	Получить свойство (**)
Object.SetHatchType(HatchType)	Установить свойство (**)

Синтаксис COM:

Object.get_HatchType(&HatchType)	Получить свойство
Object.put_HatchType(HatchType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип штриховки.

IsSheetAngle – Угол наклона штриховки относительно листа или границ

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- обычная штриховка (угол штриховки - постоянный),
FALSE	- угол штриховки относительно ее границ сохраняется при повороте границ (используется при изображении накатки на деталях).

Синтаксис Automation:

IsSheetAngle = Object.IsSheetAngle	Получить свойство(*)
Object.IsSheetAngle = IsSheetAngle	Установить свойство (*)

IsSheetAngle = Object.GetIsSheetAngle()	Получить свойство (**)
Object.SetIsSheetAngle(IsSheetAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_IsSheetAngle(&IsSheetAngle)	Получить свойство
Object.put_IsSheetAngle(IsSheetAngle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак, является ли угол наклона штриховки листовым.

LibraryStyleNumber – Идентификатор стиля в библиотеке

Интерфейс...

Тип данных: double

Синтаксис Automation:

LibraryStyleNumber = Object. LibraryStyleNumber	Получить свойство(*)
LibraryStyleNumber = Object.GetLibraryStyleNumber()	Получить свойство (**)

Синтаксис COM:

Object.get_LibraryStyleNumber(&LibraryStyleNumber)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Step – Шаг или масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство(*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг или масштаб штриховки.

Style – Стиль штриховки

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = Object.Style	Получить свойство(*)
Object.Style = Style	Установить свойство (*)
Style = Object.GetStyle()	Получить свойство (**)
Object.SetStyle(Style)	Установить свойство (**)

Синтаксис COM:

Object.get_Style(&Style)	Получить свойство
Object.put_Style(Style)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать стиль штриховки.
2. Системные стили штриховки берутся из перечисления ksHatchStyleEnum.

Width – Ширина полосы

Интерфейс...

Тип данных: double

Значения свойства:

TRUE	- область,
FALSE	- полоса.

Синтаксис Automation:

Width = Object.Width	Получить свойство(*)
Object.Width = Width	Установить свойство (*)
Width = Object.GetWidth()	Получить свойство (**)
Object.SetWidth(Width)	Установить свойство (**)

Синтаксис COM:

Object.get_Width(&Width)	Получить свойство
Object.put_Width(Width)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать ширину полосы штриховки.

IHatchParam – методы

AddStyleFromLibrary – Инициализация стиля штриховки из библиотеки стилей

Интерфейс...

Синтаксис Automation:

BOOL AddStyleFromLibrary(BSTR FileName, double StyleNumber);

Синтаксис COM:

HRESULT AddStyleFromLibrary(BSTR FileName, double StyleNumber, BOOL * Result);

Входные параметры:

FileName	- имя файла для стиля штриховки,
StyleNumber	- идентификатор стиля в библиотеке.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IPhantom2D

Параметры фантома 2D.

Иерархия:

IDispatch

IKompasAPIObject

IPhantom2D

Интерфейс можно получить с помощью метода:IProcess2D::Phantom2D

IPhantom2D – свойства

Angle – Угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Horizontal – True – подходим к курсору по горизонтали, False – по вертикали

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Horizontal = Object. Horizontal	Получить свойство(*)
Object. Horizontal = Horizontal	Установить свойство (*)
Horizontal = Object.Get Horizontal()	Получить свойство (**)
Object.Set Horizontal(Horizontal)	Установить свойство (**)

Синтаксис COM:

Object.get_ Horizontal(& Horizontal)	Получить свойство
Object.put_ Horizontal(Horizontal)	Установить свойство

PhantomGroup – Фантомная группа

Интерфейс...

Тип данных: Указатель на интерфейс IDrawingGroup

Синтаксис Automation:

PhantomGroup = Object.PhantomGroup	Получить свойство(*)
Object.PhantomGroup = PhantomGroup	Установить свойство (*)
PhantomGroup = Object.GetPhantomGroup()	Получить свойство (**)
Object.SetPhantomGroup(PhantomGroup)	Установить свойство (**)

Синтаксис COM:

Object.get_PhantomGroup(&PhantomGroup)	Получить свойство
Object.put_PhantomGroup(PhantomGroup)	Установить свойство

PhantomType – Тип резинки

Интерфейс...

Тип данных: из перечисления ksPhantomTypeEnum

Синтаксис Automation:

PhantomType = Object.PhantomType	Получить свойство(*)
Object.PhantomType = PhantomType	Установить свойство (*)
PhantomType = Object.GetPhantomType()	Получить свойство (**)
Object.SetPhantomType(PhantomType)	Установить свойство (**)

Синтаксис COM:

Object.get_PhantomType(&PhantomType)	Получить свойство
Object.put_PhantomType(PhantomType)	Установить свойство

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство(*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство(*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство(*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

IPhantom2D – методы

Hide – Погасить фантом

Интерфейс...

Синтаксис Automation:

BOOL Hide();

Синтаксис COM:

HRESULT Hide(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Show – Показать фантом

Интерфейс...

Синтаксис Automation:

BOOL Show();

Синтаксис COM:

HRESULT Show(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Update – Обновить фантом

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- если фантом обновился.

Интерфейс IRoughParams

Справка системы КОМПАС2D...

kompas.chm::/223_28_3_2_Nastrojka_otrisovki_.htm

Справка системы КОМПАС3D...

kompas.chm::/CM_ROUGH_3D.htm

Интерфейс параметров обозначения шероховатости.

Иерархия:

IDispatch

IRoughParams

Описание:

1. Является дополнительным интерфейсом для 2D и 3D обозначений шероховатости.
2. Позволяет задать и получить параметры обозначения шероховатости.
3. Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейсов IRough3D или IRough.

IRoughParams – свойства

ArrowInside – Расположение стрелки линии-выноски

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- внутри,
FALSE	- снаружи.

Синтаксис Automation:

ArrowInside = Object.ArrowInside	Получить свойство(*)
Object.ArrowInside = ArrowInside	Установить свойство (*)
ArrowInside = Object.GetArrowInside()	Получить свойство (**)
Object.SetArrowInside(ArrowInside)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowInside(&ArrowInside)	Получить свойство
Object.put_ArrowInside(ArrowInside)	Установить свойство

ArrowType – Тип стрелки линии-выноски

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = Object.ArrowType	Получить свойство(*)
Object.ArrowType = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType()	Получить свойство (**)
Object.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство
Object.put_ArrowType(ArrowType)	Установить свойство

Примечание:

Для обозначения шероховатости используются следующие типы стрелок:

- без стрелки;

- стрелка (внутри/снаружи);
- засечка;
- засечка с наклоном влево.

BaseLengthText – Текст "Базовая длина по ГОСТ 2789-73"

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

BaseLengthText = Object.BaseLengthText	Получить свойство(*)
BaseLengthText = Object.GetBaseLengthText()	Получить свойство (**)

Синтаксис COM:

Object.get_BaseLengthText(&BaseLengthText)	Получить свойство

Примечание:

1. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста "Базовая длина по ГОСТ 2789-73".
2. Свойство доступно только для чтения.

LeaderAngle – Угол наклона линии-выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeaderAngle = Object.LeaderAngle	Получить свойство(*)
Object.LeaderAngle = LeaderAngle	Установить свойство (*)
LeaderAngle = Object.GetLeaderAngle()	Получить свойство (**)
Object.SetLeaderAngle(LeaderAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_Leader Angle(&LeaderAngle)	Получить свойство
Object.put_Leader Angle(LeaderAngle)	Установить свойство

LeaderLength – Длина линии-выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeaderLength = Object.LeaderLength	Получить свойство(*)
Object.LeaderLength = LeaderLength	Установить свойство (*)
LeaderLength = Object.GetLeaderLength()	Получить свойство (**)
Object.SetLeaderLength(LeaderLength)	Установить свойство (**)

Синтаксис COM:

Object.get_Leader Length(&LeaderLength)	Получить свойство
Object.put_Leader Length(LeaderLength)	Установить свойство

ProcessingByContour- Обработка по контуру

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- обработка по контуру,
FALSE	- не по контуру.

Синтаксис Automation:

ProcessingByContour = Object.ProcessingByContour	Получить свойство(*)

Object.ProcessingByContour = ProcessingByContour	Установить свойство (*)
ProcessingByContour = Object.GetProcessingByContour()	Получить свойство (**)
Object.SetProcessingByContour(ProcessingByContour)	Установить свойство (**)

Синтаксис COM:

Object.get_ProcessingByContour(&ProcessingByContour)	Получить свойство
Object.put_ProcessingByContour(ProcessingByContour)	Установить свойство

ProcessText – Текст "Способ обработки поверхности и (или) другие дополнительные указания"

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

ProcessText = Object.ProcessText	Получить свойство(*)
ProcessText = Object.GetProcessText()	Получить свойство (**)

Синтаксис COM:

Object.get_ProcessText(&ProcessText)	Получить свойство

Примечание:

1. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста "Способ обработки поверхности и (или) другие дополнительные указания".
2. Свойство доступно только для чтения.

RoughParamText – Текст "Параметр (параметры) шероховатости по ГОСТ 2789–73"

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

RoughParamText = Object.RoughParamText	Получить свойство(*)
RoughParamText = Object.GetRoughParamText()	Получить свойство (**)

Синтаксис COM:

Object.get_RoughParamText(&RoughParamText)	Получить свойство

Примечание:

1. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста "Параметр (параметры) шероховатости по ГОСТ 2789-73".
2. Свойство доступно только для чтения.

ShelfDirection – Направление полки

Интерфейс...

Тип данных: из перечисления ksShelfDirectionEnum

Синтаксис Automation:

ShelfDirection = Object.ShelfDirection	Получить свойство(*)
Object.ShelfDirection = ShelfDirection	Установить свойство (*)
ShelfDirection = Object.GetShelfDirection()	Получить свойство (**)
Object.SetShelfDirection(ShelfDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ShelfDirection(&ShelfDirection)	Получить свойство
Object.put_ShelfDirection(ShelfDirection)	Установить свойство

SignType – Тип значка

Интерфейс...

Тип данных: из перечисления ksRoughSignEnum.

Синтаксис Automation:

SignType = Object.SignType	Получить свойство(*)
Object.SignType = SignType	Установить свойство (*)
SignType = Object.GetSignType()	Получить свойство (**)
Object.SetSignType(SignType)	Установить свойство (**)

Синтаксис COM:

Object.get_SignType(&SignType)	Получить свойство
Object.put_SignType(SignType)	Установить свойство

TrendText – Текст "Условное обозначение направления неровностей"

Интерфейс...

Тип данных: указатель на интерфейс IText

Синтаксис Automation:

TrendText = Object.TrendText	Получить свойство(*)
TrendText = Object.GetTrendText()	Получить свойство (**)

Синтаксис COM:

Object.get_TrendText(&TrendText)	Получить свойство

Примечание:

1. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста "Условное обозначение направления неровностей".
2. Свойство доступно только для чтения.

Интерфейс IToleranceParam

Справка системы КОМПАС...

kompas.chm::/CM_FORMTOLERANCE.htm

Справка системы КОМПАС...

kompas.chm::/CM_TOLERANCE_3D.htm

Интерфейс параметров обозначения допуска формы.

Иерархия:

IDispatch

IToleranceParam

Описание:

1. Является дополнительным интерфейсом для 2D и 3D обозначений допуска формы.
2. Позволяет задать и получить параметры обозначения допуска формы.
3. Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейсов ITolerance и ITolerance3D.

IToleranceParam – свойства

BasePointPos – Базовая точка

Интерфейс...

Тип данных: из перечисления ksTablePointEnum

Синтаксис Automation:

BasePointPos = Object.BasePointPos	Получить свойство(*)
Object.BasePointPos = BasePointPos	Установить свойство (*)
BasePointPos = Object.GetBasePointPos( )	Получить свойство (**)
Object.SetBasePointPos(BasePointPos)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePointPos(&BasePointPos)	Получить свойство
Object.put_BasePointPos(BasePointPos)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип расположения базовой точки.

BaseSign1 – Знак после базы 1

Интерфейс...

Тип данных: из перечисления ksToleranceSuffixSignEnum

Синтаксис Automation:

BaseSign1 = Object.BaseSign1	Получить свойство(*)
Object.BaseSign1 = BaseSign1	Установить свойство (*)
BaseSign1 = Object.GetBaseSign1( )	Получить свойство (**)
Object.SetBaseSign1(BaseSign1)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseSign2(&BaseSign2)	Получить свойство
Object.put_BaseSign1(BaseSign1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать знак после базы 1.

BaseSign2 – Знак после базы 2

Интерфейс...

Тип данных: из перечисления ksToleranceSuffixSignEnum

Синтаксис Automation:

BaseSign2 = Object.BaseSign2	Получить свойство(*)
Object.BaseSign2 = BaseSign2	Установить свойство (*)
BaseSign2 = Object.GetBaseSign2( )	Получить свойство (**)
Object.SetBaseSign2(BaseSign2)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseSign2(&BaseSign2)	Получить свойство
Object.put_BaseSign2(BaseSign2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать знак после базы 2.

BaseValue1 – Значение базы 1

Интерфейс...

Тип данных: указатель на интерфейс ITextLine

Синтаксис Automation:

BaseValue1 = Object.BaseValue1	Получить свойство(*)
BaseValue1 = Object.GetBaseValue1()	Получить свойство (**)

Синтаксис COM:

Object.get_BaseValue1(&BaseValue1)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текстовой строки со значением базы 1.
2. Свойство доступно только для чтения.

BaseValue2 – Значение базы 2

Интерфейс...

Тип данных: указатель на интерфейс ITextLine

Синтаксис Automation:

BaseValue2 = Object.BaseValue1	Получить свойство(*)
BaseValue2 = Object.GetBaseValue2()	Получить свойство (**)

Синтаксис COM:

Object.get_BaseValue2(&BaseValue2)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс текстовой строки со значением базы 2.
2. Свойство доступно только для чтения.

PrefixSign – Знак перед числовым значением

Интерфейс...

Тип данных: из перечисления ksTolerancePrefixSignEnum

Синтаксис Automation:

PrefixSign = Object.PrefixSign	Получить свойство(*)
Object.PrefixSign = PrefixSign	Установить свойство (*)
PrefixSign = Object.GetPrefixSign( )	Получить свойство (**)
Object.SetPrefixSign(PrefixSign)	Установить свойство (**)

Синтаксис COM:

Object.get_PrefixSign(&PrefixSign)	Получить свойство
Object.put_PrefixSign(PrefixSign)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать знак перед числовым значением допуска формы.

Sign – Номер спецзнака

Интерфейс...

Тип данных: long

Синтаксис Automation:

Sign = Object.Sign	Получить свойство(*)
Object.Sign = Sign	Установить свойство (*)
Sign = Object.GetSign( )	Получить свойство (**)
Object.SetSign(Sign)	Установить свойство (**)

Синтаксис COM:

Object.get_Sign(&Sign)	Получить свойство
Object.put_Sign(Sign)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать номер спецзнака. Пример спецзнаков: см. файл SDK\NumbSymb.frw. Коды спецзнаков хранятся в файле Sys\GRAPHIC.SSS.

SuffixSign – Знак после числового значения

Интерфейс...

Тип данных: из перечисления ksToleranceSuffixSignEnum

Синтаксис Automation:

SuffixSign = Object.SuffixSign	Получить свойство(*)
Object.SuffixSign = SuffixSign	Установить свойство (*)
SuffixSign = Object.GetSuffixSign( )	Получить свойство (**)
Object.SetSuffixSign(SuffixSign)	Установить свойство (**)

Синтаксис COM:

Object.get_SuffixSign(&SuffixSign)	Получить свойство
Object.put_SuffixSign(SuffixSign)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать знак после числового значения допуска формы.

Table – Текст надписи

Интерфейс...

Тип данных: указатель на интерфейс ITable

Синтаксис Automation:

Table = Object.Table	Получить свойство(*)
Table = Object.GetTable()	Получить свойство (**)

Синтаксис COM:

Object.get_Table(&Table)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс таблицы, содержащей текст допуска формы.
2. Свойство доступно только для чтения.

Value – Строка с числовым значением

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Value = Object.Value	Получить свойство(*)

Object.Value = Value	Установить свойство (*)
Value = Object.GetValue( )	Получить свойство (**)
Object.SetValue(Value)	Установить свойство (**)

Синтаксис COM:

Object.get_Value(&Value)	Получить свойство
Object.put_Value(Value)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать строку с числовым значением допуска формы.

Vertical – Вертикальное расположение объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Vertical = Object.Vertical	Получить свойство(*)
Object.Vertical = Vertical	Установить свойство (*)
Vertical = Object.GetVertical( )	Получить свойство (**)
Object.SetVertical(Vertical)	Установить свойство (**)

Синтаксис COM:

Object.get_Vertical (&BasePointPos)	Получить свойство
Object.put_Vertical(BasePointPos)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак вертикального расположения объекта.

Интерфейс ICopyObjectParam

Интерфейс параметров копирования объектов.

Иерархия:

IDispatch

IKompasAPIObject

ICopyObjectParam

ICurveCopyObjectParam

ICopyObjectParam1

Описание:

Интерфейс позволяет задавать и получать параметры копирования объектов.

Примечание:

1. Данный интерфейс можно получить с помощью метода IKompasDocument1::GetInterface.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы ICurveCopyObjectParam и ICopyObjectParam1.

ICopyObjectParam – свойства

Angle – Угол поворота

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство(*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол поворота в градусах.

AttributeCopy – Признак копирования атрибутов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AttributeCopy = Object.AttributeCopy	Получить свойство(*)
Object.AttributeCopy = AttributeCopy	Установить свойство (*)
AttributeCopy = Object.GetAttributeCopy()	Получить свойство (**)
Object.SetAttributeCopy(AttributeCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_AttributeCopy(&AttributeCopy)	Получить свойство
Object.put_AttributeCopy(AttributeCopy)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак копирования атрибутов.

DimensionLineScale – Масштабировать выносные линии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DimensionLineScale = Object.DimensionLineScale	Получить свойство(*)
Object.DimensionLineScale = DimensionLineScale	Установить свойство (*)
DimensionLineScale = Object.GetDimensionLineScale()	Получить свойство (**)
Object.SetDimensionLineScale(DimensionLineScale)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionLineScale(&DimensionLineScale)	Получить свойство
Object.put_DimensionLineScale(DimensionLineScale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак масштабирования выносных линий.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство(*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать масштаб копии.

SpecificationObjectCopy – Копировать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SpecificationObjectCopy = Object.SpecificationObjectCopy	Получить свойство(*)
Object.SpecificationObjectCopy = SpecificationObjectCopy	Установить свойство (*)
SpecificationObjectCopy = Object.GetSpecificationObjectCopy()	Получить свойство (**)
Object.SetSpecificationObjectCopy(SpecificationObjectCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_SpecificationObjectCopy(&SpecificationObjectCopy)	Получить свойство
Object.put_SpecificationObjectCopy(SpecificationObjectCopy)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак копирования объектов спецификации.

XOld – Координата базовой точки объекта по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

XOld = Object.XOld	Получить свойство(*)
Object.XOld = XOld	Установить свойство (*)
XOld = Object.GetXOld()	Получить свойство (**)
Object.SetXOld(XOld)	Установить свойство (**)

Синтаксис COM:

Object.get_XOld(&XOld)	Получить свойство
Object.put_XOld(XOld)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки объекта по оси X.

XNew – Координата новой точки привязки объекта по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

XNew = Object.XNew	Получить свойство(*)
Object.XNew = XNew	Установить свойство (*)
XNew = Object.GetXNew()	Получить свойство (**)
Object.SetXNew(XNew)	Установить свойство (**)

Синтаксис COM:

Object.get_XNew(&XNew)	Получить свойство
Object.put_XNew(XNew)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату новой точки привязки объекта по оси X.

YOld – Координата базовой точки объекта по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

YOld = Object.YOld	Получить свойство(*)
Object.YOld = YOld	Установить свойство (*)
YOld = Object.GetYOld()	Получить свойство (**)
Object.SetYOld(YOld)	Установить свойство (**)

Синтаксис COM:

Object.get_YOld(&YOld)	Получить свойство
Object.put_YOld(YOld)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату базовой точки объекта по оси Y.

YNew – Координата новой точки привязки объекта по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

YNew = Object.YNew	Получить свойство(*)
Object.YNew = YNew	Установить свойство (*)
YNew = Object.GetYNew()	Получить свойство (**)
Object.SetYNew(YNew)	Установить свойство (**)

Синтаксис COM:

Object.get_YNew(&YNew)	Получить свойство
Object.put_YNew(YNew)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату новой точки привязки объекта по оси Y.

Интерфейс ICircleCopyObjectParam

Справка системы КОМПАС...

`kompas.chm::/make_copy.htm#copy_of_circle`

Интерфейс параметров копирования объектов по окружности.

Иерархия:

IKompasAPIObject

ICopyObjectParam

ICircleCopyObjectParam

Примечание:

Интерфейс можно получить с помощью функции IKompasDocument1::GetInterface.

ICircleCopyObjectParam – свойства

ByStep – Способ определения шага: заданный или расчетный

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- с заданным шагом,
FALSE	- шаг рассчитывается по количеству копий.

Синтаксис Automation:

ByStep = Object.ByStep	Получить свойство (*)
Object.ByStep = ByStep	Установить свойство (**)
ByStep = Object.GetByStep()	Получить свойство (**)
Object.SetByStep(ByStep)	Установить свойство (**)

Синтаксис COM:

Object.get_ByStep(&ByStep)	Получить свойство
Object.put_ByStep(ByStep)	Установить свойство

Count – Количество копий

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Object.Count = Count	Установить свойство (**)
Count = Object.GetCount()	Получить свойство (**)
Object.SetCount(Count)	Установить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство
Object.put_Count(Count)	Установить свойство

Свойство позволяет устанавливать и получать количество копий объекта.

PositiveDirection – Положительное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PositiveDirection = Object.PositiveDirection	Получить свойство (*)
Object.PositiveDirection = PositiveDirection	Установить свойство (**)
PositiveDirection = Object.GetPositiveDirection()	Получить свойство (**)
Object.SetPositiveDirection(PositiveDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_PositiveDirection(&PositiveDirection)	Получить свойство
Object.put_PositiveDirection(PositiveDirection)	Установить свойство

Свойство позволяет устанавливать и получать направление копирования.

Step – Шаг копирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (**)
Step = Object.GetStep( )	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Step( &Step )      Получить свойство
Object.put_Step( Step )       Установить свойство
```

Свойство позволяет устанавливать и получать шаг копирования объектов.

Xc – Координата центра по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = Object.Xc                Получить свойство (* )
Object.Xc = Xc                Установить свойство (**)
Xc = Object.GetXc()           Получить свойство (**)
Object.SetXc( Xc )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Xc( &Xc )          Получить свойство
Object.put_Xc( Xc )           Установить свойство
```

Свойство позволяет устанавливать и получать координату центра по X.

Yc – Координата центра по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = Object.Yc                Получить свойство (* )
Object.Yc = Yc                Установить свойство (**)
Yc = Object.GetYc()           Получить свойство (**)
Object.SetYc( Yc )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Yc( &Yc )          Получить свойство
```

Object.put_Yc(Yc)	Установить свойство

Свойство позволяет устанавливать и получать координату центра по Y.

Интерфейс ICircularCopyObjectParam

Справка системы КОМПАС...

kompas.chm::/make_copy.htm#copy_of_concentric

Параметры копирования объектов по концентрической сетке

Иерархия:

IKompasAPIObject

ICopyObjectParam

ICircularCopyObjectParam

Примечание:

Интерфейс можно получить с помощью функции IKompasDocument1::GetInterface.

ICircularCopyObjectParam – свойства

Angle2 – Начальный угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство (*)
Object.Angle2 = Angle2	Установить свойство (**)
Angle2 = Object.GetAngle2()	Получить свойство (**)
Object.SetAngle2(Angle2)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle2(&Angle2)	Получить свойство
Object.put_Angle2(Angle2)	Установить свойство

Свойство позволяет устанавливать и получать начальный угол.

Count1 – Количество копий в радиальном направлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count1 = Object.Count1	Получить свойство (*)
Object.Count1 = Count1	Установить свойство (**)
Count1 = Object.GetCount1()	Получить свойство (**)
Object.SetCount1(Count1)	Установить свойство (**)

Синтаксис COM:

Object.get_Count1(&Count1)	Получить свойство
Object.put_Count1(Count1)	Установить свойство

Свойство позволяет устанавливать и получать количество копий в радиальном направлении.

Count2 – Количество копий в кольцевом направлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count2 = Object.Count2	Получить свойство (*)
Object.Count2 = Count2	Установить свойство (**)
Count2 = Object.GetCount2()	Получить свойство (**)
Object.SetCount2(Count2)	Установить свойство (**)

Синтаксис COM:

Object.get_Count2(&Count2)	Получить свойство

Object.put_Count2(Count2)	Установить свойство

Свойство позволяет устанавливать и получать количество копий в кольцевом направлении.

Radius – Начальный радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство (*)
Object.Radius = Radius	Установить свойство (**)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство
Object.put_Radius(Radius)	Установить свойство

Свойство позволяет устанавливать и получать начальный радиус сетки.

SaveCentreCopy – Признак наличия копии в центре сетки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- оставлять копию в центре сетки,
FALSE	- не оставлять копию в центре сетки.

Синтаксис Automation:

SaveCentreCopy = Object.SaveCentreCopy	Получить свойство (*)
Object.SaveCentreCopy = SaveCentreCopy	Установить свойство (**)

SaveCentreCopy = Object.GetSaveCentreCopy ()	Получить свойство (**)
Object.SetSaveCentreCopy (SaveCentreCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveCentreCopy(&SaveCentreCopy)	Получить свойство
Object.put_SaveCentreCopy(SaveCentreCopy)	Установить свойство

Свойство позволяет оставлять копию в центре сетки.

Step1 – Шаг копирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step1 = Object.Step1	Получить свойство (*)
Object.Step1 = Step1	Установить свойство (**)
Step1 = Object.GetStep1()	Получить свойство (**)
Object.SetStep1(Step1)	Установить свойство (**)

Синтаксис COM:

Object.get_Step1(&Step1)	Получить свойство
Object.put_Step1(Step1)	Установить свойство

Свойство позволяет устанавливать и получать шаг копирования в радиальном направлении.

Step2 – Угловой шаг (градусы)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step2 = Object.Step2	Получить свойство (*)
Object.Step2 = Step2	Установить свойство (**)
Step2 = Object.GetStep2()	Получить свойство (**)
Object.SetStep2(Step2)	Установить свойство (**)

Синтаксис COM:

Object.get_Step2(&Step2)	Получить свойство
Object.put_Step2(Step2)	Установить свойство

Свойство позволяет устанавливать и получать угловой шаг копирования в градусах.

StepFactor1 – Интерпретация шага в радиальном направлении: расстояние между соседними копиями или между крайними

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- расстояние между соседними копиями,
FALSE	- расстояние между крайними копиями.

Синтаксис Automation:

StepFactor1 = Object.StepFactor1	Получить свойство (*)
Object.StepFactor1 = StepFactor1	Установить свойство (**)
StepFactor1 = Object.GetStepFactor1()	Получить свойство (**)
Object.SetStepFactor1(StepFactor1)	Установить свойство (**)

Синтаксис COM:

Object.get_StepFactor1(&StepFactor1)	Получить свойство
Object.put_StepFactor1(StepFactor1)	Установить свойство

Свойство позволяет устанавливать и получать способ задания шага в радиальном направлении.

StepFactor2 – Интерпретация шага в кольцевом направлении: угол между соседними копиями или между крайними

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- угол между соседними копиями,
FALSE	- угол между крайними копиями.

Синтаксис Automation:

StepFactor2 = Object.StepFactor2	Получить свойство (*)
Object.StepFactor2 = StepFactor2	Установить свойство (**)
StepFactor2 = Object.GetStepFactor2()	Получить свойство (**)
Object.SetStepFactor2(StepFactor2)	Установить свойство (**)

Синтаксис COM:

Object.get_StepFactor2(&StepFactor2)	Получить свойство
Object.put_StepFactor2(StepFactor2)	Установить свойство

Свойство позволяет устанавливать и получать способ задания углового шага.

TurnObject – Признак доворота копий до радиального направления

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- доворачивать копии до радиального направления,
FALSE	- не доворачивать копии до радиального направления.

Синтаксис Automation:

TurnObject = Object.TurnObject	Получить свойство (*)
Object.TurnObject = TurnObject	Установить свойство (**)
TurnObject = Object.GetTurnObject()	Получить свойство (**)
Object.SetTurnObject(TurnObject)	Установить свойство (**)

Синтаксис COM:

Object.get_TurnObject(&TurnObject)	Получить свойство
Object.put_TurnObject(TurnObject)	Установить свойство

Свойство позволяет доворачивать копии до радиального направления.

Интерфейс ICurveCopyObjectParam

Интерфейс параметров копирования объектов вдоль кривой.

Иерархия:

IDispatch

IKompasAPIObject

ICurveCopyObjectParam

Описание:

Интерфейс позволяет задавать и получать параметры копирования объектов вдоль кривой.

Примечание:

Интерфейс является дополнительным для интерфейса ICopyObjectParam. Он может быть получен с помощью метода IUnknown::QueryInterface.

ICurveCopyObjectParam – свойства

BaseCurve – Базовая кривая

Интерфейс...

Тип данных: указатель на интерфейс IDrawingObject

Синтаксис Automation:

```
BaseCurve = Object.BaseCurve                Получить свойство (* )
Object.BaseCurve = BaseCurve                Установить свойство (**)
BaseCurve                    =              Получить свойство (**)
Object.GetBaseCurve()
Object.SetBaseCurve( BaseCurve )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BaseCurve( &BaseCurve )          Получить свойство
Object.put_BaseCurve( BaseCurve )           Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать базовую кривую.

Count – Количество копий

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Count = Object.Count                        Получить свойство (* )
Object.Count = Count                        Установить свойство (**)
Count = Object.GetCount()                   Получить свойство (**)
Object.SetCount( Count )                    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Count( &Count )                  Получить свойство
Object.put_Count(  Count )                  Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать количество копий.

PositiveDirection – Положительное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PositiveDirection = Object.PositiveDirection	=	Получить свойство (*)
Object.PositiveDirection = PositiveDirection	=	Установить свойство (**)
PositiveDirection = Object.GetPositiveDirection()	=	Получить свойство (**)
Object.SetPositiveDirection(PositiveDirection)		Установить свойство (**)

Синтаксис COM:

Object.get_PositiveDirection(&PositiveDirection)	Получить свойство
Object.put_PositiveDirection(PositiveDirection)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак положительного направления.

Step – Шаг

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (**)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг копирования.

StepFactor – Способ задания шага

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- шаг задан между крайними копиями,
FALSE	- шаг задан между соседними копиями.

Синтаксис Automation:

StepFactor = Object.StepFactor	Получить свойство (*)
Object.StepFactor = StepFactor	Установить свойство (**)
StepFactor = Object.GetStepFactor()	Получить свойство (**)
Object.SetStepFactor(StepFactor)	Установить свойство (**)

Синтаксис COM:

Object.get_StepFactor(&StepFactor)	Получить свойство
Object.put_StepFactor(StepFactor)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ задания шага.

TurnToNormal – Доворачивать до нормали

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TurnToNormal = Object.TurnToNormal	Получить свойство (*)
Object.TurnToNormal = TurnToNormal	Установить свойство (**)
TurnToNormal = Object.GetTurnToNormal()	Получить свойство (**)
Object.SetTurnToNormal(TurnToNormal)	Установить свойство (**)

Синтаксис COM:

Object.get_TurnToNormal(&TurnToNormal)	Получить свойство
Object.put_TurnToNormal(TurnToNormal)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак доворачивания до нормали.

Интерфейс IMeshCopyObjectParam

Справка системы КОМПАС...

kompas.chm::/make_copy.htm#copy_of_parallelogram

Параметры копирования объектов по параллелограммной сетке

Иерархия:

IKompasAPIObject

ICopyObjectParam

IMeshCopyObjectParam

Примечание:

Интерфейс можно получить с помощью функции IKompasDocument1::GetInterface.

IMeshCopyObjectParam – свойства

Angle1 – Угол наклона первой оси сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle1 = Object.Angle1	Получить свойство (*)
Object.Angle1 = Angle1	Установить свойство (**)
Angle1 = Object.GetAngle1()	Получить свойство (**)
Object.SetAngle1(Angle1)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle1(&Angle1)	Получить свойство
Object.put_Angle1(Angle1)	Установить свойство

Свойство позволяет устанавливать и получать угол наклона первой оси сетки.

Angle2 – Угол между первой и второй осями сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство (*)
Object.Angle2 = Angle2	Установить свойство (**)
Angle2 = Object.GetAngle2()	Получить свойство (**)
Object.SetAngle2(Angle2)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Angle2( &Angle2 )        Получить свойство
Object.put_Angle2( Angle2 )         Установить свойство
```

Свойство позволяет устанавливать и получать угол между первой и второй осями сетки.

Count1 – Количество копий по оси 1

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Count1 = Object.Count1              Получить свойство (* )
Object.Count1 = Count1              Установить свойство (**)
Count1 = Object.GetCount1()         Получить свойство (**)
Object.SetCount1( Count1 )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Count1( &Count1 )        Получить свойство
Object.put_Count1( Count1 )         Установить свойство
```

Свойство позволяет устанавливать и получать количество копий по первой оси.

Count2 – Количество копий по оси 2

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Count2 = Object.Count2              Получить свойство (* )
Object.Count2 = Count2              Установить свойство (**)
Count2 = Object.GetCount2()         Получить свойство (**)
Object.SetCount2( Count2 )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Count2( &Count2 )        Получить свойство
Object.put_Count2( Count2 )         Установить свойство
```

Свойство позволяет устанавливать и получать количество копий по второй оси.

SaveCentreCopy – Признак наличия копий внутри сетки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- оставлять копию в центре сетки,
FALSE	- оставлять копию в центре сетки.

Синтаксис Automation:

SaveCentreCopy = Object.SaveCentreCopy	Получить свойство (*)
Object.SaveCentreCopy = SaveCentreCopy	Установить свойство (**)
SaveCentreCopy = Object.GetSaveCentreCopy()	Получить свойство (**)
Object.SetSaveCentreCopy(SaveCentreCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveCentreCopy(&SaveCentreCopy)	Получить свойство
Object.put_SaveCentreCopy(SaveCentreCopy)	Установить свойство

Свойство позволяет оставлять копии внутри сетки.

SaveCornersCopy – Признак наличия копий в углах сетки

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- оставлять копии в углах сетки,
FALSE	- не оставлять копии в углах сетки.

Синтаксис Automation:

SaveCornersCopy = Object.SaveCornersCopy	Получить свойство (*)
Object.SaveCornersCopy = SaveCornersCopy	Установить свойство (**)
SaveCornersCopy = Object.GetSaveCornersCopy()	Получить свойство (**)
Object.SetSaveCornersCopy(SaveCornersCopy)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveCornersCopy(&SaveCornersCopy)	Получить свойство
Object.put_SaveCornersCopy(SaveCornersCopy)	Установить свойство

Свойство позволяет оставлять копии в углах сетки.

Step1 – Шаг копирования по оси 1

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step1 = Object.Step1                    Получить свойство (* )
Object.Step1 = Step1                    Установить свойство (**)
Step1 = Object.GetStep1()               Получить свойство (**)
Object.SetStep1( Step1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step1( &Step1 )              Получить свойство
Object.put_Step1( Step1 )               Установить свойство
```

Свойство позволяет устанавливать и получать шаг копирования по первой оси.

Step2 – Шаг копирования по оси 2

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step2 = Object.Step2                    Получить свойство (* )
Object.Step2 = Step2                    Установить свойство (**)
Step2 = Object.GetStep2()               Получить свойство (**)
Object.SetStep2( Step2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step2( &Step2 )              Получить свойство
Object.put_Step2( Step2 )               Установить свойство
```

Свойство позволяет устанавливать и получать шаг копирования по второй оси.

StepFactor1 – Интерпретация шага по первой оси: расстояние между соседними копиями или между крайними копиями

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE                                    - расстояние между соседними копиями,
FALSE                                   - расстояние между крайними копиями.
```

Синтаксис Automation:

```
StepFactor1 = Object.StepFactor1        Получить свойство (* )
Object.StepFactor1 = StepFactor1        Установить свойство (**)
```

StepFactor1 = Object.GetStepFactor1() Получить свойство (**)
Object.SetStepFactor1(StepFactor1) Установить свойство (**)

Синтаксис COM:

Object.get_StepFactor1(&StepFactor1) Получить свойство
Object.put_StepFactor1(StepFactor1) Установить свойство

Свойство позволяет устанавливать и получать способ задания шага по первой оси.

StepFactor2 – Интерпретация шага по второй оси: расстояние между соседними копиями или между крайними копиями

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - расстояние между соседними копиями,

FALSE - расстояние между крайними копиями.

Синтаксис Automation:

StepFactor2 = Object.StepFactor2 Получить свойство (*)
Object.StepFactor2 = StepFactor2 Установить свойство (**)
StepFactor2 = Object.GetStepFactor2() Получить свойство (**)
Object.SetStepFactor2(StepFactor2) Установить свойство (**)

Синтаксис COM:

Object.get_StepFactor2(&StepFactor2) Получить свойство
Object.put_StepFactor2(StepFactor2) Установить свойство

Свойство позволяет устанавливать и получать способ задания шага по второй оси.

Поиск объектов

Интерфейс IFindObjectParameters

Параметры поиска объектов.

Иерархия:

IDispatch

IKompasAPIObject

IFindObjectParameters

KOMPAS v19

IFindObjectParameters – свойства

DisabledObjects – Игнорируемые объекты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
DisabledObjects = Object.DisabledObjects                Получить свойство (* )
Object.DisabledObjects = DisabledObjects                Установить свойство (**)
DisabledObjects = Object.GetDisabledObjects()           Получить свойство (**)
Object.SetDisabledObjects( DisabledObjects )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DisabledObjects( &DisabledObjects )          Получить свойство
Object.put_DisabledObjects(  DisabledObjects )          Установить свойство
```

Примечание.

1. Позволяет получать и устанавливать объект или список объектов, игнорируемые при поиске.
2. Массив возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

DisabledViews – Игнорируемые виды

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
DisabledViews = Object.DisabledViews                    Получить свойство (* )
Object.DisabledViews = DisabledViews                    Установить свойство (**)
DisabledViews = Object.GetDisabledViews()               Получить свойство (**)
Object.SetDisabledViews( DisabledViews )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DisabledViews( &DisabledViews )              Получить свойство
Object.put_DisabledViews(  DisabledViews )              Установить свойство
```

Примечание.

1. Позволяет получать и устанавливать вид или список видов, игнорируемые при поиске.
2. Массив возвращается в виде массива SAFEARRAY видов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

DrawingObjectType – Фильтрация по типу графического объекта

Интерфейс...

Тип данных: из перечисления DrawingObjectTypeEnum

Синтаксис Automation:

Object.DrawingObjectType DrawingObjectType	=	Получить свойство (*)
Object.FindObjectsType = FindObjectsType		Установить свойство (**)
DrawingObjectType Object.GetDrawingObjectType()	=	Получить свойство (**)
Object.SetDrawingObjectType(DrawingObjectType)		Установить свойство (**)

Синтаксис COM:

Object.get_DrawingObjectType(&DrawingObjectType)	Получить свойство
Object.put_DrawingObjectType(DrawingObjectType)	Установить свойство

FindInBackgroundViewsAndLayers – Искать в фоновых видах и слоях

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FindInBackgroundViewsAndLayers Object.FindInBackgroundViewsAndLayers	=	Получить свойство (*)
Object.FindInBackgroundViewsAndLayers FindInBackgroundViewsAndLayers	=	Установить свойство (**)
FindInBackgroundViewsAndLayers Object.GetFindInBackgroundViewsAndLayers()	=	Получить свойство (**)
Object.SetFindInBackgroundViewsAndLayers(FindInBackgroundViewsAndLayers)		Установить свойство (**)

Синтаксис COM:

Object.get_FindInBackgroundViewsAndLayers(&FindInBackgroundViewsAndLayers)	Получить свойство
Object.put_FindInBackgroundViewsAndLayers(FindInBackgroundViewsAndLayers)	Установить свойство

FindObjectsType – Тип поиска объектов

Интерфейс...

Тип данных: из перечисления ksFindObjectsTypeEnum

Синтаксис Automation:

FindObjectsType = Object.FindObjectsType	Получить свойство (*)
Object.FindObjectsType = FindObjectsType	Установить свойство (**)
FindObjectsType = Object.GetFindObjectsType()	Получить свойство (**)
Object.SetFindObjectsType(FindObjectsType)	Установить свойство (**)

Синтаксис COM:

Object.get_FindObjectsType(&FindObjectsType)	Получить свойство
Object.put_FindObjectsType(FindObjectsType)	Установить свойство

GeometryOnly – Искать только среди геометрических объектов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

GeometryOnly = Object.GeometryOnly	Получить свойство (*)
Object.GeometryOnly = GeometryOnly	Установить свойство (**)
GeometryOnly = Object.GetGeometryOnly()	Получить свойство (**)
Object.SetGeometryOnly(GeometryOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_GeometryOnly(&GeometryOnly)	Получить свойство
Object.put_GeometryOnly(GeometryOnly)	Установить свойство

IFindObjectParameters – методы

Clear – Очистить параметры поиска

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM :

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Таблица

Интерфейс ITable

Справка системы КОМПАС...

KOMPAS.chm::/DLG_FORMATCELL.htm

Интерфейс сетки таблицы.

Иерархия:

IDispatch

ITable

Описание:

Интерфейс можно получить с помощью следующих методов:

- IToleranceParam::Table у интерфейса параметров обозначения допуска формы,
- метода IUnknown::QueryInterface у следующих интерфейсов:
  - интерфейса таблицы на чертеже IDrawingTable,
  - допуска формы ITolerance и ITolerance3D.

ITable – свойства

Cell – Значение в ячейке таблицы

Интерфейс...

Тип данных: указатель на интерфейс ITableCell

Синтаксис Automation:

Cell = Object.Cell	Получить свойство(*)
Cell = Object.GetCell()	Получить свойство (**)

Синтаксис COM:

Object.get_Cell(&Cell)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс ячейки таблицы.
2. Свойство доступно только для чтения.
3. Для допуска формы возвращается интерфейс ITextLine.

CellById – Значение в ячейке таблицы, заданное по идентификатору

Интерфейс...

Тип данных: указатель на интерфейс ITableCell

Синтаксис Automation:

CellById = Object.CellById(CellID)	Получить свойство(*)
CellById = Object.GetCellById(CellID)	Получить свойство (**)

Синтаксис COM:

Object.get_CellById(&CellById)	CellID,	Получить свойство

Входные параметры:

CellID	- Идентификатор ячейки.

Примечание:

1. Свойство позволяет получить интерфейс ячейки таблицы.
2. Свойство доступно только для чтения.
3. Идентификаторы ячеек задаются, начиная с 1.

ColumnsCount – Количество столбцов

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnsCount = Object.ColumnsCount	Получить свойство(*)
ColumnsCount = Object.GetColumnsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(&ColumnsCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество столбцов таблицы.
2. Свойство доступно только для чтения.

Range – Интерфейс для групповых операций над колонками

Интерфейс...

Тип данных: указатель на интерфейс ITableRange

Синтаксис Automation:

Cell = Object.Range(BeginRow, BeginColumn EndRow, EndColumn)	Получить свойство(*)
Cell = Object.Range(BeginRow, BeginColumn EndRow, EndColumn)	Получить свойство (**)

Синтаксис COM:

Object.get_Range(BeginRow, BeginColumn, EndRow, EndColumn, &Range)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс для групповых операций над ячейками таблицы.
2. Свойство доступно только для чтения.

RowsCount – Количество строк

Интерфейс...

Тип данных: long

Синтаксис Automation:

RowsCount = Object.RowsCount	Получить свойство(*)
RowsCount = Object.GetRowsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_RowsCount(&RowsCount)	Получить свойство

Примечание:

1. Свойство позволяет получить количество строк таблицы.
2. Свойство доступно только для чтения.

ITable – методы

AddColumn – Добавить колонку

Интерфейс...

Синтаксис Automation:

BOOL AddColumn(long ColN,
BOOL right);

Синтаксис COM:

HRESULT AddColumn(long ColN,
BOOL right,
BOOL * PVal);

Входные параметры:

ColN	- номер столбца, относительно которого нужно вставить новый столбец,
right	- справа (TRUE) или слева (FALSE) от указанного.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет добавить в таблицу столбец справа или слева от указанного.

AddRow – Добавить строку

Интерфейс...

Синтаксис Automation:

BOOL AddRow(long RowN, BOOL down);

Синтаксис COM:

HRESULT AddRow(long RowN, BOOL down, BOOL * PVal);

Входные параметры:

RowN	- номер строки, относительно которой нужно вставить новую строку,
down	- снизу (TRUE) или сверху (FALSE) от указанной.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Clear – Очистить всю таблицу

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * PVal);

Входные параметры:

ColN	- номер удаляемого столбца.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет очистить таблицу.

DeleteColumn – Удалить колонку из таблицы

Интерфейс...

Синтаксис Automation:

BOOL DeleteColumn(long ColN);

Синтаксис COM:

HRESULT DeleteColumn(long ColN, BOOL * PVal);

Входные параметры:

ColN	- номер удаляемого столбца.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет удалить столбец из таблицы.

DeleteRow – Удалить строку из таблицы

Интерфейс...

Синтаксис Automation:

BOOL DeleteRow(long RowN);

Синтаксис COM:

HRESULT DeleteRow(long RowN,
BOOL * PVal);

Входные параметры:

ColN	- номер удаляемого столбца.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет удалить строку из таблицы.

Интерфейс ITableRange

Справка системы КОМПАС...

KOMPAS.chm::/582_Glava69_Obshchie_svedenija.htm

Интерфейс для групповых операций над ячейками таблицы.

Иерархия:

IDispatch

ITableRange

Описание:

Интерфейс можно получить с помощью метода интерфейса таблицы ITable::Range

ITableRange – свойства

Cells – Массив ячеек в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Значения свойства:

- Массив SafeArray типа VT_ARRAY | VT_DISPATCH.

Синтаксис Automation:

Cells = Object.Cells()	Получить свойство(*)
Cells = Object.GetCells()	Получить свойство (**)

Синтаксис COM:

Object.get_Cells(&Cells)	Получить свойство

Примечание:

1. Свойство позволяет получить массив ячеек SAFEARRAY. Это массив VT_DISPATCH, которые можно преобразовать в интерфейсы ITableCell. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

CellsBoundaries – Границы ячеек

Интерфейс...

Тип данных: указатель на интерфейс ICellBoundaries

Синтаксис Automation:

CellsBoundaries = Object.CellsBoundaries()	Получить свойство(*)
CellsBoundaries = Object.GetCellsBoundaries()	Получить свойство (**)

Синтаксис COM:

Object.get_CellsBoundaries(&CellsBoundaries)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс границы ячеек ICellBoundaries.
2. Свойство доступно только для чтения.

CellsFormat – Формат ячеек

Интерфейс...

Тип данных: указатель на интерфейс ICellFormat

Синтаксис Automation:

CellsFormat = Object.CellsFormat()	Получить свойство(*)
CellsFormat = Object.GetCellsFormat()	Получить свойство (**)

Синтаксис COM:

Object.get_CellsFormat(&CellsFormat)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс формата ячеек ICellFormat.
2. Свойство доступно только для чтения.

Texts – Получить тексты из ячеек в виде массива VT_ARRAY | VT_BSTR

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Texts = Object.Texts                    Получить свойство(* )
Object.Texts = Texts                    Установить свойство (* )
Texts = Object.GetTexts( )              Получить свойство (**)
Object.SetTexts( Texts )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Texts( &Texts )              Получить свойство
Object.put_Texts( Texts )               Установить свойство
```

ITableRange – методы

ClearCells – Очистить ячейки таблицы

Интерфейс...

Синтаксис Automation:

BOOL ClearCells();

Синтаксис COM:

HRESULT ClearCells(BOOL * PVal);

Возвращаемое значение:

```
TRUE        - в случае успеха,
FALSE       - в случае неудачи.
```

Примечание:

Позволяет очистить ячейки таблицы.

CombineCells – Объединить ячейки

Интерфейс...

Синтаксис Automation:

BOOL CombineCells();

Синтаксис COM:

HRESULT CombineCells(BOOL * Val);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет объединить ячейки таблицы.

DivideCells – Разбить ячейки

Интерфейс...

Синтаксис Automation:

BOOL DivideCells(long ColumnCount,
long RowCount,
BOOL PrepareCombine);

Синтаксис COM:

HRESULT DivideCells(long ColumnCount,
long RowCount,
BOOL PrepareCombine,
BOOL * PVal);

Входные параметры:

ColumnCount	- число столбцов,
RowCount	- число строк,
PrepareCombine	- объединить ячейки перед разбиением.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет разбить ячейки таблицы.

Интерфейс ICellFormat

Справка системы КОМПАС...

KOMPAS.chm::/DLG_FORMATCELL.htm

Интерфейс формата ячейки.

Иерархия:

IDispatch

ICellFormat

Описание:

Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейса ячейки таблицы ITableCell и с помощью свойства интерфейса операций над ячейками ITableRange::CellsFormat.

ICellFormat – свойства

Height – Высота строки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство(*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight( )	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height)	Получить свойство
Object.put_Height(Height)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать высоту строки.

HFormat – Признак горизонтального форматирования

Интерфейс...

Тип данных: ksTextHorizontalFormatEnum

Синтаксис Automation:

HFormat = Object.HFormat	Получить свойство(*)
Object.HFormat = HFormat	Установить свойство (*)
HFormat = Object.GetHFormat( )	Получить свойство (**)
Object.SetHFormat(HFormat)	Установить свойство (**)

Синтаксис COM:

Object.get_HFormat(&HFormat)	Получить свойство
Object.put_HFormat(HFormat)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак горизонтального форматирования.

LeftEdge – Отступ слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeftEdge = Object.LeftEdge	Получить свойство(*)
Object.LeftEdge = LeftEdge	Установить свойство (*)
LeftEdge = Object.GetLeftEdge( )	Получить свойство (**)
Object.SetLeftEdge(LeftEdge)	Установить свойство (**)

Синтаксис COM:

Object.get_LeftEdge(&LeftEdge)	Получить свойство
Object.put_LeftEdge(LeftEdge)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать величину отступа слева.

OneLine – Однострочный текст

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

OneLine = Object.OneLine	Получить свойство(*)
Object.OneLine = OneLine	Установить свойство (*)
OneLine = Object.GetOneLine( )	Получить свойство (**)
Object.SetOneLine(OneLine)	Установить свойство (**)

Синтаксис COM:

Object.get_OneLine(&OneLine)	Получить свойство
Object.put_OneLine(OneLine)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак однострочного текста.

ReadOnly – Запретить изменение текста в ячейке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство(*)
Object.ReadOnly = ReadOnly	Установить свойство (*)

ReadOnly = Object.GetReadOnly( )	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать признак запрета на изменение текста в ячейке.
2. Для того, чтобы сузить текст в ячейке таблицы необходимо:

- снять признак запрета на редактирование ячейки и изменить текст,
- вызвать метод IDrawingObject::Update,
- установить признак запрета на редактирование ячейки,
- вызвать метод IDrawingObject::Update.

RightEdge – Отступ справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

RightEdge = Object.RightEdge	Получить свойство(*)
Object.RightEdge = RightEdge	Установить свойство (*)
RightEdge = Object.GetRightEdge( )	Получить свойство (**)
Object.SetRightEdge(RightEdge)	Установить свойство (**)

Синтаксис COM:

Object.get_RightEdge(&RightEdge)	Получить свойство
Object.put_RightEdge(RightEdge)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать величину отступа справа.

SpaceAfter – Отступ снизу

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpaceAfter = Object.SpaceAfter	Получить свойство(*)
Object.SpaceAfter = SpaceAfter	Установить свойство (*)
SpaceAfter = Object.GetSpaceAfter( )	Получить свойство (**)
Object.SetSpaceAfter(SpaceAfter)	Установить свойство (**)

Синтаксис COM:

Object.get_SpaceAfter(&SpaceAfter)	Получить свойство
Object.put_SpaceAfter(SpaceAfter)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать величину отступа снизу.

SpaceBefore – Отступ сверху

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpaceBefore = Object.SpaceBefore	Получить свойство(*)
Object.SpaceBefore = SpaceBefore	Установить свойство (*)
SpaceBefore = Object.GetSpaceBefore( )	Получить свойство (**)
Object.SetSpaceBefore(SpaceBefore)	Установить свойство (**)

Синтаксис COM:

Object.get_SpaceBefore(&SpaceBefore)	Получить свойство
Object.put_SpaceBefore(SpaceBefore)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать величину отступа сверху.

TextStyle – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

TextStyle = Object.TextStyle	Получить свойство(*)
Object.TextStyle = TextStyle	Установить свойство (*)
TextStyle = Object.GetTextStyle( )	Получить свойство (**)

Object.SetTextStyle(TextStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_TextStyle(&TextStyle)	Получить свойство
Object.put_TextStyle(TextStyle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль текста в ячейке.

VFormat – Признак вертикального форматирования

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- изменение шага строк,
FALSE	- нет форматирования.

Синтаксис Automation:

VFormat = Object.VFormat	Получить свойство(*)
Object.VFormat = VFormat	Установить свойство (*)
VFormat = Object.GetVFormat( )	Получить свойство (**)
Object.SetVFormat(VFormat)	Установить свойство (**)

Синтаксис COM:

Object.get_VFormat(&VFormat)	Получить свойство
Object.put_VFormat(VFormat)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак вертикального форматирования.

Width – Ширина столбца

Интерфейс...

Тип данных: double

Синтаксис Automation:

Width = Object.Width	Получить свойство(*)
Object.Width = Width	Установить свойство (*)
Width = Object.GetWidth( )	Получить свойство (**)

Object.SetWidth(Width) Установить свойство (**)

Синтаксис COM:

Object.get_Width(&Width) Получить свойство
Object.put_Width(Width) Установить свойство

Примечание:

Свойство позволяет устанавливать и получать ширину столбца.

Интерфейс ICellBoundaries

Справка системы КОМПАС...

KOMPAS.chm::/582_Glava69_Obshchie_svedenija.htm

Интерфейс границ ячейки.

Иерархия:

IDispatch

ICellBoundaries

Описание:

Интерфейс можно получить с помощью метода IUnknown::QueryInterface у интерфейса ячейки таблицы ITableCell и с помощью свойства интерфейса операций над ячейками ITableRange::CellsBoundaries.

ICellBoundaries – свойства

LineStyle – Стиль линии границы

Интерфейс...

Тип данных: из перечисления ksCurveStyleEnum

Входные параметры:

index - граница, для которой можно установить видимость (из перечисления ksCellBoundariesEnum).

Синтаксис Automation:

LineStyle = Object.LineStyle(index) Получить свойство(*)
Object.LineStyle(index) = LineStyle Установить свойство (*)
LineStyle = Object.GetLineStyle(index) Получить свойство (**)
Object.SetLineStyle(index, LineStyle) Установить свойство (**)

Синтаксис COM:

Object.get_LineStyle(&LineStyle)	index,	Получить свойство
Object.put_LineStyle(LineStyle)	index,	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль линии границы ячейки.

LineVisible – Видимость границы

Интерфейс...

Тип данных: BOOL

Входные параметры:

index	- граница, для которой можно установить видимость (из перечисления ksCellBoundariesEnum).

Синтаксис Automation:

LineVisible = Object.LineVisible(index)	Получить свойство(*)
Object.LineVisible(index) = LineVisible	Установить свойство (*)
LineVisible = Object.GetLineVisible(index)	Получить свойство (**)
Object.SetLineVisible(index, LineVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_LineVisible(&LineVisible)	index,	Получить свойство
Object.put_LineVisible(LineVisible)	index,	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать видимость границы ячейки.

Интерфейс ITableCell

Справка системы КОМПАС...

KOMPAS.chm::/DLG_FORMATCELL.htm

Интерфейс для работы с таблицами.

Иерархия:

IDispatch

IKompasAPIObject

ITableCell

ICellFormat

ICellBoundaries

Описание:

1. Интерфейс можно получить с помощью метода интерфейса сетки таблицы ITable::Cell или свойства ITable::CellById.
2. Для ячейки таблицы 2D документа посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительные интерфейсы:

- ICellFormat - формат ячейки,
- ICellBoundaries - границы ячейки.

3. Для ячейки таблицы допуска формы интерфейсы ICellFormat и ICellBoundaries не поддерживаются.

ITableCell – свойства

CellID – Идентификатор ячейки

Интерфейс...

Тип данных: long

Синтаксис Automation:

CellID = Object.CellID	Получить свойство(*)
CellID = Object.GetCellID()	Получить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Column – Номер колонки

Интерфейс...

Тип данных: long

Синтаксис Automation:

Row = Object.Row	Получить свойство(*)
Row = Object.GetRow()	Получить свойство (**)

Синтаксис COM:

Object.get_Row(&Row) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство возвращает номер колонки в виртуальной сетке таблицы.

Row – Номер строки

Интерфейс...

Тип данных: long

Синтаксис Automation:

Row = Object.Row Получить свойство(*)
Row = Object.GetRow() Получить свойство (**)

Синтаксис COM:

Object.get_Row(&Row) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство возвращает номер строки в виртуальной сетке таблицы.

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Text = Object.Text Получить свойство(*)
Text = Object.GetText() Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Для ячейки таблицы 2D документа возвращается интерфейс IText.
3. Для допуска формы возвращается интерфейс ITextLine.

Текст

Интерфейс ITabulator

Интерфейс позиции табулятора.

Иерархия:

IKompasAPIObject

ITabulator

Описание:

Позволяет получить доступ к параметрам позиции табулятора и редактировать их.

Примечание:

Данный интерфейс может быть получен от интерфейса коллекции позиций табулятора ITabulators.

ITabulator – свойства

Align – Выравнивание

Интерфейс...

Тип данных: из перечисления ksAlignEnum

Синтаксис Automation:

```
Align = Object.Align;                  Получить свойство(* )
Object.Align = Align;                  Установить свойство (* )
Align = Object.GetAlign();             Получить свойство (**)
Object.SetAlign(Align);                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Align(&Align);              Получить свойство
Object.put_Align(Align);               Установить свойство
```

Filling – Заполнение

Интерфейс...

Тип данных: из перечисления ksTabulatorFillingEnum

Синтаксис Automation:

```
Filling = Object.Filling;              Получить свойство(* )
Object.Filling = Filling;              Установить свойство (* )
Filling = Object.GetFilling();         Получить свойство (**)
Object.SetFilling(Filling);            Установить свойство (**)
```

Синтаксис COM:

Object.get_Filling(&Filling);	Получить свойство
Object.put_Filling(Filling);	Установить свойство

Position – Позиция табуляции в мм * 100

Интерфейс...

Тип данных: long

Синтаксис Automation:

Position = Object.Position;	Получить свойство(*)
Object.Position = Position;	Установить свойство (*)
Position = Object.GetPosition();	Получить свойство (**)
Object.SetPosition(Position);	Установить свойство (**)

Синтаксис COM:

Object.get_Position(&Position);	Получить свойство
Object.put_Position(Position);	Установить свойство

Интерфейс IText

Справка системы КОМПАС...

`KOMPAS.chm::/306_29_11_5_Dopolnitelqnye_oboz.htm#Rfv45465`

Интерфейс текста для работы с аннотационными объектами.

Иерархия:

IKompasAPIObject

IText

Описание:

Интерфейс позволяет получить и отредактировать текст.

Текст имеет стиль. Стиль может быть системным из перечисления ksTextStyleEnum или пользовательским.

Состоит из строк, которые представлены интерфейсом ITextLine.

Строки текста состоят из компонент, которые представлены интерфейсом ITextItem.

Компонент текста - это часть строки, отличающаяся типом ksTextItemEnum и параметрами шрифта.

Примечание:

Данный интерфейс можно получить как дополнительный через интерфейс IDrawingText посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) или как самостоятельный объект у аннотационных объектов, имеющих тексты (линия-выноска, размер и т.п.).

IText – свойства

Count – Количество строк текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Count = iObject.Count;          Получить свойство(* )
Count = iObject.GetCount();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Count(&Count);     Получить свойство
```

Примечание:

Свойство позволяет получить количество строк текста.

Str – Текст в виде строки

Интерфейс...

Тип данных: BSTR(строка)

Синтаксис Automation:

```
Str = iObject.Str;              Получить свойство(* )
iObject.Str = Str;              Установить свойство (* )
Str = iObject.GetStr();         Получить свойство (**)
iObject.SetStr(Str);            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Str(&Str);         Получить свойство
iObject->put_Str(Str);          Установить свойство
```

Примечание:

Свойство позволяет установить и получить текст в виде массива символов.

Дроби представляются в формате $d числитель $ знаменатель $.

Выражение типа суммы в формате $ верхний индекс $ нижний индекс $.

Спецзнак в формате &N или @N.

При замене текста все строки очищаются и создаются новые с учетом символа перевода строки.

Style – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

Style = iObject.Style;	Получить свойство(*)
iObject.Style = Style;	Установить свойство (*)
Style = iObject.GetStyle();	Получить свойство (**)
iObject.SetStyle(Style);	Установить свойство (**)

Синтаксис COM:

iObject->get_Style(&Style);	Получить свойство
iObject->put_Style(Style);	Установить свойство

Примечание:

1. Свойство позволяет получить и установить стиль текста.
2. Стиль может быть системным из перечисления ksTextStyleEnum или пользовательским.

TextLine – Строка текста с заданным индексом

Интерфейс...

Тип данных: указатель на интерфейс строки текста ITextLine

Синтаксис Automation:

TextLine = iObject.TextLine(Index);	Получить свойство(*)
TextLine = iObject.GetTextLine(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_TextLine(Index, &TextLine);	Получить свойство

Входные параметры:

Index (long)	- индекс строки.

Примечание:

Свойство позволяет получить интерфейс строки текста ITextLine.

TextLines – Массив SAFEARRAY строк текста

Интерфейс...

Тип данных: VARIANT (VT_ARRAY|VT_DISPATCH)

Синтаксис Automation:

TextLines = iObject.TextLines;	Получить свойство(*)
TextLines = iObject.GetTextLines();	Получить свойство (**)

Синтаксис COM:

iObject->get_TextLines(&TextLines);	Получить свойство

Примечание:

Свойство позволяет получить массив SAFEARRAY строк текста. Это массив интерфейсов ITextLine. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

IText – методы

Add – Добавить строку в конец текста

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add([out, retval] ITextLine ** Value);

Возвращаемое значение:

	- указатель на интерфейс ITextLine строка текста.

Примечание:

Позволяет добавить строку в конец текста.

AddBefore – Добавить строку перед строкой с заданным индексом

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddBefore(long Index);

Синтаксис COM:

HRESULT AddBefore([in] long Index, [out, retval] ITextLine ** Value);

Входные параметры:

Index (long)	- индекс строки, перед которой нужно вставить новую строку.

Возвращаемое значение:

- указатель на интерфейс ITextLine строка текста.

Примечание:

Позволяет добавить строку перед строкой с заданным индексом.

AddTable – Добавить таблицу в текст (для текстового документа)

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddTable(long IndexAt,
long RowsCount,
long ColumnsCount,
double RowHeigh,
double ColumnsWidth,
ksTableTileLayoutEnum TitlePos);

Синтаксис COM:

HRESULT AddTable(long IndexAt,
long RowsCount,
long ColumnsCount,
double RowHeigh,
double ColumnsWidth,
ksTableTileLayoutEnum TitlePos,
ITextTable ** Result);

Возвращаемое значение:

Указатель на интерфейс таблицы ITextTable

Входные параметры:

RowsCount	- количество строк,
ColumnsCount	- количество столбцов,
RowHeigh	- высота строк,
ColumnsWidth	- ширина столбцов,
TitlePos	- расположение заголовка таблицы (из перечисления ksTableTileLayoutEnum).

Выходные параметры:

IndexAt	- индекс строки текста, с которой вставляется таблица.

Метод позволяет вставить таблицу в текстовый документ

AddTextLine – Добавить строку с внешними данными

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddTextLine(ksTextLineType type,
BSTR filename);

Синтаксис COM:

HRESULT AddTextLine(ksTextLineType type,
BSTR filename,
ITextLine ** Value);

Входные параметры:

type	- тип строки ksTextLineType,
filename	- имя файла - источник вставки.

Возвращаемое значение:

Указатель на интерфейс ITextLine строки текста.

Примечание:

1. В 2D объекте таблица IDrawingTable доступна возможность вставки вертикального текста, растрового объекта и фрагмента.
2. В текстовом документе есть возможность вставки таблицы в текст.
3. С помощью функции AddTextLine и AddTextLineBefore можно выполнить вставку следующих объектов через API системы КОМПАС:

   - для типов:

- ksTLText - простой текст,
- ksTLVerticalText - вертикальный текст.

  Параметр filename используется для передачи пути к файлу с расширениями txt, rtf и kdw. Из файла будет вставлена первая строчка текста. Если расширение не задано, то параметр filename воспринимается как обычная строка и она добавляется в текст.

- ksTLFragment - вставка фрагмента.

  Параметр filename используется для передачи пути к файлу фрагмента системы КОМПАС с расширением frw.

- ksTLRaster - вставка рисунка.

  Параметр filename используется для передачи пути к растровым файлам.

- ksTLTable - вставка таблицы.

  Параметр filename используется для передачи пути к файлу таблицы с расширением kdw. Не реализовано, вставка таблицы будет доступна в текстовом документе.

4. Получить и изменить параметры вставленного объекта можно с помощью свойства ITextLine::TextLineData.

AddTextLineBefore – Добавить строку с внешними данными перед строкой с заданным индексом

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddTextLineBefore(ksTextLineType type,
long Index,
BSTR filename);

Синтаксис COM:

HRESULT AddTextLineBefore(ksTextLineType type,
long Index,
BSTR filename,
ITextLine ** Value);

Входные параметры:

type	- тип строки ksTextLineType,
Index	- индекс строки,
filename	- имя файла - источник вставки.

Возвращаемое значение:

Указатель на интерфейс ITextLine строки текста.

Примечание:

1. В 2D объекте таблица IDrawingTable доступна возможность вставки вертикального текста, растрового объекта и фрагмента.
2. В текстовом документе есть возможность вставки таблицы в текст.
3. С помощью функции AddTextLine и AddTextLineBefore можно выполнить вставку следующих объектов через API системы КОМПАС:

   - для типов:

- ksTLText - простой текст,
- ksTLVerticalText - вертикальный текст.

  Параметр filename используется для передачи пути к файлу с расширениями txt, rtf и kdw. Из файла будет вставлена первая строчка текста. Если расширение не задано, то параметр filename воспринимается как обычная строка и она добавляется в текст.

- ksTLFragment - вставка фрагмента.

  Параметр filename используется для передачи пути к файлу фрагмента системы КОМПАС с расширением frw.

- ksTLRaster - вставка рисунка.

  Параметр filename используется для передачи пути к растровым файлам.

- ksTLTable - вставка таблицы.

  Параметр filename используется для передачи пути к файлу таблицы с расширением kdw. Не реализовано, вставка таблицы будет доступна в текстовом документе.

4. Получить и изменить параметры вставленного объекта можно с помощью свойства ITextLine::TextLineData.

Assign – Скопировать текст

Интерфейс...

Синтаксис Automation:

BOOL Assign(LPDISPATCH Other);

Синтаксис COM:

HRESULT Assign(IText * Other, BOOL * Result);

Входные параметры:

Other	- копируемый текст - указатель на интерфейс IText.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Clear – Очистить текст

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear([out, retval] VARIANT_BOOL* Value);

Возвращаемое значение:

TRUE	- текст очищен,
FALSE	- ошибка.

Примечание:

Позволяет очистить текст. После этого можно вводить новый.

Edit – Редактировать в окне

Интерфейс...

Синтаксис Automation:

BOOL Edit(OLE_HANDLE HWnd);

Синтаксис COM:

HRESULT Edit(OLE_HANDLE HWnd, BOOL * Result);

Возвращаемое значение:

TRUE - в случае изменения текста,
FALSE - в случае отмены.

Replace – Заменить подстроку текста

Интерфейс...

Синтаксис Automation:

BOOL Replace(BSTR SrcText, BSTR NewText,
BOOL Case,
BOOL WordOnly,
BOOL ReplaceAll);

Синтаксис COM:

HRESULT Replace(BSTR SrcText, BSTR NewText,
BOOL Case,
BOOL WordOnly,
BOOL ReplaceAll, BOOL * Result);

Входные параметры:

SrcText - искомая подстрока,
NewText - новая строка,
Case - TRUE - различать регистр,
- FALSE - поиск без учета регистра,
WordOnly - TRUE - только слово целиком,
ReplaceAll - TRUE - заменить все вхождения,
- FALSE - заменить только первое вхождение.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Интерфейс ITextFont

Справка системы КОМПАС...

kompas.chm::/525_65_1_Vybor_shrifta_i_ustano.htm

Интерфейс параметров шрифта.

Иерархия:

IKompasAPIObject
    ITextFont

Описание:

Данный интерфейс позволяет назначить параметры шрифта, которые будут использоваться при вводе надписей в документ.

Примечание:

1. Данный интерфейс можно получить от интерфейса параметров стиля текста ITextStyle, используя свойство ITextStyle::Font.
2. Интерфейс ITextFont является дополнительным для интерфейса ITextItem.

ITextFont – свойства

Bold – Жирный шрифт

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Bold = iObject.Bold                 Получить свойство(* )
iObject.Bold = Bold                 Установить свойство (* )
Bold = iObject.GetBold()            Получить свойство (**)
iObject.SetBold (Bold)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Bold (&Bold)           Получить свойство
iObject->put_Bold (Bold)            Установить свойство
```

Значение свойства:

```
TRUE        - утолщенное начертание символов,
FALSE       - обычное начертание символов.
```

Примечание:

Свойство позволяет получить\изменить утолщенное начертание символов.

Color – Цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Color = iObject.Color               Получить свойство(* )
iObject.Color = Color               Установить свойство (* )
Color = iObject.GetColor()          Получить свойство (**)
iObject.SetColor (Color)            Установить свойство (**)
```

Синтаксис COM:

iObject->get_Color (&Color)	Получить свойство
iObject->put_Color (Color)	Установить свойство

Примечание:

Свойство позволяет получить\изменить цвет символов.

FontName – Имя шрифта

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

FontName = iObject.FontName	Получить свойство(*)
iObject.FontName = FontName	Установить свойство (*)
FontName = iObject.GetFontName()	Получить свойство (**)
iObject.SetFontName (FontName)	Установить свойство (**)

Синтаксис COM:

iObject->get_FontName (&FontName)	Получить свойство
iObject->put_FontName (FontName)	Установить свойство

Примечание:

Свойство позволяет получить/изменить название шрифта из доступных шрифтов.

Height – Высота текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = iObject.Height	Получить свойство(*)
iObject.Height = Height	Установить свойство (*)
Height = iObject.GetHeight()	Получить свойство (**)
iObject.SetHeight (Height)	Установить свойство (**)

Синтаксис COM:

iObject->get_Height (&Height)	Получить свойство
iObject->put_Height (Height)	Установить свойство

Примечание:

1. Свойство позволяет получить/изменить высоту символов.
2. Значение высоты символов задается в миллиметрах.

Italic – Курсив

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Italic = iObject.Italic	Получить свойство(*)
iObject.Italic = Italic	Установить свойство (*)
Italic = iObject.GetItalic()	Получить свойство (**)
iObject.SetItalic (Italic)	Установить свойство (**)

Синтаксис COM:

iObject->get_Italic (&Italic)	Получить свойство
iObject->put_Italic (Italic)	Установить свойство

Значение свойства:

TRUE	- курсивное начертание символов,
FALSE	- обычное начертание символов.

Примечание:

Свойство позволяет получить\изменить курсивное (наклонное) начертание символов.

TextLineStep – Шаг строк для настроек

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TextLineStep = Object.TextLineStep	Получить свойство(*)
Object.TextLineStep = TextLineStep	Установить свойство (*)
TextLineStep = Object.GetTextLineStep()	Получить свойство (**)
Object.SetTextLineStep (TextLineStep)	Установить свойство (**)

Синтаксис COM:

Object.get_TextLineStep (&TextLineStep)	Получить свойство
Object.put_TextLineStep (TextLineStep)	Установить свойство

Underline – Подчеркивание

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Underline = iObject.Underline	Получить свойство(*)
iObject.Underline = Underline	Установить свойство (*)
Underline = iObject.GetUnderline()	Получить свойство (**)
iObject.SetUnderline (Underline)	Установить свойство (**)

Синтаксис COM:

iObject->get_Underline (&Underline)	Получить свойство
iObject->put_Underline (Underline)	Установить свойство

Значение свойства:

TRUE	- подчеркивание символов,
FALSE	- обычное начертание символов.

Примечание:

Свойство позволяет получить\изменить курсивное (наклонное) подчеркивание текста.

WidthFactor – Коэффициент сужения

Интерфейс...

Тип данных: double

Синтаксис Automation:

WidthFactor = iObject.WidthFactor	Получить свойство(*)
iObject.WidthFactor = WidthFactor	Установить свойство (*)
WidthFactor = iObject.GetWidthFactor()	Получить свойство (**)
iObject.SetWidthFactor (WidthFactor)	Установить свойство (**)

Синтаксис COM:

iObject->get_WidthFactor (&WidthFactor)	Получить свойство
iObject->put_WidthFactor (WidthFactor)	Установить свойство

Примечание:

Свойство позволяет получить\изменить величину коэффициента сужения для символов.

Интерфейс ITextItem

Справка системы КОМПАС...

KOMPAS.chm::/CM_TEXT.htm#add_text_label

Интерфейс компоненты строки текста.

Иерархия:

IKompasAPIObject

ITextItem

Описание:

Интерфейс позволяет получить и отредактировать компонент строки текста. Компонент строки - это часть строки, отличающаяся типом ksTextItemEnum и параметрами шрифта.

Интерфейс имеет дополнительные интерфейсы ITextFont и IHypertextReferenceParam.

Примечание:

Данный интерфейс можно получить через интерфейс ITextLine, используя свойство ITextLine::TextItem.

ITextItem – свойства

ItemType – Стиль текста

Интерфейс...

Тип данных: ksTextItemEnum

Синтаксис Automation:

ItemType = iObject.ItemType	Получить свойство(*)
iObject.ItemType = ItemType	Установить свойство (*)
ItemType = iObject.GetItemType()	Получить свойство (**)
iObject.SetItemType(ItemType)	Установить свойство (**)

Синтаксис COM:

iObject->get_ItemType(&ItemType)	Получить свойство
iObject->put_ItemType(ItemType)	Установить свойство

Примечание:

Свойство позволяет установить и получить тип компоненты текста. Используя соответствующий тип компоненты, можно формировать дроби, выражения типа суммы, вставлять спецзнаки, символы из произвольных шрифтов, вводить текст с разными параметрами шрифта (цвет, высота, сужение.и т.п.).

NewLine – Признак начала строки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NewLine = iObject.NewLine	Получить свойство(*)
iObject.NewLine = NewLine	Установить свойство (*)
NewLine = iObject.GetNewLine()	Получить свойство (**)
iObject.SetNewLine(NewLine)	Установить свойство (**)

Синтаксис COM:

iObject->get_NewLine(&NewLine)	Получить свойство
iObject->put_NewLine(NewLine)	Установить свойство

Примечание:

Свойство позволяет установить и получить признак начала строки.

Number – Номер спецсимвола, символа из произвольного шрифта

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство(*)
iObject.Number = Number	Установить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)
iObject.SetNumber(Number)	Установить свойство (**)

Синтаксис COM:

iObject->get_Number(&Number)	Получить свойство
iObject->put_Number(Number)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить номер спецсимвола, символа из произвольного шрифта.
2. Это свойство используется для компонент типа ksTItSpecialSymbol и ksTItFontSymbol из перечисления ksTextItemEnum.

SizeFactor – Размерный коэффициент текста для дроби, отклонений, выражения типа суммы

Интерфейс...

Тип данных: из перечисления ksTextSizeEnum

Синтаксис Automation:

```
SizeFactor = iObject.SizeFactor                Получить свойство(* )
iObject.SizeFactor = SizeFactor                Установить свойство (* )
SizeFactor = iObject.GetSizeFactor()           Получить свойство (**)
iObject.SetSizeFactor(SizeFactor)              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_SizeFactor(&SizeFactor)           Получить свойство
iObject->put_SizeFactor(SizeFactor)            Установить свойство
```

Примечание:

1. Свойство позволяет установить и получить размерный коэффициент текста для дроби, отклонений, выражения типа суммы.
2. Это свойство используется для компонент типа ksTItNumerator, ksTItDenominator, ksTItSBase из перечисления ksTextItemEnum.
3. Размерный коэффициент текста используется для задания размера текста дроби, отклонений, для выражения типа суммы

   Для дроби и отклонений используются:

```
ksTextNormal    - полной высоты,
ksTextMiddle    - в 1.5 раза меньше,
ksTextSmall     - в 2 раза меньше.
```

   Для выражения типа суммы используются:

```
ksTextNormal    - нормальной высоты,
ksTextBig       - в 1.5 раза больше.
```

Str – Текстовое значение компоненты текста

Интерфейс...

Тип данных: BSTR(строка)

Синтаксис Automation:

```
Str = iObject.Str;                             Получить свойство(* )
iObject.Str = Str;                             Установить свойство (* )
Str = iObject.GetStr();                        Получить свойство (**)
iObject.SetStr(Str);                           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Str(&Str);                        Получить свойство
```

iObject->put_Str(Str);	Установить свойство

Примечание:

1. Свойство позволяет установить и получить текст в виде массива символов.
2. Это свойство не используется для компонент типа ksTItSpecialSymbol и ksTItFontSymbol из перечисления ksTextItemEnum.

SymbolFontName – Имя шрифта для символа

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

SymbolFontName = iObject.SymbolFontName	Получить свойство(*)
iObject.SymbolFontName = SymbolFontName	Установить свойство (*)
iObject.GetSymbolFontName()	Получить свойство (**)
iObject.SetSymbolFontName(SymbolFontName)	Установить свойство (**)

Синтаксис COM:

iObject->get_SymbolFontName(&SymbolFontName)	Получить свойство
iObject->put_SymbolFontName(SymbolFontName)	Установить свойство

Примечание:

1. Свойство позволяет установить и получить имя шрифта для символа.
2. Это свойство используется для компонент типа ksTItFontSymbol из перечисления ksTextItemEnum.

ITextItem – методы

Delete – Удалить компоненту строки

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete([out, retval] VARIANT_BOOL* Value);

Возвращаемое значение:

TRUE	- компонента строки текста удалена,
FALSE	- ошибка.

Примечание:

Позволяет удалить компоненту строки

Update – Обновить данные компонента

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update([out, retval] VARIANT_BOOL* Value);

Возвращаемое значение:

TRUE	- обновление данных компонента прошло успешно,
FALSE	- ошибка.

Примечание:

Обновление данных компонента необходимо, так как многие параметры компонента взаимосвязаны и зависят от типа ksTextItemEnum. Поэтому менять компоненту в строке нужно целиком.

Интерфейс ITextLine

Справка системы КОМПАС...

`KOMPAS.chm::/CM_TEXT.htm#add_text_label`

Интерфейс строки текста.

Иерархия:

IKompasAPIObject

ITextLine

Описание:

Интерфейс позволяет получить и отредактировать строку текста.

Строка может иметь стиль, отличающийся от стиля текста. Стиль может быть системным из перечисления ksTextStyleEnum или пользовательским.

Строки текста состоят из компонентов, которые представлены интерфейсом ITextItem.

Компонент - это часть строки, отличающаяся типом ksTextItemEnum и параметрами шрифта.

Примечание:

Данный интерфейс можно получить через интерфейс IText, используя свойство IText::TextLine.

ITextLine – свойства

Align – Выравнивание

Интерфейс...

Тип данных: типы выравнивания ksAlignEnum

Синтаксис Automation:

```
Align = iObject.Align;          Получить свойство(* )
iObject.Align = Align;          Установить свойство (* )
Align = iObject.GetAlign ();    Получить свойство (**)
iObject.SetAlign (Align);       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Align (&Align);    Получить свойство
iObject->put_Align (Align);     Установить свойство
```

Примечание:

Свойство позволяет установить и получить выравнивание строк.

Строка представляет собой абзац текста и может размещаться на нескольких строках, если включено выравнивание текста по горизонтали и вертикали.

BlockIndex – Индекс листа в документе

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
BlockIndex = Object.BlockIndex          Получить свойство(* )
BlockIndex = Object.GetBlockIndex()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_BlockIndex( &BlockIndex )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Count – Количество компонентов строки текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = iObject.Count;	Получить свойство(*)
Count = iObject.GetCount ();	Получить свойство (**)

Синтаксис COM:

iObject->get_Count (&Count);	Получить свойство

Примечание:

Свойство позволяет получить количество компонентов строки текста.

IndentedLine – Смещение красной строки

Интерфейс...

Тип данных: double

Синтаксис Automation:

IndentedLine = iObject.IndentedLine;	Получить свойство(*)
iObject.IndentedLine = IndentedLine;	Установить свойство (*)
IndentedLine = iObject.GetIndentedLine ();	Получить свойство (**)
iObject.SetIndentedLine (IndentedLine);	Установить свойство (**)

Синтаксис COM:

iObject->get_IndentedLine (&IndentedLine);	Получить свойство
iObject->put_IndentedLine (IndentedLine);	Установить свойство

Примечание:

Свойство позволяет установить и получить смещение красной строки.

LeftEdge – Отступ текста слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeftEdge = iObject.LeftEdge;	Получить свойство(*)
iObject.LeftEdge = LeftEdge;	Установить свойство (*)
LeftEdge = iObject.GetLeftEdge ();	Получить свойство (**)
iObject.SetLeftEdge (LeftEdge);	Установить свойство (**)

Синтаксис COM:

iObject->get_LeftEdge (&LeftEdge);	Получить свойство
iObject->put_LeftEdge (LeftEdge);	Установить свойство

Примечание:

Свойство позволяет установить и получить отступ текста слева.

Level – Уровень вложенности нумерации

Интерфейс...

Тип данных: long

Синтаксис Automation:

Level = iObject.Level;	Получить свойство(*)
iObject.Level = Level;	Установить свойство (*)
Level = iObject.GetLevel ();	Получить свойство (**)
iObject.SetLevel (Level);	Установить свойство (**)

Синтаксис COM:

iObject->get_Level (&Level);	Получить свойство
iObject->put_Level (Level);	Установить свойство

римечание:

Свойство позволяет установить и получить уровень вложенности нумерации.

NewPage – Признак начала абзаца с новой страницы (Ctrl+Enter)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NewPage = Object.NewPage	Получить свойство(*)
Object.NewPage = NewPage	Установить свойство (*)
NewPage = Object.GetNewPage()	Получить свойство (**)
Object.SetNewPage(NewPage)	Установить свойство (**)

Синтаксис COM:

Object.get_NewPage(&NewPage)	Получить свойство
Object.put_NewPage(NewPage)	Установить свойство

NewSection – Признак начала раздела

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NewSection = Object.NewSection	Получить свойство(*)
Object.NewSection = NewSection	Установить свойство (*)
NewSection = Object.GetNewSection()	Получить свойство (**)
Object.SetNewSection(NewSection)	Установить свойство (**)

Синтаксис COM:

Object.get_NewSection(&NewSection)	Получить свойство
Object.put_NewSection(NewSection)	Установить свойство

Numbering – Тип нумерации абзаца

Интерфейс...

Тип данных: из перечисления ksTextNumberingEnum

Синтаксис Automation:

Numbering = iObject.Numbering;	Получить свойство(*)
iObject.Numbering = Numbering;	Установить свойство (*)
Numbering = iObject.GetNumbering ();	Получить свойство (**)
iObject.SetNumbering (Numbering);	Установить свойство (**)

Синтаксис COM:

iObject->get_Numbering (&Numbering);	Получить свойство
iObject->put_Numbering (Numbering);	Установить свойство

Примечание:

Свойство позволяет установить и получить тип нумерации абзаца.

RightEdge – Отступ текста справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

RightEdge = iObject.RightEdge;	Получить свойство(*)
iObject.RightEdge = RightEdge;	Установить свойство (*)
RightEdge = iObject.GetRightEdge ();	Получить свойство (**)
iObject.SetRightEdge (RightEdge);	Установить свойство (**)

Синтаксис COM:

iObject->get_RightEdge (&RightEdge);	Получить свойство
iObject->put_RightEdge (RightEdge);	Установить свойство

Примечание:

Свойство позволяет установить и получить отступ текста справа.

SectionBlockIndex – Индекс листа в разделе

Интерфейс...

Тип данных: long

Синтаксис Automation:

SectionBlockIndex = Object.SectionBlockIndex	Получить свойство(*)
SectionBlockIndex = Object.GetSectionBlockIndex()	Получить свойство (**)

Синтаксис COM:

Object.get_SectionBlockIndex(&SectionBlockIndex)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Step – Шаг строк

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = iObject.Step;	Получить свойство(*)
iObject.Step = Step;	Установить свойство (*)
Step = iObject.GetStep ();	Получить свойство (**)
iObject.SetStep (Step);	Установить свойство (**)

Синтаксис COM:

iObject->get_Step (&Step);	Получить свойство
iObject->put_Step (Step);	Установить свойство

Примечание:

Свойство позволяет установить и получить шаг строк.

Строка представляет собой абзац текста и может размещаться на нескольких строках, если включено выравнивание текста по горизонтали и вертикали.

StepAfterParagraph – Дополнительный шаг после абзаца

Интерфейс...

Тип данных: double

Синтаксис Automation:

StepAfterParagraph = iObject.StepAfterParagraph;	Получить свойство(*)
iObject.StepAfterParagraph = StepAfterParagraph;	Установить свойство (*)
StepAfterParagraph = iObject.GetStepAfterParagraph ();	Получить свойство (**)
iObject.SetStepAfterParagraph (StepAfterParagraph);	Установить свойство (**)

Синтаксис COM:

iObject->get_StepAfterParagraph (&StepAfterParagraph);	Получить свойство
iObject->put_StepAfterParagraph (StepAfterParagraph);	Установить свойство

Примечание:

Свойство позволяет установить и получить дополнительный шаг после абзаца.

StepBeforeParagraph – Дополнительный шаг перед абзацем

Интерфейс...

Тип данных: double

Синтаксис Automation:

StepBeforeParagraph = iObject.StepBeforeParagraph;	Получить свойство(*)
iObject.StepBeforeParagraph = StepBeforeParagraph;	Установить свойство (*)
iObject.GetStepBeforeParagraph ();	Получить свойство (**)
iObject.SetStepBeforeParagraph (StepBeforeParagraph);	Установить свойство (**)

Синтаксис COM:

iObject->get_StepBeforeParagraph (&StepBeforeParagraph);	Получить свойство
iObject->put_StepBeforeParagraph (StepBeforeParagraph);	Установить свойство

Примечание:

Свойство позволяет установить и получить дополнительный шаг перед абзацем.

Str – Текст в виде строки

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Str = iObject.Str;	Получить свойство(*)
iObject.Str = Str;	Установить свойство (*)
Str = iObject.GetStr ();	Получить свойство (**)
iObject.SetStr (Str);	Установить свойство (**)

Синтаксис COM:

iObject->get_Str (&Str);	Получить свойство
iObject->put_Str (Str);	Установить свойство

Примечание:

Свойство позволяет установить и получить текст в виде массива символов.

Дроби представляются в формате $d числитель $ знаменатель $.

Выражение типа суммы в формате $ верхний индекс $ нижний индекс $.

Спецзнак в формате &N или @N.

При замене строки очищаются и создаются новые компоненты.

Style – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Style = iObject.Style;                  Получить свойство(* )
iObject.Style = Style;                  Установить свойство (* )
Style = iObject.GetStyle ();            Получить свойство (**)
iObject.SetStyle (Style);               Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Style (&Style);            Получить свойство
iObject->put_Style (Style);             Установить свойство
```

Примечание:

1. Свойство позволяет получить и установить стиль текста.
2. Стиль может быть системным из перечисления ksTextStyleEnum или пользовательским.

Tabulators – Коллекция позиции табуляторов

Интерфейс...

Тип данных: Указатель на интерфейс ITabulators

Синтаксис Automation:

```
Tabulators = Object.Tabulators;         Получить свойство(* )
Tabulators = Object.GetTabulators();    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Tabulators(                  Получить свойство
&Tabulators );
```

Примечание:

Свойство доступно только для чтения.

TextItem – Компонент строки текста с заданным индексом

Интерфейс...

Тип данных: указатель на интерфейс компонента строки текста ITextItem

Синтаксис Automation:

TextItem = iObject.TextItem (Index);	Получить свойство(*)
TextItem = iObject.GetTextItem (Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_TextItem &TextItem);	(Index,	Получить свойство

Входные параметры:

Index (long)	- индекс компонента строки.

Примечание:

Свойство позволяет получить интерфейс компонента строки текста ITextItem.

TextItems – Массив SAFEARRAY компонентов строки текста

Интерфейс...

Тип данных: VARIANT (VT_ARRAY|VT_DISPATCH)

Синтаксис Automation:

TextItems = iObject.TextItems;	Получить свойство(*)
TextItems = iObject.GetTextItems ();	Получить свойство (**)

Синтаксис COM:

iObject->get_TextItems (&TextItems);	Получить свойство

Примечание:

Свойство позволяет получить массив SAFEARRAY компонентов строки текста. Это массив интерфейсов ITextItem. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

TextLineData – Интерфейс дополнительных данных строки

Интерфейс...

Тип данных: IKompasAPIObject

Синтаксис Automation:

TextLineData = Object.TextLineData	Получить свойство(*)
TextLineData = Object.GetTextLineData ()	Получить свойство (**)

Синтаксис COM:

Object.get_TextLineData (&TextLineData)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. В зависимости от типа строки TextLineType возвращаются следующие значения свойства:

- ksTLText - NULL,
- ksTLVerticalText - указатель на интерфейс IText,
- ksTLFragment - указатель на интерфейс IInsertionParameters,
- ksTLRaster - указатель на интерфейс IInsertionParameters,
- ksTLTable - указатель на интерфейс ITextTable.

TextLineType – Тип строки

Интерфейс...

Тип данных: ksTextLineType

Синтаксис Automation:

TextLineType = Object.TextLineType	Получить свойство(*)
TextLineType = Object.GetTextLineType ()	Получить свойство (**)

Синтаксис COM:

Object.get_TextLineType (&TextLineType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ITextLine – методы

Add – Добавить компонент строки в конец строки

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add ();

Синтаксис COM:

HRESULT Add ([out, retval] ITextItem ** Value);

Возвращаемое значение:

TRUE	- указатель на интерфейс ITextItem компонента строки текста.

Примечание:

Позволяет добавить компонент строки в конец строки.

AddBefore – Добавить компонент строки перед компонентом с заданным индексом

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddBefore (long Index);

Синтаксис COM:

HRESULT AddBefore ([in] long Index, [out, retval] ITextItem ** Value);

Выходные параметры:

Index (long)	- индекс компонента строки, перед которым нужно вставить новый компонент.

Возвращаемое значение:

	- указатель на интерфейс ITextItem строка текста.

Примечание:

Позволяет добавить компонент строки перед компонентом с заданным индексом.

Assign – Скопировать текст

Интерфейс...

Синтаксис Automation:

BOOL Assign(LPDISPATCH Other);

Синтаксис COM:

HRESULT Assign(ITextLine * Other, BOOL * Result);

Входной параметр:

Other	- копируемый текст - указатель на интерфейс ITextLine.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Clear – Очистить строку текста

Интерфейс...

Синтаксис Automation:

BOOL Clear ();

Синтаксис COM:

HRESULT Clear ([out, retval] VARIANT_BOOL* Value);

Возвращаемое значение:

TRUE	- строка очищена,
FALSE	- ошибка.

Примечание:

Позволяет очистить строку текста. После этого можно вводить новую строку.

Delete – Удалить строку текста

Интерфейс...

Синтаксис Automation:

BOOL Delete ();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL* Value);

Возвращаемое значение:

TRUE	- строка текста удалена,
FALSE	- ошибка.

Примечание:

Позволяет удалить строку текста.

Edit – Редактировать в окне

Интерфейс...

Синтаксис Automation:

BOOL Edit(OLE_HANDLE HWnd);

Синтаксис COM:

HRESULT Edit(OLE_HANDLE HWnd, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае изменения текста,
FALSE	- в случае отмены.

InsertHypertextReference – Добавить ссылку в текст

Интерфейс...

Синтаксис Automation:

LPDISPATCH InsertHypertextReference(long TextItemIndex,
IKompasAPIObject * Object,
ksHypertextTypeEnum Type,
BOOL Brackets,
long TextLineIndex,
long Precision,
double PropertyId);

Синтаксис COM:

HRESULT InsertHypertextReference(long TextItemIndex,
IKompasAPIObject * Object,
ksHypertextTypeEnum Type,
BOOL Brackets,
long TextLineIndex,
long Precision,
double PropertyId,
ITextItem ** Result);

Входные параметры:

TextItemIndex	- индекс компонента текста,
Object	- объект, на который сделана ссылка,
Type	- тип ссылки,
Brackets	- признак скобок,
TextLineIndex	- индекс строки в тексте,
Precision	- точность вывода числовых значений,
PropertyId	- идентификатор свойства.

Возвращаемое значение:

- указатель на интерфейс компонента текста, содержащего ссылку.

Интерфейс ITextStyle

Справка системы КОМПАС...

kompas.chm::/527_65_5_Formatirovanie_teksta.htm

Интерфейс параметров стиля текста.

Иерархия:

IKompasAPIObject

ITextStyle

Примечание:

Данный интерфейс можно получить от интерфейса настроек спецификации ISpecificationTuning, используя:

- ISpecificationTuning::SectionTextStyleFirst;
- ISpecificationTuning::SectionTextStyleNext;
- ISpecificationTuning::ObjectTextStyle;
- ISpecificationTuning::PerformanceBlockTextStyleFirst;
- ISpecificationTuning::PerformanceBlockTextStyleNext;
- ISpecificationTuning::AdditionalBlockTextStyleFirst;
- ISpecificationTuning::AdditionalBlockTextStyleNext;
- ISpecificationTuning::NestingBlockTextStyleFirst;
- ISpecificationTuning::NestingBlockTextStyleNext.

ITextStyle – свойства

Align – Выравнивание

Интерфейс...

Тип данных: из перечисления ksAlignEnum

Синтаксис Automation:

Align = iObject.Align	Получить свойство(*)
iObject.Align = Align	Установить свойство (*)
Align = iObject.GetAlign()	Получить свойство (**)
iObject.SetAlign (Align)	Установить свойство (**)

Синтаксис COM:

iObject->get_Align (&Align)	Получить свойство
iObject->put_Align (Align)	Установить свойство

Примечание:

1. Свойство позволяет получить/изменить выравнивание абзацев (по левой границе, по правой границе, центрирование, выравнивание по двум границам).
2. Свойство используется для стиля, полученного через методы интерфейса ISpecificationTuning.

EnableLine1 – Разрешена ли первая комбинация для использования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableLine1 = Object.EnableLine1	Получить свойство(*)
Object.EnableLine1 = EnableLine1	Установить свойство (*)
EnableLine1 = Object.GetEnableLine1()	Получить свойство (**)
Object.SetEnableLine1 (EnableLine1)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableLine1 (&EnableLine1)	Получить свойство
Object.put_EnableLine1(EnableLine1)	Установить свойство

EnableLine2 – Разрешена ли вторая комбинация для использования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableLine2 = Object.EnableLine2	Получить свойство(*)
Object.EnableLine2 = EnableLine2	Установить свойство (*)
EnableLine2 = Object.GetEnableLine2()	Получить свойство (**)
Object.SetEnableLine2 (EnableLine2)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableLine2 (&EnableLine2)	Получить свойство
Object.put_EnableLine2(EnableLine2)	Установить свойство

EnableLine3 – Разрешена ли третья комбинация для использования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableLine3 = Object.EnableLine3	Получить свойство(*)
Object.EnableLine3 = EnableLine3	Установить свойство (*)
EnableLine3 = Object.GetEnableLine3()	Получить свойство (**)
Object.SetEnableLine3 (EnableLine3)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableLine3 (&EnableLine3)	Получить свойство
Object.put_EnableLine3(EnableLine3)	Установить свойство

Extended – Расширенный стиль текста

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Extended = Object.Extended	Получить свойство(*)
Object.Extended = Extended	Установить свойство (*)
Extended = Object.GetExtended()	Получить свойство (**)
Object.SetExtended (Extended)	Установить свойство (**)

Синтаксис COM:

Object.get_Extended (&Extended)	Получить свойство
Object.put_Extended(Extended)	Установить свойство

Font – Параметры шрифта

Интерфейс...

Тип данных: указатель на интерфейс ITextFont параметров шрифта

Синтаксис Automation:

Font = iObject.Font	Получить свойство(*)
Font = iObject.GetFont()	Получить свойство (**)

Синтаксис COM:

iObject->get_Font (&Font)	Получить свойство

Примечание:

1. Свойство позволяет параметры шрифта.
2. Свойство доступно только для чтения.
3. Свойство используется для стиля, полученного через методы интерфейса ISpecificationTuning.

Height1 – Высота первой строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height1 = Object.Height1	Получить свойство(*)
Object.Height1 = Height1	Установить свойство (*)
Height1 = Object.GetHeight1()	Получить свойство (**)
Object.SetHeight1 (Height1)	Установить свойство (**)

Синтаксис COM:

Object.get_Height1 (&Height1)	Получить свойство
Object.put_Height1(Height1)	Установить свойство

Height2 – Высота второй строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height2 = Object.Height2	Получить свойство(*)
Object.Height2 = Height2	Установить свойство (*)
Height2 = Object.GetHeight2()	Получить свойство (**)
Object.SetHeight2 (Height2)	Установить свойство (**)

Синтаксис COM:

Object.get_Height2 (&Height2)	Получить свойство
Object.put_Height2(Height2)	Установить свойство

Height3 – Высота третьей строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height3 = Object.Height3	Получить свойство(*)
Object.Height3 = Height3	Установить свойство (*)
Height3 = Object.GetHeight3()	Получить свойство (**)
Object.SetHeight3 (Height3)	Установить свойство (**)

Синтаксис COM:

Object.get_Height3 (&Height3)	Получить свойство
Object.put_Height3(Height3)	Установить свойство

IndentedLine – Смещение красной строки

Интерфейс...

Тип данных: double

Синтаксис Automation:

IndentedLine = Object.IndentedLine	Получить свойство(*)
Object.IndentedLine = IndentedLine	Установить свойство (*)
IndentedLine = Object.GetIndentedLine()	Получить свойство (**)
Object.SetIndentedLine (IndentedLine)	Установить свойство (**)

Синтаксис COM:

Object.get_IndentedLine (&IndentedLine)	Получить свойство
Object.put_IndentedLine(IndentedLine)	Установить свойство

LeftEdge – Отступ текста слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

LeftEdge = Object.LeftEdge	Получить свойство(*)
Object.LeftEdge = LeftEdge	Установить свойство (*)
LeftEdge = Object.GetLeftEdge()	Получить свойство (**)
Object.SetLeftEdge (LeftEdge)	Установить свойство (**)

Синтаксис COM:

Object.get_LeftEdge (&LeftEdge)	Получить свойство

Object.put_LeftEdge(LeftEdge)	Установить свойство

LinesCount – Количество строк

Интерфейс...

Тип данных: long

Синтаксис Automation:

LinesCount = Object.LinesCount	Получить свойство(*)
Object.LinesCount = LinesCount	Установить свойство (*)
LinesCount = Object.GetLinesCount()	Получить свойство (**)
Object.SetLinesCount (LinesCount)	Установить свойство (**)

Синтаксис COM:

Object.get_LinesCount (&LinesCount)	Получить свойство
Object.put_LinesCount(LinesCount)	Установить свойство

Name – Имя стиля

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство(*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName (Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name (&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Number – Номер стиля

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = Object.Number	Получить свойство(*)
Object.Number = Number	Установить свойство (*)
Number = Object.GetNumber()	Получить свойство (**)
Object.SetNumber (Number)	Установить свойство (**)

Синтаксис COM:

Object.get_Number (&Number)	Получить свойство
Object.put_Number(Number)	Установить свойство

RightEdge – Отступ текста справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

RightEdge = Object.RightEdge	Получить свойство(*)
Object.RightEdge = RightEdge	Установить свойство (*)
RightEdge = Object.GetRightEdge()	Получить свойство (**)
Object.SetRightEdge (RightEdge)	Установить свойство (**)

Синтаксис COM:

Object.get_RightEdge (&RightEdge)	Получить свойство
Object.put_RightEdge(RightEdge)	Установить свойство

Step – Шаг строк

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = iObject.Step	Получить свойство(*)
iObject.Step = Step	Установить свойство (*)
Step = iObject.GetStep()	Получить свойство (**)
iObject.SetStep (Step)	Установить свойство (**)

Синтаксис COM:

iObject->get_Step (&Step)	Получить свойство
iObject->put_Step(Step)	Установить свойство

Примечание:

1. Свойство позволяет получить/изменить шаг строк (расстояние между строками текста).
2. Значение шага задается в миллиметрах.
3. Свойство используется для стиля, полученного через методы интерфейса ISpecificationTuning.

Step1 – Шаг первой строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step1 = Object.Step1                    Получить свойство(* )
Object.Step1 = Step1                    Установить свойство (* )
Step1 = Object.GetStep1()               Получить свойство (**)
Object.SetStep1 (Step1)                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step1 (&Step1)               Получить свойство
Object.put_Step1(Step1)                 Установить свойство
```

Step2 – Шаг второй строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step2 = Object.Step2                    Получить свойство(* )
Object.Step2 = Step2                    Установить свойство (* )
Step2 = Object.GetStep2()               Получить свойство (**)
Object.SetStep2 (Step2)                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step2 (&Step2)               Получить свойство
Object.put_Step2(Step2)                 Установить свойство
```

Step3 – Шаг третьей строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step3 = Object.Step3                    Получить свойство(* )
Object.Step3 = Step3                    Установить свойство (* )
Step3 = Object.GetStep3()               Получить свойство (**)
Object.SetStep3 (Step3)                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step3 (&Step3)               Получить свойство
Object.put_Step3(Step3)                 Установить свойство
```

StepAfterParagraph – Дополнительный шаг после абзаца

Интерфейс...

Тип данных: double

Синтаксис Automation:

StepAfterParagraph = Object.StepAfterParagraph	Получить свойство(*)
Object.StepAfterParagraph = StepAfterParagraph	Установить свойство (*)
StepAfterParagraph = Object.GetStepAfterParagraph()	Получить свойство (**)
Object.SetStepAfterParagraph (StepAfterParagraph)	Установить свойство (**)

Синтаксис COM:

Object.get_StepAfterParagraph (&StepAfterParagraph)	Получить свойство
Object.put_StepAfterParagraph(StepAfterParagraph)	Установить свойство

StepBeforeParagraph – Дополнительный шаг перед абзацем

Интерфейс...

Тип данных: double

Синтаксис Automation:

StepBeforeParagraph = Object.StepBeforeParagraph	Получить свойство(*)
Object.StepBeforeParagraph = StepBeforeParagraph	Установить свойство (*)
StepBeforeParagraph = Object.GetStepBeforeParagraph()	Получить свойство (**)
Object.SetStepBeforeParagraph (StepBeforeParagraph)	Установить свойство (**)

Синтаксис COM:

Object.get_StepBeforeParagraph (&StepBeforeParagraph)	Получить свойство
Object.put_StepBeforeParagraph (StepBeforeParagraph)	Установить свойство

Tabulators – Коллекция позиции табуляторов

Интерфейс...

Тип данных: указатель на интерфейс ITabulators

Синтаксис Automation:

Tabulators = Object.Tabulators	Получить свойство(*)
Tabulators = Object.GetTabulators()	Получить свойство (**)

Синтаксис COM:

Object.get_Tabulators (&Tabulators)	Получить свойство

Примечание:

Свойство доступно только для чтения.

WidthFactor1 – Коэффициент сужения первой строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

WidthFactor1 = Object.WidthFactor1	Получить свойство(*)
Object.WidthFactor1 = WidthFactor1	Установить свойство (*)
WidthFactor1 = Object.GetWidthFactor1()	Получить свойство (**)
Object.SetWidthFactor1 (WidthFactor1)	Установить свойство (**)

Синтаксис COM:

Object.get_WidthFactor1 (&WidthFactor1)	Получить свойство
Object.put_WidthFactor1(WidthFactor1)	Установить свойство

WidthFactor2 – Коэффициент сужения второй строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

WidthFactor2 = Object.WidthFactor2	Получить свойство(*)

Object.WidthFactor2 = WidthFactor2	Установить свойство (*)
WidthFactor2 = Object.GetWidthFactor2()	Получить свойство (**)
Object.SetWidthFactor2 (WidthFactor2)	Установить свойство (**)

Синтаксис COM:

Object.get_WidthFactor2 (&WidthFactor2)	Получить свойство
Object.put_WidthFactor2(WidthFactor2)	Установить свойство

WidthFactor3 – Коэффициент сужения третьей строки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

WidthFactor3 = Object.WidthFactor3	Получить свойство(*)
Object.WidthFactor3 = WidthFactor3	Установить свойство (*)
WidthFactor3 = Object.GetWidthFactor3()	Получить свойство (**)
Object.SetWidthFactor3 (WidthFactor3)	Установить свойство (**)

Синтаксис COM:

Object.get_WidthFactor3 (&WidthFactor3)	Получить свойство
Object.put_WidthFactor3(WidthFactor3)	Установить свойство

Интерфейс ITextTable

Справка системы КОМПАС...

`kompas.chm::/597_70_2_Tablicy_v_tekstovom_do.htm`

Таблица в тексте текстового документа.

Иерархия:

IDispatch

IKompasAPIObject

ITextTable

Позволяет вставить таблицу в текстовый документ.

Примечание:

Интерфейс можно получить с помощью метода IText::AddTable и свойства ITextLine::TextLineData.

ITextTable – свойства

FixedCellsSize – Зафиксировать габариты ячеек

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedCellsSize = Object.FixedCellsSize	Получить свойство(*)
Object.FixedCellsSize = FixedCellsSize	Установить свойство (*)
FixedCellsSize = Object.GetFixedCellsSize()	Получить свойство (**)
Object.SetFixedCellsSize(FixedCellsSize)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedCellsSize(&FixedCellsSize)	Получить свойство
Object.put_FixedCellsSize(FixedCellsSize)	Установить свойство

FixedRowCount – Запретить изменять число строк

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedRowCount = Object.FixedRowCount	Получить свойство(*)
Object.FixedRowCount = FixedRowCount	Установить свойство (*)
FixedRowCount = Object.GetFixedRowCount()	Получить свойство (**)
Object.SetFixedRowCount(FixedRowCount)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedRowCount(&FixedRowCount)	Получить свойство
Object.put_FixedRowCount(FixedRowCount)	Установить свойство

FixedColumnCount – Запретить изменять число столбцов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedColumnCount = Object.FixedColumnCount	Получить свойство(*)
Object.FixedColumnCount = FixedColumnCount	Установить свойство (*)
FixedColumnCount = Object.GetFixedColumnCount()	Получить свойство (**)
Object.SetFixedColumnCount(FixedColumnCount)	Установить свойство (**)

Синтаксис COM:

Object.get_FixedColumnCount(&FixedColumnCount) Получить свойство
Object.put_FixedColumnCount(FixedColumnCount) Установить свойство

ITextTable – методы

Save – Сохранить таблицу в файл

Интерфейс...

Синтаксис Automation:

BOOL Save(BSTR FileName);

Синтаксис COM:

HRESULT Save(BSTR FileName, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

FileName - имя сохраняемого файла.

Интерфейс ITabulators

Интерфейс коллекции позиций табулятора.

Иерархия:

IKompasAPIObject
    IKompasCollection
        ITabulators

Описание:

Позволяет получить доступ к коллекции позиций табулятора, считывать параметры позиций, добавлять и удалять позиции из коллекции.

ITabulators – свойства

Item – Возвращает позицию табулятора, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс ITabulators.

Синтаксис Automation:

Item = Object.Item(Index) Получить свойство(*)
Item = Object.GetItem(Index) Получить свойство (**)

Синтаксис COM:

Object.get_Item(Index, &Item) Получить свойство

Примечание:

Свойство доступно только для чтения.

ITabulators – методы

Add – Создает новую позицию табулятора и добавляет ее в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(long Position);

Синтаксис COM:

HRESULT Add(long Position, ITabulator ** Result);

Возвращаемое значение:

- указатель на интерфейс ITabulator.

Входные параметры:

Position - позиция.

Clear – Очищает коллекцию, удаляет из коллекции все позиции табулятора

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IHypertextReferenceParam

Справка системы КОМПАС...

kompas.chm::/566_67_3_Ssylki.htm

Интерфейс параметров вставки ссылки в текст

Иерархия:

IDispatch

IHypertextReferenceParam

Интерфейс является дополнительным для интерфейса ITextItem.

IHypertextReferenceParam – свойства

Brackets – Признак скобок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Brackets = Object.Brackets	Получить свойство(*)
Object.Brackets = Brackets	Установить свойство (*)
Brackets = Object.GetBrackets()	Получить свойство (**)
Object.SetBrackets(Brackets)	Установить свойство (**)

Синтаксис COM:

Object.get_Brackets(&Brackets)	Получить свойство
Object.put_Brackets(Brackets)	Установить свойство

Свойство позволяет заключить содержимое ссылки в скобки. Доступность свойства зависит от типа источника.

HypertextType – Тип ссылки

Интерфейс...

Тип данных: из перечисления ksHypertextTypeEnum

Синтаксис Automation:

HypertextType = Object.HypertextType	Получить свойство(*)
Object.HypertextType = HypertextType	Установить свойство (*)
HypertextType = Object.GetHypertextType()	Получить свойство (**)
Object.SetHypertextType(HypertextType)	Установить свойство (**)

Синтаксис COM:

```
Object.get_HypertextType( &HypertextType )      Получить свойство
Object.put_HypertextType(  HypertextType )      Установить свойство
```

Свойство позволяет указать, что будет являться содержимым ссылки. Набор вариантов зависит от типа источника.

LinkObject – Объект, на который сделана ссылка

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
LinkObject = Object.LinkObject                  Получить свойство(* )
Object.LinkObject = LinkObject                  Установить свойство (* )
LinkObject = Object.GetLinkObject()             Получить свойство (**)
Object.SetLinkObject( LinkObject )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LinkObject( &LinkObject )            Получить свойство
Object.put_LinkObject(  LinkObject )            Установить свойство
```

Свойство позволяет устанавливать и получать объект-источник ссылки.

Precission – Точность вывода числовых значений

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Precission = Object.Precission                  Получить свойство(* )
Object.Precission = Precission                  Установить свойство (* )
Precission = Object.GetPrecission()             Получить свойство (**)
Object.SetPrecission( Precission )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Precission( &Precission )            Получить свойство
Object.put_Precission(  Precission )            Установить свойство
```

Свойство позволяет устанавливать и получать количество знаков после запятой, если источником ссылки является переменная.

PropertyId – Идентификатор свойства

Интерфейс...

Тип данных: double

Синтаксис Automation:

PropertyId = Object.PropertyId	Получить свойство(*)
Object.PropertyId = PropertyId	Установить свойство (*)
PropertyId = Object.GetPropertyId()	Получить свойство (**)
Object.SetPropertyId(PropertyId)	Установить свойство (**)

Синтаксис COM:

Object.get_PropertyId(&PropertyId)	Получить свойство
Object.put_PropertyId(PropertyId)	Установить свойство

TextLineIndex – Индекс строки в тексте

Интерфейс...

Тип данных: long

Синтаксис Automation:

TextLineIndex = Object.TextLineIndex	Получить свойство(*)
Object.TextLineIndex = TextLineIndex	Установить свойство (*)
TextLineIndex = Object.GetTextLineIndex()	Получить свойство (**)
Object.SetTextLineIndex(TextLineIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_TextLineIndex(&TextLineIndex)	Получить свойство
Object.put_TextLineIndex(TextLineIndex)	Установить свойство

Свойство позволяет выбрать номер строки, если источник ссылки содержит несколько строк текста.

IHypertextReferenceParam – методы

Destroy – Разрушить ссылку

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Метод позволяет удалить связь между ссылкой и ее источником. В результате разрушения содержимое ссылки становится обычным текстом. Обновление разрушенной ссылки становится невозможным.

Документ 3D

Вспомогательные объекты, 3D-кривые и элементы тела

Контейнер

Интерфейс IAuxiliaryGeomContainer

Интерфейс контейнера объектов вспомогательной геометрии.

Иерархия:

IDispatch

IAuxiliaryGeomContainer

Описание:

Позволяет получить коллекции объектов вспомогательной геометрии (ЛСК, сплайн и т.д).

Примечание:

Дополнительный интерфейс компонента. Данный интерфейс можно получить у компонента IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IAuxiliaryGeomContainer - свойства

Arcs3D - Интерфейс коллекции дуг 3D

Интерфейс...

Тип данных: указатель на интерфейс IArcs3D

Синтаксис Automation:

```
Arcs3D = Object.Arcs3D      Получить свойство (* )
Arcs3D                  =   Получить свойство (**)
Object.GetArcs3D()
```

Синтаксис COM:

```
Object.get_Arcs3D(          Получить свойство (* )
&Arcs3D )
```

Примечание:

1. Свойство позволяет получить коллекцию всех дуг, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

Axes3D – Интерфейс коллекции вспомогательных осей 3D

Интерфейс...

Тип данных: указатель на интерфейс IAxes3D

Синтаксис Automation:

```
Axes3D = Object.Axes3D;                Получить свойство (* )
Axes3D = Object.GetAxes3D();           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Axes3D( &Axes3D );          Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

ConjunctivePoints – Интерфейс коллекции присоединительных точек

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/Postroenye_kontr_tochek.htm#connecting_point`

Тип данных: указатель на интерфейс IConjunctivePoints

Синтаксис Automation:

```
ConjunctivePoints = Object.ConjunctivePoints;    Получить свойство (* )
ConjunctivePoints                          =     Получить свойство (**)
Object.GetConjunctivePoints();
```

Синтаксис COM:

```
Object->get_ConjunctivePoints(                   Получить свойство (* )
&ConjunctivePoints )
```

Примечание:

Свойство доступно только для чтения.

ConnectCurves – Интерфейс коллекции операций соединения кривых

Интерфейс...

Тип данных: указатель на интерфейс IConnectCurves

Синтаксис Automation:

```
ConnectCurves = Object.ConnectCurves                    Получить свойство (* )
ConnectCurves = Object.GetConnectCurves()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ConnectCurves( &ConnectCurves )              Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех операций соединения кривых, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

Contours3D – Интерфейс коллекции контуров 3D

Интерфейс...

Тип данных: указатель на интерфейс IContours3D

Синтаксис Automation:

```
Contours3D = Object.Contours3D                          Получить свойство (* )
Contours3D = Object.GetContours3D()                     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Contours3D( &Contours3D )                    Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех контуров 3D, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

ControlPoints – Интерфейс коллекции контрольных точек

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/Postroenye_kontr_tochek.htm`

Тип данных: указатель на интерфейс IControlPoints

Синтаксис Automation:

```
ControlPoints = Object.ControlPoints;                   Получить свойство (* )
ControlPoints = Object.GetControlPoints();              Получить свойство (**)
```

Синтаксис COM:

Object->get_ControlPoints(&ControlPoints)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

CurveByLaws – Интерфейс коллекции кривых по закону

Интерфейс...

Тип данных: указатель на интерфейс ICurveByLaws

Синтаксис Automation:

CurveByLaws = Object.CurveByLaws	Получить свойство (*)
CurveByLaws = Object.GetCurveByLaws()	Получить свойство (**)

Синтаксис COM:

Object.get_CurveByLaws(&CurveByLaws)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить коллекцию всех кривых по закону, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

CurveOutLines – Интерфейс коллекции линий очерка

Интерфейс...

Тип данных: указатель на интерфейс ICurveOutLines

Синтаксис Automation:

CurveOutLines = Object.CurveOutLines	Получить свойство (*)
CurveOutLines = Object.GetCurveOutLines()	Получить свойство (**)

Синтаксис COM:

Object.get_CurveOutLines(&CurveOutLines)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить коллекцию всех линий очерка, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

CurvesBy2Projectionses – Интерфейс коллекции кривых по двум проекциям

Интерфейс...

Тип данных: указатель на интерфейс ICurvesBy2Projectionses

Синтаксис Automation:

```
CurvesBy2Projectionses = Получить свойство (* )
Object.CurvesBy2Projectionses;
CurvesBy2Projectionses = Получить свойство (**)
Object.GetCurvesBy2Projectionses();
```

Синтаксис COM:

```
Object.get_CurvesBy2Projectionses( Получить свойство (* )
&CurvesBy2Projectionses );
```

Примечание:

Свойство доступно только для чтения.

Equidistants3D – Интерфейс коллекции 3D–эквидистант

Интерфейс...

Тип данных: указатель на интерфейс IEquidistants3D

Синтаксис Automation:

```
Equidistants3D = Object.Equidistants3D Получить свойство (* )
Equidistants3D = Object.GetEquidistants3D() Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Equidistants3D( &Equidistants3D ) Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию 3D-эквидистант.
2. Свойство доступно только для чтения.

FilletCurves – Интерфейс коллекции операций скругления кривых

Интерфейс...

Тип данных: указатель на интерфейс IFilletCurves

Синтаксис Automation:

```
FilletCurves = Object.FilletCurves              Получить свойство (* )
FilletCurves = Object.GetFilletCurves()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_FilletCurves( &FilletCurves )        Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех операций скругления кривых, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

IsoparametricCurves – Интерфейс коллекции изопараметрических кривых

Интерфейс...

Тип данных: указатель на интерфейс IIsoparametricCurves

Синтаксис Automation:

```
IsoparametricCurves = Object.IsoparametricCurves            Получить свойство (* )
IsoparametricCurves = Object.GetIsoparametricCurves()       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsoparametricCurves( &IsoparametricCurves )      Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех изопараметрических кривых, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

IsoparametricCurvesSets – Интерфейс коллекции групп изопараметрических кривых

Интерфейс...

Тип данных: указатель на интерфейс IIsoparametricCurvesSets

Синтаксис Automation:

```
IsoparametricCurvesSets = Object.IsoparametricCurvesSets          Получить свойство (* )
IsoparametricCurvesSets = Object.GetIsoparametricCurvesSets()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsoparametricCurvesSets( &IsoparametricCurvesSets )    Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех групп изопараметрических кривых, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

LineSegments3D – Интерфейс коллекции отрезков 3D

Интерфейс...

Тип данных: указатель на интерфейс ILineSegments3D

Синтаксис Automation:

```
LineSegments3D = Object.LineSegments3D                Получить свойство (* )
LineSegments3D = Object.GetLineSegments3D()           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_LineSegments3D( &LineSegments3D )          Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех отрезков 3D, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

LocalCoordinateSystems – Интерфейс коллекции локальных систем координат

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/108_8_7_Ispolqzovanie_LSK.htm`

Тип данных: указатель на интерфейс ЛСК ILocalCoordinateSystems

Синтаксис Automation:

```
LocalCoordinateSystems                =   Получить свойство (* )
Object.LocalCoordinateSystems;
Object.LocalCoordinateSystems         =   Получить свойство (**)
LocalCoordinateSystems;
```

Синтаксис COM:

```
Object->get_LocalCoordinateSystems();     Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

Planes3D – Интерфейс коллекции плоскостей

Интерфейс...

Тип данных: указатель на интерфейс IPlanes3D

Синтаксис Automation:

Planes3D = Object.Planes3D	Получить свойство (*)
Planes3D = Object.GetPlanes3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Planes3D(&Planes3D)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

PointsArrsFromFiles – Интерфейс коллекции групп точек из файлов

Интерфейс...

Тип данных: указатель на интерфейс IPointsArrsFromFiles

Синтаксис Automation:

PointsArrsFromFiles = Object.PointsArrsFromFiles	Получить свойство (*)
PointsArrsFromFiles = Object.GetPointsArrsFromFiles()	Получить свойство (**)

Синтаксис COM:

Object.get_PointsArrsFromFiles(&PointsArrsFromFiles)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

PointsArrsOnCurves – Интерфейс коллекции групп точек по кривым

Интерфейс...

Тип данных: указатель на интерфейс IPointsArrsOnCurves

Синтаксис Automation:

PointsArrsOnCurves = Object.PointsArrsOnCurves	Получить свойство (*)
PointsArrsOnCurves = Object.GetPointsArrsOnCurves()	Получить свойство (**)

Синтаксис COM:

Object.get_PointsArrsOnCurves(&PointsArrsOnCurves) Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

PointsArrsOnSurfaces – Интерфейс коллекции групп точек по поверхности

Интерфейс...

Тип данных: указатель на интерфейс IPointsArrsOnSurfaces

Синтаксис Automation:

PointsArrsOnSurfaces = Object.PointsArrsOnSurfaces Получить свойство (*)
PointsArrsOnSurfaces = Object.GetPointsArrsOnSurfaces() Получить свойство (**)

Синтаксис COM:

Object.get_PointsArrsOnSurfaces(&PointsArrsOnSurfaces) Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

PolyLines – Интерфейс коллекции пространственных ломаных

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/CM_CCURVE_POLYLINE_BY_VERTEX.htm

Тип данных: указатель на интерфейс IPolyLines

Синтаксис Automation:

PolyLines = Object.PolyLines; Получить свойство (*)
Object.PolyLines = PolyLines; Получить свойство (**)

Синтаксис COM:

Object->get_PolyLines(); Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

ProjectionCurves – Интерфейс коллекции проекционных кривых

Интерфейс...

Тип данных: указатель на интерфейс IProjectionCurves

Синтаксис Automation:

```
ProjectionCurves = Object.ProjectionCurves;        Получить свойство (* )
ProjectionCurves = Object.GetProjectionCurves();   Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ProjectionCurves( &ProjectionCurves );   Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

Spirals3D – Интерфейс коллекции спиралей 3D

Интерфейс...

Тип данных: указатель на интерфейс ISpirals3D

Синтаксис Automation:

```
Spirals3D = Object.Spirals3D          Получить свойство (* )
Spirals3D = Object.GetSpirals3D()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Spirals3D( &Spirals3D )    Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

Splines3D – Интерфейс коллекции пространственных сплайнов

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/1091_116_6_Splajn.htm

Тип данных: указатель на интерфейс ISplines3D

Синтаксис Automation:

```
Splines3D = Object.Splines3D;          Получить свойство (* )
Object.Splines3D = Splines3D;          Получить свойство (**)
```

Синтаксис COM:

```
Object->get_Splines3D();               Получить свойство (* )
```

Примечание:

Свойство доступно только для чтения.

SplinesOnSurfaces – Интерфейс коллекции сплайнов на поверхностях

Интерфейс...

Тип данных: указатель на интерфейс ISplinesOnSurfaces

Синтаксис Automation:

```
SplinesOnSurfaces = Object.SplinesOnSurfaces          Получить свойство (* )
SplinesOnSurfaces = Object.GetSplinesOnSurfaces()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SplinesOnSurfaces( &SplinesOnSurfaces )    Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех сплайнов на поверхностях, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

SplitLines – Интерфейс коллекции линий разъема

Интерфейс...

Тип данных: указатель на интерфейс ISplitLines

Синтаксис Automation:

```
SplitLines = Object.SplitLines();          Получить свойство (* )
SplitLines = Object.GetSplitLines();       Получить свойство (**)
```

Синтаксис COM:

```
Object->get_SplitLines( &SplitLines )      Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию линий разъема.
2. Свойство доступно только для чтения.

SurfacesIntersectionCurves – Интерфейс коллекции кривых пересечений поверхностей

Интерфейс...

Тип данных: указатель на интерфейс ISurfacesIntersectionCurves

Синтаксис Automation:

```
SurfacesIntersectionCurves = Object.SurfacesIntersectionCurves    Получить свойство (* )
SurfacesIntersectionCurves = Object.GetSurfacesIntersectionCurves()    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SurfacesIntersectionCurves( &SurfacesIntersectionCurves )    Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию кривых пересечений поверхностей.
2. Свойство доступно только для чтения.

TrimmedCurves – Интерфейс коллекции операций усечения кривой

Интерфейс...

Тип данных: указатель на интерфейс ITrimmedCurves

Синтаксис Automation:

```
TrimmedCurves = Object.TrimmedCurves    Получить свойство (* )
TrimmedCurves = Object.GetTrimmedCurves()    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_TrimmedCurves( &TrimmedCurves )    Получить свойство (* )
```

Примечание:

1. Свойство позволяет получить коллекцию всех операций усечения кривой, входящих в состав данного объекта.
2. Свойство доступно только для чтения.

UnhistoredCurves3D – Интерфейс коллекции кривых без истории

Интерфейс...

Тип данных: указатель на интерфейс IUnhistoredCurves3D

Синтаксис Automation:

UnhistoredCurves3D = Object.UnhistoredCurves3D	Получить свойство (*)
UnhistoredCurves3D = Object.GetUnhistoredCurves3D()	Получить свойство (**)

Синтаксис COM:

Object.get_UnhistoredCurves3D(&UnhistoredCurves3D)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

Вспомогательные объекты

Интерфейс IConjunctivePoints

Справка системы КОМПАС: Команда Присоединительная точка

`kompas.chm::/Postroenye_kontr_tochek.htm#connecting_point`

Интерфейс коллекции присоединительных точек.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IConjunctivePoints

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера объектов вспомогательной геометрии IAuxiliaryGeomContainer::ConjunctivePoints.

IConjunctivePoints – свойства

ConjunctivePoint – Указатель на элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IConjunctivePoint

Синтаксис Automation:

ConjunctivePoint Object.ConjunctivePoint(Index)	=	Получить свойство (*)
ConjunctivePoint Object.GetConjunctivePoint(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ConjunctivePoint(Index, &ConjunctivePoint)	Получить свойство

Входные параметры:

Index	- индекс объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IConjunctivePoints – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IConjunctivePoint ** Result);

Возвращаемое значение:

	- Указатель на интерфейс присоединительной точки IConjunctivePoint.

Примечание:

1. Метод позволяет создать новый интерфейс присоединительной точки.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IModelObject::Update.

Интерфейс IControlPoints

Справка системы КОМПАС: Команда Контрольная точка

kompas.chm::/Postroenye_kontr_tochek.htm#kontrol_point

Интерфейс коллекции контрольных точек.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IControlPoints

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера объектов вспомогательной геометрии IAuxiliaryGeomContainer::ControlPoints.

IControlPoints – свойства

ControlPoint – Указатель на элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IControlPoint

Синтаксис Automation:

ControlPoint Object.ControlPoint(Index)	=	Получить свойство (*)
ControlPoint Object.GetControlPoint(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ControlPoint(&ControlPoint)	Index,	Получить свойство

Входные параметры:

Index	VT_I4 - индекс объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

IControlPoints – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IControlPoint ** Result);

Возвращаемое значение:

- Указатель на интерфейс контрольной точки IControlPoint.

Примечание:

1. Метод позволяет создать новый интерфейс локальной системы координат.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IModelObject::Update.

Интерфейс ILocalCoordinateSystems

Интерфейс коллекции локальных систем координат.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ILocalCoordinateSystems

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера объектов вспомогательной геометрии IAuxiliaryGeomContainer::LocalCoordinateSystems.

ILocalCoordinateSystems – свойства

Current – Указатель на текущую СК

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

LocalCoordinateSystem = iObject.Current()	Получить свойство (*)
LocalCoordinateSystems = iObject.GetCurrent()	Получить свойство (**)

Синтаксис COM:

iObject->get_Current(&LocalCoordinateSystem)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Если на момент получения свойства текущей является ЛСК, то можно привести интерфейс IModelObject, возвращаемый свойством к ILocalCoordinateSystem, с помощью IUnknown::QueryInterface (const GUID far& iid, void** pif). Если же текущей является мировая СК, то привести возвращаемый интерфейс к ILocalCoordinateSystem не удастся.

LocalCoordinateSystem – Указатель на элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

LocalCoordinateSystem iObject.LocalCoordinateSystem(index)	=	Получить свойство (*)
LocalCoordinateSystem iObject.GetLocalCoordinateSystem(index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LocalCoordinateSystem(index, &LocalCoordinateSystem)	Получить свойство

Входные параметры:

Index	- VT_I4 - индекс объекта.

Примечание:

Свойство доступно только для чтения.

ILocalCoordinateSystems – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILocalCoordinateSystem** Result);

Возвращаемое значение:

- Указатель на интерфейс ЛСК ILocalCoordinateSystem.

Примечание:

1. Метод позволяет создать новый интерфейс локальной системы координат.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IModelObject::Update.

SetCurrent – Установить ЛСК в качестве текущей СК

Интерфейс...

Синтаксис Automation:

BOOL SetCurrent(ILocalCoordinateSystem * object);

Синтаксис COM:

HRESULT SetCurrent(ILocalCoordinateSystem * object,
BOOL * Result);

Входные параметры:

- Указатель на интерфейс ЛСК ILocalCoordinateSystem.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет установить ЛСК, передаваемую входным параметром, в качестве текущей СК.
2. Если входным параметром передать NULL, текущей станет мировая СК. См. также ILocalCoordinateSystem::Current.

Интерфейс ILocalCSAxesDirectionParam

Интерфейс параметров ЛСК для типа ориентации "Направление осей".

Иерархия:

IDispatch

IKompasAPIObject

ILocalCSAxesDirectionParam

Примечание:

Указатель на интерфейс можно получить, используя свойство ЛСК ILocalCoordinateSystem::LocalCSParameters.

ILocalCSAxesDirectionParam– свойства

AngleByOwnAxis – Угол поворота вокруг собственной оси

Интерфейс...

Синтаксис Automation:

```
iObject.AngleByOwnAxis = AngleByOwnAxis( axis );      Установить свойство (* )
iObject.SetAngleByOwnAxis( axis, angle );             Установить свойство (**)
```

Синтаксис COM:

```
iObject->put_AngleByOwnAxis( axis, angle );           Установить свойство
```

Входные параметры:

axis	- тип оси из перечисления ksObj3dTypeEnum, допустимые значения: - o3d_axisOX - o3d_axisOY - o3d_axisOZ,
angle	- угол поворота.

Примечание:

Свойство доступно только для записи.

DirectingObject – Направляющий объект для оси

Интерфейс...

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

```
DirectingObject = iObject.DirectingObject( axis );      Получить свойство (* )
DirectingObject = iObject.GetDirectingObject( axis );   Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_DirectingObject( axis, &DirectingObject );   Получить свойство
```

Входные параметры:

axis - тип оси из перечисления ksObj3dTypeEnum,
допустимые значения:
- o3d_axisOX
- o3d_axisOY
- o3d_axisOZ.

Примечание:

Свойство доступно только для чтения.

ILocalCSAxesDirectionParam – методы

RotateAxis – Сменить направление оси на противоположное

Интерфейс...

Синтаксис Automation:

BOOL RotateAxis(ksObj3dTypeEnum axis);

Синтаксис Automation:

HRESULT RotateAxis(ksObj3dTypeEnum axis,
BOOL * Result);

Входные параметры:

axis - тип оси из перечисления ksObj3dTypeEnum,
допустимые значения:
- o3d_axisOX
- o3d_axisOY
- o3d_axisOZ.

Возвращаемое значение:

TRUE - в случае удачного завершения.
FALSE - в случае неудачи.

Примечание:

1. Метод позволяет менять направления осей на противоположные.
2. Для осей можно выбирать направляющие объекты (см. метод ILocalCSAxesDirectionParam::SetDirectingObject).
3. Можно выбрать направление для двух осей, третья ось ориентируется, исходя из положения двух выбранных. Менять направление на противоположное можно только у выбранных осей.
4. У оси с автоориентацией смена направления на противоположное недоступна.

SetDirectingObject – Установить направляющий объект для оси

Интерфейс...

Синтаксис Automation:

BOOL SetDirectingObject(ksObj3dTypeEnum axis,
LPDISPATCH obj);

Синтаксис Automation:

HRESULT SetDirectingObject(ksObj3dTypeEnum axis,
IModelObject * obj,
VARIANT_BOOL * Result);

Входные параметры:

obj	- направляющий объект,
axis	- тип оси из перечисления ksObj3dTypeEnum,
	допустимые значения:
	- o3d_axisOX
	- o3d_axisOY
	- o3d_axisOZ.

Возвращаемое значение:

TRUE	- в случае удачного завершения.
FALSE	- в случае неудачи.

Примечание:

1. Объекты, которые могут определять направление оси: прямолинейные объекты, плоские объекты, цилиндрические и конические поверхности. Также могут быть выбраны произвольные кривые или ребра при условии, что позиция начала ЛСК лежит на этой кривой (ребре).
2. Можно выбрать направление для двух осей. Третья ось ориентируется, исходя из положения двух выбранных. Менять направление на противоположное можно только у выбранных осей. У оси с автоориентацией смена направления на противоположное недоступна.

Интерфейс ILocalCSEulerParam

Интерфейс параметров ЛСК для типа ориентации "Система углов Эйлера".

Иерархия:

IDispatch

IKompasAPIObject

ILocalCSEulerParam

Примечание:

Указатель на интерфейс можно получить, используя свойство ЛСК ILocalCoordinateSystem::LocalCSParameters.

ILocalCSEulerParam – свойства

NutationAngle – Угол нутации

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
NutationAngle = iObject.NutationAngle;          Получить свойство (* )
iObject.NutationAngle = NutationAngle;          Установить свойство (* )
NutationAngle = iObject.GetNutationAngle();     Получить свойство (**)
iObject.SetNutationAngle( NutationAngle );      Установить свойство (* )
```

Синтаксис COM:

```
iObject->get_NutationAngle( &NutationAngle );   Получить свойство
iObject->put_NutationAngle( NutationAngle );    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол нутации.

PrecessionAngle – Угол прецессии

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
PrecessionAngle = iObject.PrecessionAngle;          Получить свойство (* )
iObject.PrecessionAngle = PrecessionAngle;          Установить свойство (* )
PrecessionAngle = iObject.GetPrecessionAngle();     Получить свойство (**)
iObject.SetPrecessionAngle( PrecessionAngle );      Установить свойство (* )
```

Синтаксис COM:

```
iObject->get_PrecessionAngle( &PrecessionAngle );   Получить свойство
iObject->put_PrecessionAngle( PrecessionAngle );    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол прецессии.

RotationAngle – Угол вращения

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
RotationAngle = iObject.RotationAngle;          Получить свойство (* )
iObject.RotationAngle = RotationAngle;          Установить свойство (* )
RotationAngle = iObject.GetRotationAngle();     Получить свойство (**)
iObject.SetRotationAngle( RotationAngle );      Установить свойство (* )
```

Синтаксис COM:

```
iObject->get_RotationAngle( &RotationAngle );   Получить свойство
iObject->put_RotationAngle( RotationAngle );    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол вращения.

См.Рисунок...

Интерфейс ILocalCSOrientByObjectParam

Интерфейс параметров ЛСК для типа ориентации "По объекту".

Иерархия:

IDispatch

IKompasAPIObject

ILocalCSOrientByObjectParam

Примечание:

Указатель на интерфейс можно получить, используя свойство ЛСК ILocalCoordinateSystem::LocalCSParameters.

ILocalCSOrientByObjectParam – свойства

OrientationObject – Получить объект, обладающий ориентацией

Интерфейс...

Синтаксис Automation:

```
OrientationObject = iObject.OrientationObject();      Получить свойство (* )
OrientationObject = iObject.GetOrientationObject();   Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_OrientationObject( &OrientationObject );   Получить свойство
```

Возвращаемое значение:

- Указатель на интерфейс IModelObject

Примечание:

Свойство доступно только для чтения.

ILocalCSOrientByObjectParam – методы

SetOrientationObject – Установить объект, обладающий ориентацией

Интерфейс...

Синтаксис Automation:

HRESULT SetOrientationObject(IModelObject * obj, VARIANT_BOOL * Result);

Синтаксис Automation:

HRESULT SetDirectingObject(ksObj3dTypeEnum axis,
IModelObject * obj,
VARIANT_BOOL * Result);

Входные параметры:

obj - объект, обладающий ориентацией.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Интерфейс ILocalCSObject

Интерфейс подчиненного объекта ЛСК.

Иерархия:

IDispatch

ILocalCSObject

Описание

Интерфейс является дополнительным для следующих объектов:

- IPoint3D - Точка 3D,
- IControlPoint - Контрольная точка,
- IConjunctivePoint - Присоединительная точка,
- ILocalCoordinateSystem - ЛСК,
- ISketch- Эскиз,
- ITablePattern - Массив по таблице,
- IMeshPointsSurface - Поверхность по сети точек,
- ICloudPointsSurface - Поверхность по пласту (облаку) точек,
- IImportedSurface - Импортированная поверхность,
- IArc3D - 3D дуга,

- IPolyLine - Пространственная ломаная,
- ISpline3D - Сплайн.

Данный интерфейс можно получить от интерфейсов перечисленных объектов посредством вызова метода IUnknown::QueryInterface.

Интерфейс позволяет получать и изменять параметры объектов в удобной для этого системе координат. По умолчанию используется СК компонента.

ILocalCSObject – свойства

ModelObjectParamType – Тип параметров объекта

Интерфейс...

Тип данных: из перечисления ksModelObjectParamTypeEnum

Синтаксис Automation:

```
ModelObjectParamType                =   Получить свойство (* )
Object.ModelObjectParamType
Object.ModelObjectParamType         =   Установить свойство (* )
ModelObjectParamType
ModelObjectParamType                =   Получить свойство (**)
Object.GetModelObjectParamType()
Object.SetModelObjectParamType(         Установить свойство (**)
ModelObjectParamType )
```

Синтаксис COM:

```
Object.get_ModelObjectParamType(        Получить свойство
&ModelObjectParamType )
Object.put_ModelObjectParamType(        Установить свойство
ModelObjectParamType )
```

Примечание:

Свойство позволяет установить тип параметров объекта.

После установки данного свойства параметры объекта будут выдаваться пересчитанными в установленную систему координат. При установке параметров также будет производиться пересчет параметров. Для правильной работы библиотек требуется установить нужный тип параметров, получить или изменить параметры и вернуть тип параметров ksMOPartAllParam-Параметры в системе координат детали (см. ksModelObjectParamTypeEnum-Тип параметров объекта).

LocalCoordinateSystem – Тип параметров объекта

Интерфейс...

Тип данных: из перечисления ksModelObjectParamTypeEnum

Синтаксис Automation:

LocalCoordinateSystem = Получить свойство (*)
Object.LocalCoordinateSystem
LocalCoordinateSystem = Получить свойство (**)
Object.GetLocalCoordinateSystem(
)

Синтаксис COM:

Object.get_LocalCoordinateSystem(Получить свойство
&LocalCoordinateSystem)

Примечание:

Свойство позволяет получить локальную систему координат объекта.

Если свойство возвращает NULL, объект создан в базовой СК объекта. Для переноса объекта в другую СК используется метод IPart7::TransferObjects.

Интерфейс IPlacement3D

Интерфейс локальной системы координат (положение объекта).

Иерархия:

IDispatch

IKompasAPIObject

IPlacement3D

IPlacement3D – методы

GetMatrix3D – Получить матрицу системы координат в виде массива. SAFEARRAY double (VT_ARRAY | VT_R8)

Интерфейс...

Синтаксис Automation:

VARIANT GetMatrix3D();

Синтаксис COM:

HRESULT GetMatrix3D(VARIANT * Result);

Возвращаемое значение:

- массив координат SAFEARRAY double (VT_ARRAY | VT_R8).

GetOrigin – Получить координаты начала локальной системы координат

Интерфейс...

Синтаксис Automation:

BOOL GetOrigin(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetOrigin(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Выходные параметры

X - координата точки по X,
Y - координата точки по Y,
Z - координата точки по Z.

GetVector – Получить вектор для указанной оси

Интерфейс...

Синтаксис Automation:

BOOL GetVector(ksObj3dTypeEnum Axis, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetVector(ksObj3dTypeEnum Axis, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры

Axis - ось.

Выходные параметры

X - координата точки по X,
Y - координата точки по Y,
Z - координата точки по Z.

GetPoint3D – Получить пространственную точку по точке на плоскости xy

Интерфейс...

Синтаксис Automation:

BOOL GetPoint3D(double XIn, double YIn, double * XOut, double * YOut, double * ZOut);

Синтаксис COM:

HRESULT GetPoint3D(double XIn, double YIn, double * XOut, double * YOut, double * ZOut, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры

XIn - координата по X на плоскости XY,
YIn - координата по Y на плоскости XY.

Выходные параметры

YIn - координата по X в пространстве,
YOut - координата по Y в пространстве,
ZOut - координата по Z в пространстве.

GetPointProjectionToXY – Проекция точки на плоскость xy

Интерфейс...

Синтаксис Automation:

BOOL GetPointProjectionToXY(double XIn, double YIn, double ZIn, double * XOut, double * YOut);

Синтаксис COM:

HRESULT GetPointProjectionToXY(double XIn, double YIn, double ZIn, double * XOut, double * YOut, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры

X - координата точки по X,
Y - координата точки по Y,
Z - координата точки по Z.

Выходные параметры

XOut - координата по X на плоскости XY,
YOut - координата по Y на плоскости XY.

InitByMatrix3D – Установить систему координат по матрице. SAFEARRAY double (VT_ARRAY | VT_R8)

Интерфейс...

Синтаксис Automation:

BOOL InitByMatrix3D(VARIANT mtr);

Синтаксис COM:

HRESULT InitByMatrix3D(VARIANT mtr, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры

mtr - массив координат SAFEARRAY double (VT_ARRAY | VT_R8).

Rotate – Повернуть систему координат на угол вокруг оси

Интерфейс...

Синтаксис Automation:

BOOL Rotate(double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle);

Синтаксис COM :

HRESULT Rotate(double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

X0 - первая координата точки,
Y0 - вторая координата точки,
Z0 - третья координата точки,
AxisZX - вектор "X" стандартного базиса,
AxisZX Y - вектор "Y" стандартного базиса,
AxisZZ - вектор "Z" стандартного базиса,
Angle - угол поворота.

SetOrigin – Получить координаты начала локальной системы координат

Интерфейс...

Синтаксис Automation:

BOOL SetOrigin(double X, double Y, double Z);

Синтаксис COM:

HRESULT SetOrigin(double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры

X	- координата точки по X,
Y	- координата точки по Y,
Z	- координата точки по Z.

SetVector – Задать вектор для указанной оси

Интерфейс...

Синтаксис Automation:

BOOL SetVector(ksObj3dTypeEnum Axis, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetVector(ksObj3dTypeEnum Axis, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры

Axis	- ось,
X	- координата точки по X,
Y	- координата точки по Y,
Z	- координата точки по Z.

3-D кривые и элементы тела

Интерфейс IModelObjects

Базовый интерфейс для всех коллекций модельных объектов.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

Примечание:

1. Данный интерфейс является базовым для коллекций модельных объектов.
2. Интерфейс может быть получен с помощью метода IUnknown QueryInterface от коллекций модельных объектов, например, IParts7.

IModelObjects – свойства

Item – Объект, заданный по индексу или по имени

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Item iObject.Item (Index)	=	Получить свойство (*)
Item iObject.GetItem (Index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_Item(Index, &Item)	Получить свойство

Входные параметры:

Index (Variant)	- VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

Свойство доступно только для чтения.

Интерфейс IArcs3D

Справка системы КОМПАС...

kompas.chm::/1031_110_2_Duga_okr.htm

Коллекция дуг 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IArcs3D

Интерфейс позволяет получать и создавать дуги 3D.

Интерфейс можно получить в контейнере вспомогательной геометрии с помощью метода IAuxiliaryGeomContainer::Arcs3D.

IArcs3D – свойства

Arc3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IArc3D

Синтаксис Automation:

Arc3D = Object.Arc3D(Index)	Получить свойство (*)
Arc3D = Object.GetArc3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Arc3D(Index, &Arc3D)	Получить свойство

Свойство позволяет получить указатель на интерфейс дуги 3D.

Примечание:

Свойство доступно только для чтения.

IArcs3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IArc3D ** Result);

Возвращаемое значение:

Указатель на интерфейс дуги 3D IArc3D

Интерфейс IAxes3D

Коллекция вспомогательных осей 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IAxes3D

IAxes3D – свойства

Axis3D – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IAxis3D

Синтаксис Automation:

```
Axis3D = Object.Axis3D( Index )        Получить свойство(* )
Axis3D = Object.GetAxis3D( Index )     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Axis3D( Index,      Получить свойство
&Axis3D )
```

Примечание:

Свойство доступно только для чтения.

IAxes3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum Type, IAxis3D ** Result);

Входные параметры:

Type	- тип оси.

Интерфейс IBilletsObsoletes

Коллекция деталей заготовок и зеркальных деталей.

Иерархия:

IDispatch
  IKompasAPIObject
    IKompasCollection
      IModelObjects
        IBilletsObsoletes

IBilletsObsoletes – свойства

BilletObsolete – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: указатель указатель на интерфейс IBilletObsolete

Синтаксис Automation:

BilletObsolete = Object.BilletObsolete(Index)	Получить свойство (*)
BilletObsolete = Object.GetBilletObsolete(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_BilletObsolete(Index, &BilletObsolete)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IBilletsObsoletes – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(BSTR FileName, BOOL Mirror);

Синтаксис COM:

HRESULT Add(BSTR FileName, BOOL Mirror, IBilletObsolete ** Result);

Возвращаемое значение:

Указатель на интерфейс IBilletObsolete

Входные параметры:

FileName	- имя файла детали,
Mirror	- признак зеркальной симметрии.

Интерфейс IBooleans

Интерфейс коллекции булевых операций.

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IBooleans

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Booleans.

КОМПАС версия v18

IBooleans – свойства

Boolean – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IBoolean

Синтаксис Automation:

Boolean = Object.Boolean(Index)	Получить свойство(*)
Boolean = Object.GetBoolean(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Boolean(Index, &Boolean)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IBooleans – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IBoolean ** Result);

Возвращаемое значение:

- указатель на интерфейс IBoolean.

Примечания:

Метод позволяет создать новый интерфейс булевой операции.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IChamfers

Интерфейс коллекции операций Фаска.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IChamfers

Получить интерфейс коллекции операций фаска можно, используя свойство контейнера трехмерных объектов IModelContainer::Chamfers.

КОМПАС версия v18

IChamfers – свойства

Chamfer – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IChamfer

Синтаксис Automation:

Chamfer = Object.Chamfer(Index)	Получить свойство (*)
Chamfer = Object.GetChamfer(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Chamfer(Index, &Chamfer) Получить свойство

Входные параметры:

VARIANT Index - индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IChamfers – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IChamfer ** Result);

Возвращаемое значение:

- указатель на интерфейс IChamfer.

Примечание:

1. Метод позволяет создать новый интерфейс операции фаска.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IFullFillets

Интерфейс коллекции операций полного скругления.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IFullFillets

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::FullFillets.

Версия Компас v18.1

IFullFillets – свойства

FullFillet – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IFullFillet

Синтаксис Automation:

FullFillet = Object.FullFillet(Index)	Получить свойство (*)
FullFillet = Object.GetFullFillet(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_FullFillet(Index, &FullFillet)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IFullFillets – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IFullFillet * * Result);

Возвращаемое значение:

- Указатель на интерфейс IFullFillet.

Примечания:

1. Метод позволяет создать новый интерфейс операции полного скругления.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ICollectionsGeometry

Коллекции геометрии.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ICollectionsGeometry

Интерфейс можно получить у интерфейса контейнера 3D объектов IModelContainer с помощью свойства IModelContainer::CollectionsGeometry.

ICollectionsGeometry – свойства

CollectionGeometry – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: указатель указатель на интерфейс ICollectionGeometry

Синтаксис Automation:

```
CollectionGeometry = Object.CollectionGeometry( Index )    Получить свойство (* )
CollectionGeometry = Object.GetCollectionGeometry( Index )    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_CollectionGeometry( Index, &CollectionGeometry )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ICollectionsGeometry – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICollectionGeometry ** Result);

Возвращаемое значение:

Указатель на интерфейс ICollectionGeometry

Интерфейс IConnectCurves

Справка системы КОМПАС...

kompas.chm::/1008_108_11_Soedinenie_curves.htm

Коллекция операций соединения кривых

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IConnectCurves

Интерфейс позволяет получать и создавать операции соединения кривых.

Интерфейс можно получить в контейнере вспомогательной геометрии с помощью метода IAuxiliaryGeomContainer::ConnectCurves.

IConnectCurves – свойства

ConnectCurve – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель указатель на интерфейс IConnectCurve

Синтаксис Automation:

ConnectCurve = Object.ConnectCurve(Index)	Получить свойство (*)
ConnectCurve = Object.GetConnectCurve(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_ConnectCurve(Index, &ConnectCurve)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IConnectCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IConnectCurve * * Result);

Возвращаемое значение:

Указатель на интерфейс операции соединения кривых IConnectCurve

Интерфейс IContours3D

Справка системы КОМПАС...

kompas.chm::/604_Glava59_Kontur.htm

Коллекция контуров 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IContours3D

Интерфейс позволяет получать и создавать контуры в трехмерном пространстве.

Интерфейс можно получить в контейнере вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::Contours3D

IContours3D – свойства

Contour3D – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: указатель указатель на интерфейс IContour3D

Синтаксис Automation:

Contour3D = Object.Contour3D(Index)	Получить свойство (*)
Contour3D = Object.GetContour3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Contour3D(Index, &Contour3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IContours3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IContour3D ** Result);

Возвращаемое значение:

Указатель на интерфейс контура IContour3D

Интерфейс ICopiesGeometry

Коллекция копий геометрии

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ICopiesGeometry

Интерфейс можно получить у интерфейса контейнера 3D объектов IModelContainer с помощью свойства IModelContainer::CopiesGeometry.

ICopiesGeometry – свойства

CopyGeometry – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: указатель указатель на интерфейс ICopyGeometry

Синтаксис Automation:

CopyGeometry = Object.CopyGeometry(Index)	Получить свойство (*)
CopyGeometry = Object.GetCopyGeometry(Index	Получить свойство (**)

Синтаксис COM:

Object.get_CopyGeometry(Index, &CopyGeometry)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ICopiesGeometry – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICopyGeometry ** Result);

Возвращаемое значение:

Указатель на интерфейс ICopyGeometry.

Интерфейс ICurveByLaws

Справка системы КОМПАС...

kompas.chm::/582_46_10_Krivay_po_zakonu.htm

Коллекция кривых по закону.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ICurveByLaws

Примечание:

Интерфейс можно получить у контейнера объектов вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::CurveByLaws.

ICurveByLaws – свойства

CurveByLaw – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс ICurveByLaw

Синтаксис Automation:

CurveByLaw = Object.CurveByLaw(Index)	Получить свойство (*)
CurveByLaw = Object.GetCurveByLaw(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_CurveByLaw(Index, &CurveByLaw)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ICurveByLaws – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICurveByLaw ** Result);

Возвращаемое значение:

Указатель на интерфейс кривой по закону ICurveByLaw

Интерфейс ICurvesBy2Projectionses

Коллекция кривых по двум проекциям

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ICurvesBy2Projectionses

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::CurvesBy2Projectionses.

ICurvesBy2Projectionses – свойства

CurveBy2Projections – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс ICurveBy2Projections

Синтаксис Automation:

CurveBy2Projections = Object.CurveBy2Projections(Index)	Получить свойство (*)

CurveBy2Projections = Object.GetCurveBy2Projections(Получить свойство (**) Index)

Синтаксис COM:

Object.get_CurveBy2Projections(Получить свойство Index, &CurveBy2Projections)

Примечание:

Свойство доступно только для чтения.

ICurvesBy2Projectionses – методы

Add – Создает новый элемент и добавляет его в коллекцию.

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICurveBy2Projections ** Result);

Возвращаемое значение:

Указатель на интерфейс ICurveBy2Projections

Интерфейс ICurveOutLines

Справка системы КОМПАС...

kompas.chm::/516_43_12_Liniy_ocherka.htm

Коллекция линий очерка.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ICurveOutLines

Примечание:

Интерфейс можно получить у контейнера объектов вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::CurveOutLines.

ICurveOutLines – свойства

CurveOutLine – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс ICurveOutLine

Синтаксис Automation:

```
CurveOutLine = Object.CurveOutLine( Index )          Получить свойство (* )
CurveOutLine = Object.GetCurveOutLine( Index )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_CurveOutLine( Index, &CurveOutLine )      оПолучить свойство
```

Примечание:

Свойство доступно только для чтения.

ICurveOutLines – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICurveOutLine ** Result);

Возвращаемое значение:

Указатель на интерфейс линии очерка ICurveOutLine

Интерфейс ICuts

Интерфейс коллекции операций Сечение.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ICuts

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Cuts.

КОМПАС версияv18

ICuts – свойства

Cut – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс ICut.

Синтаксис Automation:

Cut = Object.Cut(Index)	Получить свойство (*)
Cut = Object.GetCut(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Cut(Index, &Cut)	Получить свойство

Входные параметры:

VARIANT Index	- индекс в коллекции или имя объекта.

Примечание:

Свойство доступно только для чтения.

ICuts – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICut ** Result);

Возвращаемое значение:

- указатель на интерфейс ICut.

Примечания:

Метод позволяет создать новый интерфейс операции сечения.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IEquidistants3D

Коллекция 3D-эквидистант.

Иерархия

IKompasAPIObject

IKompasCollection

IModelObjects

IEquidistants3D

Описание:

Позволяет создавать и получать эквидистанты в 3D документе.

Примечание:

Данный интерфейс можно получить у контейнера вспомогательной геометрии IAuxiliaryGeomContainer::Equidistants3D.

IEquidistants3D – свойства

Equidistant3D – Возвращает 3D–эквидистанту, заданную по индексу или имени

Интерфейс...

Тип данных: указатель на интерфейс 3D-эквидистанты IEquidistant3D

Синтаксис Automation:

Equidistant3D iObject.Equidistant3D(Index)	=	Получить свойство (*)
Equidistant3D iObject.GetEquidistant3D(Index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_Equidistant3D(Index, &Equidistant3D)	Получить свойство

Входные параметры:

Index	- индекс в коллекции или имя объекта.

Примечание:

Свойство доступно только для чтения.

IEquidistants3D – методы

Add – Создать новый элемент и добавить в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEquidistant3D ** Result);

Возвращаемое значение:

- указатель на интерфейс 3D-эквидистанты IEquidistant3D.

Интерфейс IEvolutions

Интерфейс коллекции кинематических операций.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IEvolutions

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Evolutions.

Интерфейс также используется для коллекции кинематических поверхностей ISurfaceContainer::EvolutionSurfaces.

КОМПАС версия v18

IEvolutions – свойства

Evolution – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IEvolution.

Синтаксис Automation:

```
Evolution = Object.Evolution( Index )    Получить свойство (* )
Evolution = Object.GetEvolution( Index )    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Evolution( Index, &Evolution )    Получить свойство
```

Входные параметры:

```
VARIANT Index    - индекс или имя элемента.
```

Примечание:

Свойство доступно только для чтения.

IEvolutions – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum EvolutionType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum EvolutionType, IEvolution ** Result);

Возвращаемое значение:

- указатель на интерфейс IEvolution.

Входные параметры:

EvolutionType	- тип объекта из перечисления ksObj3dTypeEnum.

Примечания:

1. Метод позволяет создать новый интерфейс кинематической операции.
2. Допустимыми значениями EvolutionType являются:

- o3d_bossEvolution, o3d_cutEvolution для коллекции операций IModelContainer::Evolutions
- o3d_EvolutionSurface для коллекции поверхностей ISurfaceContainer::EvolutionSurfaces.

3. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IFillets

Интерфейс коллекции операций скруглений.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IFillets

Получить интерфейс коллекции операций скруглений можно, используя свойство контейнера трехмерных объектов IModelContainer::Fillets.

КОМПАС версия v18

IFillets – свойства

Fillet – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IFillet.

Синтаксис Automation:

Fillet = Object.Fillet(Index)	Получить свойство (*)
Fillet = Object.GetFillet(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Fillet(Index, &Fillet)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IFillets – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IFillet ** Result);

Возвращаемое значение:

- указатель на интерфейс IFillet.

Примечания:

1. Метод позволяет создать новый интерфейс операции скругления.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IFilletCurves

Справка системы КОМПАС...

kompas.chm::/CM_CURVEOPER_CORNER.htm

Коллекция операций скругления кривых.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IFilletCurves

Данный интерфейс можно получить у контейнера вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::FilletCurves.

IFilletCurves – свойства

FilletCurve – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IFilletCurve

Синтаксис Automation:

FilletCurve = Object.FilletCurve(Index)	Получить свойство (*)
FilletCurve = Object.GetFilletCurve(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_FilletCurve(Index, &FilletCurve)	Получить свойство

Свойство позволяет получить указатель на интерфейс операции скругления кривых.

Примечание:

Свойство доступно только для чтения.

IFilletCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IFilletCurve * * Result);

Возвращаемое значение:

Указатель на интерфейс операции скругления кривых IFilletCurve.

Интерфейс IHoles3D

Интерфейс коллекции отверстий.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IHoles3D

Примечание:

Интерфейс можно получить с помощью метода

IModelContainer::Holes3D

IHoles3D – свойства

Hole3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IHole3D

Синтаксис Automation:

Hole3D = Object.Hole3D(Index)	Получить свойство (*)
Hole3D = Object.GetHole3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Hole3D(Index, &Hole3D)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

IHoles3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IHole3D ** Result);

Возвращаемое значение:

- Указатель на интерфейс IHole3D

Интерфейс IInclines

Интерфейс коллекции операций Уклон.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IInclines

Получить интерфейс коллекции операций уклон можно, используя свойство контейнера трехмерных объектов IModelContainer::Inclines.

КОМПАС версия v18

IInclines – свойства

Incline – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IIncline.

Синтаксис Automation:

Incline = Object.Incline(Index)	Получить свойство (*)
Incline = Object.GetIncline(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Incline(Index, &Incline)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IInclines – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IIncline ** Result);

Возвращаемое значение:

- указатель на интерфейс IIncline.

Примечания:

1. Метод позволяет создать новый интерфейс операции **Уклон**.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IIsoparametricCurves

Справка системы КОМПАС...

kompas.chm::/462_Isoparametricheskie_krivye.htm#izoparam_curve

Коллекция изопараметрических кривых.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IIsoparametricCurves

Примечание:

Интерфейс можно получить у контейнера объектов вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::IsoparametricCurves.

IIsoparametricCurves – свойства

IsoparametricCurve – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IIsoparametricCurve

Синтаксис Automation:

IsoparametricCurve = Object.IsoparametricCurve(Index) Получить свойство (*)
IsoparametricCurve = Object.GetIsoparametricCurve(Index) Получить свойство (**)

Синтаксис COM:

Object.get_IsoparametricCurve(Index, &IsoparametricCurve) Получить свойство

Примечание:

Свойство доступно только для чтения.

IIsoparametricCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IIsoparametricCurve ** Result);

Возвращаемое значение:

Указатель на интерфейс изопараметрической кривой IIsoparametricCurve

Интерфейс IIsoparametricCurvesSets

Справка системы КОМПАС...

kompas.chm::/462_Isoparametricheskie_krivye.htm#set_izo_curves

Коллекция групп изопараметрических кривых.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IIsoparametricCurvesSets

Примечание:

Интерфейс можно получить у контейнера объектов вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::IsoparametricCurvesSets.

IIsoparametricCurvesSets – свойства

IsoparametricCurvesSet – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IIsoparametricCurvesSet

Синтаксис Automation:

```
IsoparametricCurvesSet = Получить свойство (* )
Object.IsoparametricCurvesSet( Index )
IsoparametricCurvesSet = Получить свойство (**)
Object.GetIsoparametricCurvesSet(
Index )
```

Синтаксис COM:

```
Object.get_IsoparametricCurvesSet( Index, Получить свойство
&IsoparametricCurvesSet )
```

Примечание:

Свойство доступно только для чтения.

IIsoparametricCurvesSets – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IIsoparametricCurvesSet ** Result);

Возвращаемое значение:

Указатель на интерфейс группы изопараметрических кривых IIsoparametricCurvesSet

Интерфейс IJointSurfaces

Интерфейс коллекции поверхностей соединения.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IJointSurfaces

IJointSurfaces – свойства

JointSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IJointSurface.

Синтаксис Automation:

```
JointSurface = Object.JointSurface( Index )          Получить свойство (* )
JointSurface = Object.GetJointSurface( Index )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_JointSurface( Index, &JointSurface )      Получить свойство
```

Входные параметры:

```
VARIANT Index                - индекс или имя элемента.
```

Примечание:

Свойство доступно только для чтения.

IJointSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IJointSurface * * Result);

Возвращаемое значение:

- указатель на интерфейс IJointSurface.

Примечания:

1. Метод позволяет создать новый интерфейс операции **Поверхность соединения**.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IJointSurface

Интерфейс операции Поверхность соединения.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IJointSurface

IJointSurface свойства

Curves1 – Кривая границы 1

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Curves1 = Object.Curves1          Получить свойство (* )
Object.Curves1 = Curves1          Установить свойство (* )
Curves1 = Object.GetCurves1()     Получить свойство (**)
Object.SetCurves1( Curves1 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Curves1(               Получить свойство
&Curves1 )
Object.put_ModelObjec             Установить свойство
Object.put_Curves1(
Curves1 )
```

Curves2 – Кривая границы 2

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Curves2 = Object.Curves2          Получить свойство (* )
Object.Curves2 = Curves2          Установить свойство (* )
Curves2 = Object.GetCurves2()     Получить свойство (**)
Object.SetCurves2( Curves2 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Curves2(               Получить свойство
&Curves2 )
```

Object.put_ModelObjec	Установить свойство
Object.put_Curves2(Curves2)	

Face1 – Грань границы 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Face1 = Object.Face1	Получить свойство (*)
Object.Face1 = Face1	Установить свойство (*)
Face1 = Object.GetFace1()	Получить свойство (**)
Object.SetFace1(Face1)	Установить свойство (**)

Синтаксис COM:

Object.get_Face1(&Face1)	Получить свойство
Object.put_Face1(Face1)	Установить свойство

Face2 – Грань границы 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Face2 = Object.Face2	Получить свойство (*)
Object.Face2 = Face2	Установить свойство (*)
Face2 = Object.GetFace2()	Получить свойство (**)
Object.SetFace2(Face2)	Установить свойство (**)

Синтаксис COM:

Object.get_Face2(&Face2)	Получить свойство
Object.put_Face2(Face2)	Установить свойство

Face1ConnectType – Условие сопряжения соединяющей поверхности с поверхностью сопряжения 1

Интерфейс...

Тип данных: Значение из перечисления ksConnectTypeEnum.

Синтаксис Automation:

```
Face1ConnectType                    =   Получить свойство (* )
Object.Face1ConnectType
Object.Face1ConnectType             =   Установить свойство (* )
Face1ConnectType
Face1ConnectType                    =   Получить свойство (**)
Object.GetFace1ConnectType()
Object.SetFace1ConnectType(             Установить свойство (**)
Face1ConnectType )
```

Синтаксис COM:

```
Object.get_Face1ConnectType(        Получить свойство
&Face1ConnectType )
Object.put_Face1ConnectType(        Установить свойство
Face1ConnectType )
```

Face2ConnectType – Условие сопряжения соединяющей поверхности с поверхностью сопряжения 2

Интерфейс...

Тип данных: Значение из перечисления ksConnectTypeEnum.

Синтаксис Automation:

```
Face2ConnectType            =   Получить свойство (* )
Object.Face2ConnectType
Object.Face2ConnectType     =   Установить свойство (* )
Face2ConnectType
Face2ConnectType            =   Получить свойство (**)
Object.GetFace2ConnectType()
Object.SetFace2ConnectType(     Установить свойство (**)
Face2ConnectType )
```

Синтаксис COM:

```
Object.get_Face2ConnectType(    Получить свойство
&Face2ConnectType )
Object.put_Face2ConnectType(    Установить свойство
Face2ConnectType )
```

Sense1 – Сменить направление сопряжения вдоль границы 1

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Sense1 = Object.Sense1          Получить свойство (* )
Object.Sense1 = Sense1          Установить свойство (* )
Sense1 = Object.GetSense1()     Получить свойство (**)
Object.SetSense1( Sense1 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sense1(              Получить свойство
&Sense1 )
Object.put_Sense1(              Установить свойство
Sense1 )
```

Sense2 – Сменить направление сопряжения вдоль границы 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Sense2 = Object.Sense2          Получить свойство (* )
Object.Sense2 = Sense2          Установить свойство (* )
Sense2 = Object.GetSense2()     Получить свойство (**)
Object.SetSense2( Sense2 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sense2(              Получить свойство
&Sense2 )
Object.put_Sense2(              Установить свойство
Sense2 )
```

Tension1 – Натяжение поверхности вдоль границы 1 %

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Tension1 = Object.Tension1              Получить свойство (* )
Object.Tension1 = Tension1              Установить свойство (* )
Tension1 = Object.GetTension1()         Получить свойство (**)
Object.SetTension1( Tension1 )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Tension1(        Получить свойство
&Tension1 )
Object.put_Tension1(        Установить свойство
Tension1 )
```

Tension2 – Натяжение поверхности вдоль границы 2 %

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Tension2 = Object.Tension2              Получить свойство (* )
Object.Tension2 = Tension2              Установить свойство (* )
Tension2 = Object.GetTension2()         Получить свойство (**)
Object.SetTension2( Tension2 )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Tension2(        Получить свойство
&Tension2 )
Object.put_Tension2(        Установить свойство
Tension2 )
```

Интерфейс ILineSegments3D

Справка системы КОМПАС...

`kompas.chm::/454_48_3_otrezok.htm`

Коллекция контуров 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ILineSegments3D

Интерфейс позволяет получать и создавать отрезки в трехмерном пространстве.

Интерфейс можно получить в контейнере вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::LineSegments3D.

ILineSegments3D – свойства

LineSegment3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель указатель на интерфейс ILineSegment3D

Синтаксис Automation:

```
LineSegment3D = Object.LineSegment3D( Index )          Получить свойство (* )
LineSegment3D = Object.GetLineSegment3D( Index )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_LineSegment3D( Index, &LineSegment3D )      Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ILineSegments3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILineSegment3D * * Result);

Возвращаемое значение:

Указатель на интерфейс отрезка ILineSegment3D

Интерфейс ILofts

Интерфейс коллекции операций По сечениям.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ILofts

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Lofts.

Интерфейс также используется для коллекции поверхностей по сечениям ISurfaceContainer::LoftSurfaces.

КОМПАС версия v18

ILofts – свойства

Loft – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ILoft.

Синтаксис Automation:

```
Loft = Object.Loft( Index );                    Получить свойство (* )
Loft = Object.GetLoft( Index );                 Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Loft( Index, &Loft )                 Получить свойство
```

Входные параметры:

```
VARIANT Index                                   - индекс или имя элемента.
```

Примечание:

Свойство доступно только для чтения.

ILofts – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum LoftType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum LoftType, ILoft ** Result);

Возвращаемое значение:

- Указатель на интерфейс ILoft.

Входные параметры:

LoftType	- тип объекта из перечисления ksObj3dTypeEnum.

Примечания:

1. Метод позволяет создать новый интерфейс операции по сечениям.
2. Допустимыми значениями LoftType являются:

- o3d_bossLoft, o3d_cutLoft для коллекции операций IModelContainer::Lofts ,
- o3d_LoftSurface для коллекции поверхностей ISurfaceContainer::LoftSurfaces.

3. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс INurbsSurfacesByCurvesMeshs

Интерфейс коллекции сплайновых поверхностей по сетке кривых.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
INurbsSurfacesByCurvesMeshs

Данный интерфейс можно получить, используя свойство контейнера поверхностей ISurfaceContainer::NurbsSurfacesByCurvesMeshs

КОМПАС версия v18

INurbsSurfacesByCurvesMeshs – свойства

NurbsSurfaceByCurvesMesh – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс INurbsSurfaceByCurvesMesh.

Синтаксис Automation:

NurbsSurfaceByCurvesMesh Object.NurbsSurfaceByCurvesMesh(Index) Получить свойство (*);	=	Получить свойство (*)
NurbsSurfaceByCurvesMesh Object.GetNurbsSurfaceByCurvesMesh(Index) Получить свойство (**);	=	Получить свойство (**)

Синтаксис COM:

Object.get_NurbsSurfaceByCurvesMesh(Index, &NurbsSurfaceByCurvesMesh) Получить свойство

Примечание:

Свойство доступно только для чтения.

INurbsSurfacesByCurvesMeshs – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(INurbsSurfaceByCurvesMesh * * Result);

Возвращаемое значение:

- указатель на интерфейс INurbsSurfaceByCurvesMesh

Примечания:

1. Метод позволяет создать новый интерфейс сплайновой поверхности по сетке кривых.
2. После получения нового интерфейса нужно задать параметры поверхности и вызвать метод IModelObject::Update.

Интерфейс INurbsSurfaceByCurvesMesh

Интерфейс операции Сплайновая поверхность по сетке кривых.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

INurbsSurfaceByCurvesMesh

Данный интерфейс можно получить с помощью метода коллекции операций пластина INurbsSurfacesByCurvesMeshs::Add или свойства INurbsSurfacesByCurvesMeshs::NurbsSurfaceByCurvesMesh

КОМПАС версия v18

INurbsSurfaceByCurvesMesh – свойства

CheckSelfIntersection – Проверка самопересечения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CheckSelfIntersection                =  Получить свойство (* )
Object.CheckSelfIntersection
Object.CheckSelfIntersection         =  Установить свойство (* )
CheckSelfIntersection
CheckSelfIntersection                =  Получить свойство (**)
Object.GetCheckSelfIntersection()
Object.SetCheckSelfIntersection(        Установить свойство (**)
CheckSelfIntersection )
```

Синтаксис COM:

```
Object.get_CheckSelfIntersection(    Получить свойство
&CheckSelfIntersection )
Object.put_CheckSelfIntersection(    Установить свойство
CheckSelfIntersection )
```

ConnectSurface – Поверхность сопряжения

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
ConnectSurface                       =  Получить свойство (* )
Object.ConnectSurface( Index )
Object.ConnectSurface( Index )       =  Установить свойство (* )
ConnectSurface
ConnectSurface                       =  Получить свойство (**)
Object.GetConnectSurface( Index )
Object.SetConnectSurface( Index,     Установить свойство (**)
ConnectSurface )
```

Синтаксис COM:

```
Object.get_ConnectSurface( Index,    Получить свойство
&ConnectSurface )
Object.put_ConnectSurface( Index,    Установить свойство
ConnectSurface )
```

Входные параметры:

long Index - индекс.

ConnectType – Условие сопряжения

Интерфейс...

Тип данных: из перечисления ksConnectTypeEnum

Синтаксис Automation:

ConnectType = Object.ConnectType(Index)	=	Получить свойство (*)
Object.ConnectType(Index) = ConnectType	=	Установить свойство (*)
ConnectType = Object.GetConnectType(Index)	=	Получить свойство (**)
Object.SetConnectType(Index, ConnectType)		Установить свойство (**)

Синтаксис COM:

Object.get_ConnectType(Index, &ConnectType)	Получить свойство
Object.put_ConnectType(Index, ConnectType)	Установить свойство

Входные параметры:

long Index - индекс.

UClosed – Замкнутость по направлении U

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UClosed = Object.UClosed	Получить свойство (*)
Object.UClosed = UClosed	Установить свойство (*)
UClosed = Object.GetUClosed()	Получить свойство (**)
Object.SetUClosed(UClosed)	Установить свойство (**)

Синтаксис COM:

Object.get_UClosed(&UClosed)	Получить свойство

Object.put_UClosed(UClosed)	Установить свойство

VClosed – Замкнутость по направлении V

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

VClosed = Object.VClosed	Получить свойство (*)
Object.VClosed = VClosed	Установить свойство (*)
VClosed = Object.GetVClosed()	Получить свойство (**)
Object.SetVClosed(VClosed)	Установить свойство (**)

Синтаксис COM:

Object.get_VClosed(&VClosed)	Получить свойство
Object.put_VClosed(VClosed)	Установить свойство

UCurves – Список кривых в направлении U SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

UCurves = Object.UCurves	Получить свойство (*)
Object.UCurves = UCurves	Установить свойство (*)
UCurves = Object.GetUCurves()	Получить свойство (**)
Object.SetUCurves(UCurves)	Установить свойство (**)

Синтаксис COM:

Object.get_UCurves(&UCurves)	Получить свойство
Object.put_UCurves(UCurves)	Установить свойство

VCurves – Список кривых в направлении V SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
VCurves = Object.VCurves                Получить свойство (* )
Object.VCurves = VCurves                Установить свойство (* )
VCurves = Object.GetVCurves()           Получить свойство (**)
Object.SetVCurves( VCurves )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_VCurves( &VCurves )      Получить свойство
Object.put_VCurves(  VCurves )      Установить свойство
```

Интерфейс IPointsArrsOnSurfaces

Коллекция групп точек по поверхностям.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IPointsArrsOnSurfaces

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::PointsArrsOnSurfaces.

IPointsArrsOnSurfaces – свойства

PointsArrOnSurface – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IPointsArrOnSurface

Синтаксис Automation:

```
PointsArrOnSurface = Object.PointsArrOnSurface( Index );        Получить свойство (* )
PointsArrOnSurface = Object.GetPointsArrOnSurface( Index );     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_PointsArrOnSurface( Index, &PointsArrOnSurface )      Получить свойство
```

Примечание:

Свойство доступно только для чтения.

IPointsArrsOnSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IPointsArrOnSurface ** Result);

Возвращаемое значение:

- указатель на интерфейс IPointsArrOnSurface.

Интерфейс IPointsArrsOnCurves

Коллекция групп точек по кривым.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IPointsArrsOnCurves

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::PointsArrsOnCurves.

IPointsArrsOnCurves – свойства

PointsArrOnCurve – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IPointsArrOnCurve

Синтаксис Automation:

PointsArrOnCurve = Object.PointsArrOnCurve(Index);	Получить свойство (*)
PointsArrOnCurve = Object.GetPointsArrOnCurve(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_PointsArrOnCurve(Index, &PointsArrOnCurve)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPointsArrsOnCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IPointsArrOnCurve ** Result);

Возвращаемое значение:

- указатель на интерфейс IPointsArrOnCurve.

Интерфейс IPointsArrsFromFiles

Коллекция групп точек из файла.

Иерархия:

IDispatch
  IKompasAPIObject
    IKompasCollection
      IModelObjects
        IPointsArrsFromFiles

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::PointsArrsFromFiles.

IPointsArrsFromFiles – свойства

PointsArrFromFile – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IPointsArrFromFile

Синтаксис Automation:

PointsArrFromFile = Object.PointsArrFromFile(Index);	Получить свойство (*)
PointsArrFromFile = Object.GetPointsArrFromFile(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_PointsArrFromFile(Index, &PointsArrFromFile) Получить свойство

Примечание:

Свойство доступно только для чтения.

IPointsArrsFromFiles – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IPointsArrFromFile * * Result);

Возвращаемое значение:

- указатель на интерфейс IPointsArrFromFile.

Интерфейс IPolyLines

Справка системы КОМПАС: Команда Ломаная

`kompas.chm::/CM_CCURVE_POLYLINE_BY_VERTEX.htm`

Интерфейс коллекции элементов "пространственная ломаная".

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

IPolyLines

Описание:

Интерфейс позволяет получить коллекции элементов "пространственная ломаная".

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера объектов вспомогательной геометрии IAuxiliaryGeomContainer::PolyLines.

IPolyLines – свойства

PolyLine – Пространственная ломаная, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс IPolyLine

Синтаксис Automation:

PolyLine = iObject.PolyLine(Index); Получить свойство (*)
PolyLine = iObject.GetPolyLine(Index); Получить свойство (**)

Синтаксис COM:

iObject->get_PolyLine(Index, Получить свойство
&PolyLine)

Входные параметры:

Index - VT_BSTR - имя объекта,
VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IPolyLines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IPolyLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс пространственной ломаной IPolyLine.

Интерфейс IProjectionCurves

Интерфейс коллекции проекционных кривых.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IProjectionCurves

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::ProjectionCurves.

IProjectionCurves – свойства

ProjectionCurve – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: указатель на интерфейс IProjectionCurve

Синтаксис Automation:

ProjectionCurve = Object.ProjectionCurve(index)	=	Получить свойство (*)
ProjectionCurve = Object.GetProjectionCurve(index)	=	Получить свойство (**)

Синтаксис COM:

Object->get_ProjectionCurve(index, &ProjectionCurve)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IProjectionCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IProjectionCurve** Result);

Возвращаемое значение:

- Указатель на интерфейс IProjectionCurve.

Интерфейс IRibs

Интерфейс коллекции ребер жесткости.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IRibs

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Ribs.

КОМПАС версия v18

IRibs – свойства

Rib – Возвращает элемент, заданный по индексу или имени

Интерфейс...

Тип данных: Указатель на интерфейс IRib.

Синтаксис Automation:

Rib = Object.Rib(Index)	Получить свойство (*)
Rib = Object.GetRib(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Rib(Index, &Rib)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IRibs – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRib ** Result);

Возвращаемое значение:

- указатель на интерфейс IRib.

Примечания:

Метод позволяет создать новый интерфейс операции ребро жесткости.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IShells

Интерфейс коллекции операций Оболочка.

Иерархия:

IDispatch
    IKompasAPIObject
        IKompasCollection
            IModelObjects
                IShells

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Shells.

КОМПАС версия v18

IShells – свойства

Shell – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IShell.

Синтаксис Automation:

```
Shell = Object.Shell( Index )       Получить свойство (* )
Shell = Object.GetShell( Index )    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Shell( Index, &Shell )          Получить свойство
```

Входные параметры:

```
    VARIANT          - индекс или имя элемента.
    Index
```

Примечание:

Свойство доступно только для чтения.

IShells – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IShell ** Result);

Возвращаемое значение:

- указатель на интерфейс IShell.

Примечания:

Метод позволяет создать новый интерфейс операции оболочка.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISplines3D

Справка системы КОМПАС

`kompas.chm::/1091_116_6_Splajn.htm`

Интерфейс коллекции элементов "сплайн".

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

ISplines3D

Описание:

Интерфейс позволяет получить элементы "сплайн".

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера вспомогательной геометрии IAuxiliaryGeomContainer::Splines3D.

ISplines3D – свойства

Spline3D – Сплайн, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpline3D

Синтаксис Automation:

Spline3D = Object.Spline3D(index) Получить свойство (*)

Spline3D = Object.GetSpline3D(index) Получить свойство (**)

Синтаксис COM:

Object->get_Spline3D(&Spline3D)	index,	Получить свойство

Входные параметры:

Index	VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ISplines3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(ISpline3D** Result);

Возвращаемое значение:

- Указатель на интерфейс сплайна ISpline3D.

Convert – Создает сплайны по объектам и добавляет их в коллекцию

Интерфейс...

Синтаксис Automation:

VARIANT Convert(VARIANT Objects);

Синтаксис COM:

HRESULT Convert(VARIANT Objects, VARIANT * Result);

Входной параметр:

Objects	- объект VT_DISPATCH или массив объектов VT_ARRAY | VT_DISPATCH.

Возвращаемое значение:

- объект VT_DISPATCH или массив объектов VT_ARRAY | VT_DISPATCH.

Интерфейс ISplinesOnSurfaces

Справка системы КОМПАС...

kompas.chm::/CM_SPLINE_ON_SURFACE.htm

Коллекция сплайнов на поверхностях.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISplinesOnSurfaces

Примечание:

Интерфейс можно получить у контейнера объектов вспомогательной геометрии с помощью свойства IAuxiliaryGeomContainer::SplinesOnSurfaces.

ISplinesOnSurfaces – свойства

SplineOnSurface – Возвращает элемент, заданный по ссылке или индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISplineOnSurface

Синтаксис Automation:

SplineOnSurface = Object.SplineOnSurface(Index)	Получить свойство (*)
SplineOnSurface = Object.GetSplineOnSurface(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_SplineOnSurface(Index, &SplineOnSurface)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ISplinesOnSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISplineOnSurface * * Result);

Возвращаемое значение:

Указатель на интерфейс сплайна на поверхности ISplineOnSurface.

Интерфейс ISurfacesIntersectionCurves

Интерфейс коллекции NURBS-поверхностей.

Иерархия

IKompasAPIObject

IKompasCollection

IModelObjects

ISurfacesIntersectionCurves

Позволяет получать и создавать кривые пересечения поверхностей.

Примечание:

Данный интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::SurfacesIntersectionCurves.

ISurfacesIntersectionCurves - свойства

SurfacesIntersectionCurve - Возвращает элемент, заданный по имени или индексу

Интерфейс...

Тип данных: указатель на интерфейс ISurfacesIntersectionCurve

Синтаксис Automation:

SurfacesIntersectionCurve = Object.SurfacesIntersectionCurve(Index)	Получить свойство (*)
SurfacesIntersectionCurve = Object.GetSurfacesIntersectionCurve(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_SurfacesIntersectionCurve(Index, &SurfacesIntersectionCurve)	Получить свойство

Входные параметры:

Index - имя или индекс объекта в коллекции.

Свойство позволяет получить интерфейс кривой пересечения поверхностей по имени или индексу.

ISurfacesIntersectionCurves – методы

Add – Создает новый объект «кривая пересечения поверхностей» и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISurfacesIntersectionCurve **Object);

Возвращаемое значение:

- Указатель на интерфейс кривой пересечения поверхностей ISurfacesIntersectionCurve.

После добавления объекта требуется задать параметры и вызвать метод IModelObject::Update

Интерфейс ISpirals3D

Коллекция спиралей 3D.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
ISpirals3D

Примечание:

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::Spirals3D.

ISpirals3D – свойства

Spiral3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISpiral3D

Синтаксис Automation:

Spiral3D = Object.Spiral3D(Index)	Получить свойство (*)
Spiral3D = Object.GetSpiral3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Spiral3D(&Spiral3D)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

ISpirals3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum Type, ISpiral3D ** Result);

Возвращаемое значение:

Указатель на интерфейс ISpiral3D

Входные параметры:

Type - тип создаваемой спирали (коническая или цилиндрическая)

Интерфейс ITrimmedCurves

Справка системы КОМПАС...

`kompas.chm::/CM_CURVEOPER_CUT.htm`

Коллекция операций усечения кривой

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ITrimmedCurves

ITrimmedCurves – свойства

TrimmedCurve – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ITrimmedCurve

Синтаксис Automation:

```
TrimmedCurve = Object.TrimmedCurve( Index )    Получить свойство (* )
TrimmedCurve = Object.GetTrimmedCurve( Index )    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_TrimmedCurve( Index, &TrimmedCurve )    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ITrimmedCurves – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ITrimmedCurve ** Result);

Возвращаемое значение:

Указатель на интерфейс усеченной кривой ITrimmedCurve.

Интерфейс IUnhistoredCurves3D

Интерфейс коллекции кривых без истории.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IUnhistoredCurves3D

Интерфейс можно получить с помощью свойства IAuxiliaryGeomContainer::UnhistoredCurves3D.

IUnhistoredCurves3D – свойства

UnhistoredCurve3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IUnhistoredCurve3D

Синтаксис Automation:

UnhistoredCurve3D Object.UnhistoredCurve3D(Index)	=	Получить свойство (*)
UnhistoredCurve3D Object.GetUnhistoredCurve3D(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_UnhistoredCurve3D(&UnhistoredCurve3D)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IUnhistoredCurves3D – методы

Add – Создает новую кривую по массиву точек

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(VARIANT Points);

Синтаксис COM:

HRESULT Add(VARIANT Points, IUnhistoredCurve3D ** Result);

Возвращаемое значение:

- Указатель на интерфейс IUnhistoredCurve3D

Входные параметры:

Points	- массив точек VT_ARRAY I VT_R8.

Convert – Создает кривую без истории по кривой

Интерфейс...

Синтаксис Automation:

LPDISPATCH Convert(IModelObject * Curve, BOOL DeleteSource);

Синтаксис COM:

HRESULT Convert(IModelObject * Curve, BOOL DeleteSource, IUnhistoredCurve3D ** Result);

Возвращаемое значение:

указатель на интерфейс кривой без истории IUnhistoredCurve3D

Входные параметры:

Curve	- указатель на исходную кривую,
DeleteSource	- удалить исходную кривую.

Load – Прочитать кривые из файла

Интерфейс...

Синтаксис Automation:

VARIANT Load(BSTR FileName, BOOL SewCurves);

Синтаксис COM:

HRESULT Load(BSTR FileName, BOOL SewCurves, VARIANT * Result);

Возвращаемое значение:

- массив указателей на кривые VT_ARRAY | VT_DISPATCH.

Входные параметры:

FileName	- имя файла,
SewCurves	- сшивать кривые.

Интерфейс IUserFolders

Интерфейс коллекции пользовательских директорий.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IUserFolders

Данный интерфейс можно получить с помощью метода IPart7::UserFolders

Версия: КОМПАС v18.

IUserFolders – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IUserFolder * * Result);

Возвращаемое значение:

- указатель на интерфейс IUserFolder.

Интерфейс IUserObjects3D

Интерфейс коллекции пользовательских объектов 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IUserObjects3D

IUserObjects3D – свойства

UserObject3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IUserObject3D

Синтаксис Automation:

UserObject3D = Object.UserObject3D(Index) Получить свойство (*)

UserObject3D = Object.GetUserObject3D(Index) Получить свойство (**)

Синтаксис COM:

Object.get_UserObject3D(&UserObject3D)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IUserObjects3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IUserObject3D ** Result);

Возвращаемое значение:

Указатель на интерфейс пользовательского объекта IUserObject3D.

Интерфейс IArc3D

Справка системы КОМПАС...

`kompas.chm::/1031_110_2_Duga_okr.htm`

Интерфейс параметров 3D дуги.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IArc3D
      - ILocalCSObject

Примечание:

Интерфейс можно получить у коллекции дуг, используя свойство IArcs3D::Arc3D или метод IArcs3D::Add.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IArc3D – свойства

Angle1 – Начальный угол дуги 3D

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle1 = Object.Angle1          Получить свойство (* )
Object.Angle1 = Angle1          Установить свойство (* )
Angle1 = Object.GetAngle1()     Получить свойство (**)
Object.SetAngle1( Angle1 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle1( &Angle1 )    Получить свойство
Object.put_Angle1(  Angle1 )    Установить свойство
```

Angle2 – Конечный угол дуги 3D

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle2 = Object.Angle2          Получить свойство (* )
Object.Angle2 = Angle2          Установить свойство (* )
Angle2 = Object.GetAngle2()     Получить свойство (**)
Object.SetAngle2( Angle2 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle2( &Angle2 )    Получить свойство
Object.put_Angle2(  Angle2 )    Установить свойство
```

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
AssociationObject                       =   Получить свойство (* )
Object.AssociationObject( Index )
Object.AssociationObject(  Index  )     =   Установить свойство (* )
AssociationObject
AssociationObject                       =   Получить свойство (**)
Object.GetAssociationObject( Index )
Object.SetAssociationObject(  Index,        Установить свойство (**)
AssociationObject )
```

Синтаксис COM:

```
Object.get_AssociationObject(  Index,       Получить свойство
&AssociationObject )
```

Object.put_AssociationObject(Index, AssociationObject) Установить свойство

Входные параметры

Index - индекс вершины из перечисления ksArc3DParameterEnum

Свойство позволяет связать вершину дуги с уже имеющимся точечным объектом.

BuildingObject – Вспомогательный объект для построения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BuildingObject = Object.BuildingObject Получить свойство (*)
Object.BuildingObject = BuildingObject Установить свойство (*)
BuildingObject = Object.GetBuildingObject() Получить свойство (**)
Object.SetBuildingObject(BuildingObject) Установить свойство (**)

Синтаксис COM:

Object.get_BuildingObject(&BuildingObject) Получить свойство
Object.put_BuildingObject(BuildingObject) Установить свойство

BuildingType – Способ построения дуги 3D

Интерфейс...

Тип данных: из перечисления ksArc3DBuildingTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType Получить свойство (*)
Object.BuildingType = BuildingType Установить свойство (*)
BuildingType = Object.GetBuildingType() Получить свойство (**)
Object.SetBuildingType(BuildingType) Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType) Получить свойство
Object.put_BuildingType(BuildingType) Установить свойство

BuildingVectorParameters – Параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

```
BuildingVectorParameters = Получить свойство (* )
Object.BuildingVectorParameters
BuildingVectorParameters = Получить свойство (**)
Object.GetBuildingVectorParameters()
```

Синтаксис COM:

```
Object.get_BuildingVectorParameters( Получить свойство
&BuildingVectorParameters )
```

Свойство позволяет задать направление дуги с помощью вектора.

Примечание

Свойство доступно только для чтения.

Closed – Признак замкнутости дуги

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Closed = Object.Closed                  Получить свойство (* )
Object.Closed = Closed                  Установить свойство (* )
Closed = Object.GetClosed()             Получить свойство (**)
Object.SetClosed( Closed )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Closed( &Closed )            Получить свойство
Object.put_Closed(  Closed )            Установить свойство
```

Direction – Направление построения дуги

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction        Получить свойство (* )
Object.Direction = Direction        Установить свойство (* )
Direction                    =      Получить свойство (**)
Object.GetDirection()
Object.SetDirection( Direction )    Установить свойство (**)
```

Синтаксис COM:

Object.get_Direction(&Direction)		Получить свойство
Object.put_Direction(Direction)		Установить свойство

Значения свойства:

TRUE	- дуга по умолчанию,
FALSE	- дополняющая дуга.

DirectionPointIndex – Индекс вершины, для которой задано направление

Интерфейс...

Тип данных: из перечисления ksArc3DParameterEnum

Синтаксис Automation:

DirectionPointIndex Object.DirectionPointIndex	=	Получить свойство (*)
Object.DirectionPointIndex DirectionPointIndex	=	Установить свойство (*)
DirectionPointIndex Object.GetDirectionPointIndex()	=	Получить свойство (**)
Object.SetDirectionPointIndex(DirectionPointIndex)		Установить свойство (**)

Синтаксис COM:

Object.get_DirectionPointIndex(&DirectionPointIndex)	Получить свойство
Object.put_DirectionPointIndex(DirectionPointIndex)	Установить свойство

Point3DParametersType – Тип параметров построения точки

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

Point3DParametersType Object.Point3DParametersType(Index)	=	Получить свойство (*)
Object.Point3DParametersType(Index) Point3DParametersType	=	Установить свойство (*)

Point3DParametersType = Object.GetPoint3DParametersType(Index)		Получить свойство (**)
Object.SetPoint3DParametersType(Index, Point3DParametersType)		Установить свойство (**)

Синтаксис COM:

Object.get_Point3DParametersType(&Point3DParametersType)	Index,	Получить свойство
Object.put_Point3DParametersType(Point3DParametersType)	Index,	Установить свойство

Входные параметры:

Index - индекс вершины из перечисления ksArc3DParameterEnum

Point3DParameters – Получить интерфейс параметров точки

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Point3DParameters = Object.Point3DParameters(Index)		Получить свойство (*)
Point3DParameters = Object.GetPoint3DParameters(Index)		Получить свойство (**)

Синтаксис COM:

Object.get_Point3DParameters(&Point3DParameters)	Index,	Получить свойство

Входные параметры:

Index - индекс вершины из перечисления ksArc3DParameterEnum.

Свойство позволяет задать вершину дуги, используя способы построения точки.

Примечание

Свойство доступно только для чтения.

Radius – Радиус дуги 3D

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство (*)
Object.Radius = Radius	Установить свойство (*)

Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство
Object.put_Radius(Radius)	Установить свойство

IArc3D – методы

GetPoint – Получить координаты точки дуги 3D

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(ksArc3DParameterEnum Index, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(ksArc3DParameterEnum Index, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index - индекс вершины из перечисления ksArc3DParameterEnum.

Выходные параметры

X, Y, Z - координаты точки.

SetPoint – Установить координаты точки дуги 3D

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(ksArc3DParameterEnum Index, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(ksArc3DParameterEnum Index, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

Index	- индекс вершины из перечисления ksArc3DParameterEnum,
X, Y, Z	- координаты точки.

Метод позволяет также задавать углы и радиус, которые будут построены путем проецирования точки на дугу.

Интерфейс IAxis3D

Вспомогательная ось 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IAxis3D

IAxis3D – свойства

MathCurve – Получить интерфейс математической кривой

Интерфейс...

Тип данных: указатель на интерфейс IMathCurve3D

Синтаксис Automation:

MathCurve = Object.MathCurve	Получить свойство (*)
MathCurve = Object.GetMathCurve()	Получить свойство (**)

Синтаксис COM:

Object.get_MathCurve(&MathCurve)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Интерфейс IAxisLine3D

Осевая линия 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IAxis3D

IAxisLine3D

IAxisLine3D – свойства

Object1 – Вершина 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Object1 = Object.Object1	Получить свойство (*)
Object.Object1 = Object1	Установить свойство (*)
Object1 = Object.GetObject1()	Получить свойство (**)
Object.SetObject1(Object1)	Установить свойство (**)

Синтаксис COM:

Object.get_Object1(&Object1)	Получить свойство
Object.put_Object1(Object1)	Установить свойство

Object2 – Вершина 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Object2 = Object.Object2	Получить свойство (*)
Object.Object2 = Object2	Установить свойство (*)
Object2 = Object.GetObject2()	Получить свойство (**)
Object.SetObject2(Object2)	Установить свойство (**)

Синтаксис COM:

Object.get_Object2(&Object2)	Получить свойство
Object.put_Object2(Object2)	Установить свойство

Интерфейс IAxis3DBy2Points

Ось через две вершины.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IAxis3D
      - IAxis3DBy2Points

IAxis3DBy2Points – свойства

Point1 – Вершина 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Point1 = Object.Point1                  Получить свойство (* )
Object.Point1 = Point1                  Установить свойство (* )
Point1 = Object.GetPoint1()             Получить свойство (**)
Object.SetPoint1( Point1 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Point1( &Point1 )            Получить свойство
Object.put_Point1(  Point1 )            Установить свойство
```

Point2 – Вершина 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Point2 = Object.Point2                  Получить свойство (* )
Object.Point2 = Point2                  Установить свойство (* )
Point1 = Object.GetPoint2()             Получить свойство (**)
Object.SetPoint2( Point2 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Point2( &Point2 )            Получить свойство
Object.put_Point2(  Point2 )            Установить свойство
```

Интерфейс IAxis3DBy2Planes

Ось на пересечении плоскостей.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IAxis3D
      - IAxis3DBy2Planes

IAxis3DBy2Planes – свойства

Plane1 – Плоскость 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane1 = Object.Plane1	Получить свойство (*)
Object.Plane1 = Plane1	Установить свойство (*)
Plane1 = Object.GetPlane1()	Получить свойство (**)
Object.SetPlane1(Plane1)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane1(&Plane1)	Получить свойство
Object.put_Plane1(Plane1)	Установить свойство

Plane2 – Плоскость 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane2 = Object.Plane2	Получить свойство (*)
Object.Plane2 = Plane2	Установить свойство (*)
Plane2 = Object.GetPlane2()	Получить свойство (**)
Object.SetPlane2(Plane2)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane2(&Plane2)	Получить свойство
Object.put_Plane2(Plane2)	Установить свойство

Интерфейс IAxis3DByConeface

Ось конической поверхности.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IAxis3D
      - IAxis3DByConeface

IAxis3DByConeface – свойства

Face – Поверхность

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Face = Object.Face	Получить свойство (*)
Object.Face = Face	Установить свойство (*)
Face = Object.GetFace()	Получить свойство (**)
Object.SetFace(Face)	Установить свойство (**)

Синтаксис COM:

Object.get_Face(&Face)	Получить свойство
Object.put_Face(Face)	Установить свойство

Интерфейс IAxis3DByEdge

Ось через ребро.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IAxis3D
      - IAxis3DByEdge

IAxis3DByEdge – свойства

Edge – Ребро

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Edge = Object.Edge	Получить свойство (*)
Object.Edge = Edge	Установить свойство (*)
Edge = Object.GetEdge()	Получить свойство (**)
Object.SetEdge(Edge)	Установить свойство (**)

Синтаксис COM:

Object.get_Edge(&Edge)	Получить свойство
Object.put_Edge(Edge)	Установить свойство

Интерфейс IAxis3DByPointAndObject

Ось через вершину по объекту.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IAxis3D

IAxis3DByPointAndObject

IAxis3DByPointAndObject – свойства

DirectObject – Направляющий объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

DirectObject = Object.DirectObject	Получить свойство (*)
Object.DirectObject = DirectObject	Установить свойство (*)
DirectObject = Object.GetDirectObject()	Получить свойство (**)
Object.SetDirectObject(DirectObject)	Установить свойство (**)

Синтаксис COM:

Object.get_DirectObject(&DirectObject)	Получить свойство
Object.put_DirectObject(DirectObject)	Установить свойство

Parallel – Ориентация. TRUE – параллельно направляющему объекту. FALSE – перпендикулярно направляющему объекту

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Parallel = Object.Parallel	Получить свойство (*)
Object.Parallel = Parallel	Установить свойство (*)
Parallel = Object.GetParallel()	Получить свойство (**)
Object.SetParallel(Parallel)	Установить свойство (**)

Синтаксис COM:

Object.get_Parallel(&Parallel)	Получить свойство
Object.put_Parallel(Parallel)	Установить свойство

Point – Точка

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point = Object.Point	Получить свойство (*)
Object.Point = Point	Установить свойство (*)
Point = Object.GetPoint()	Получить свойство (**)
Object.SetPoint(Point)	Установить свойство (**)

Синтаксис COM:

Object.get_Point(&Point)	Получить свойство
Object.put_Point(Point)	Установить свойство

Vector3D – Параметры вектора

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить (*)	свойство
Vector3D = Object.GetVector3D()	Получить (**)	свойство

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить (*)	свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IAxis3DByOperation

Ось через вершину по объекту.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IAxis3D
      - IAxis3DByOperation

IAxis3DByOperation – свойства

Operation – Операция

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Operation = Object.Operation                Получить свойство (* )
Object.Operation = Operation                Установить свойство (* )
Operation = Object.GetOperation()           Получить свойство (**)
Object.SetOperation( Operation )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Operation( &Operation )          Получить свойство
Object.put_Operation(  Operation )          Установить свойство
```

Интерфейс IBoolean

Интерфейс булевой операции.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IBoolean

Данный интерфейс можно получить с помощью метода коллекции булевых операций IBooleans::Add или свойства IBooleans::Boolean.

КОМПАС версия v18

IBoolean – свойства

Bodies – Массив тел

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Bodies = Object.Bodies                      Получить свойство (* )
Object.Bodies = Bodies                      Установить свойство (* )
Bodies = Object.GetBodies()                 Получить свойство (**)
Object.SetBodies( Bodies )                  Установить свойство (**)
```

Синтаксис COM:

Object.get_Bodies(&Bodies)	Получить свойство
Object.put_Bodies(Bodies)	Установить свойство

Примечание.

Позволяет получать и устанавливать массив тел, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY тел LPDISPATCH (VT_ARRAY | VT_DISPATCH).

BooleanType – Тип операции над телами

Интерфейс...

Тип данных: Значение из перечисления ksBooleanType.

Синтаксис Automation:

BooleanType = Object.BooleanType	Получить свойство (*)
Object.BooleanType = BooleanType	Установить свойство (*)
BooleanType = Object.GetBooleanType()	Получить свойство (**)
Object.SetBooleanType(BooleanType)	Установить свойство (**)

Синтаксис COM:

Object.get_BooleanType(&BooleanType)	Получить свойство
Object.put_BooleanType(BooleanType)	Установить свойство

Интерфейс ICollectionGeometry

Коллекция геометрии.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICollectionGeometry

Интерфейс можно получить у интерфейса коллекций геометрии ICollectionsGeometry с помощью свойства:
ICollectionsGeometry::CollectionGeometry или метода ICollectionsGeometry::Add.

ICollectionGeometry – свойства

Geometry – Коллекция геометрии

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Geometry = Object.Geometry                  Получить свойство (* )
Object.Geometry = Geometry                  Установить свойство (* )
Geometry = Object.GetGeometry()             Получить свойство (**)
Object.SetGeometry( Geometry )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Geometry( &Geometry )            Получить свойство
Object.put_Geometry(  Geometry )            Установить свойство
```

Интерфейс IConicSpiral3D

Коническая спираль.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISpiral3D

IConicSpiral3D

Примечание:

Интерфейс можно получить у коллекции спиралей с помощью свойства ISpirals3D::Spiral3D и метода ISpirals3D::Add

IConicSpiral3D – свойства

Diameter1 – Начальный диаметр

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Diameter1 = Object.Diameter1                Получить свойство (* )
Object.Diameter1 = Diameter1                Установить свойство (* )
Diameter1 = Object.GetDiameter1()           Получить свойство (**)
Object.SetDiameter1( Diameter1 )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Diameter1( &Diameter1 )          Получить свойство
Object.put_Diameter1( Diameter1 )           Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать начальный диаметр спирали.

Diameter2 - Конечный диаметр

Интерфейс...

Тип данных: double

Синтаксис Automation:

Diameter2 = Object.Diameter2	Получить свойство (*)
Object.Diameter2 = Diameter2	Установить свойство (*)
Diameter2 = Object.GetDiameter2()	Получить свойство (**)
Object.SetDiameter2(Diameter2)	Установить свойство (**)

Синтаксис COM:

Object.get_Diameter2(&Diameter2)	Получить свойство
Object.put_Diameter2(Diameter2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать конечный диаметр спирали.

DiameterBaseObject1 – Объект, задающий начальный диаметр

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DiameterBaseObject1 = Object.DiameterBaseObject1	Получить свойство (*)
Object.DiameterBaseObject1 = DiameterBaseObject1	Установить свойство (*)
DiameterBaseObject1 = Object.GetDiameterBaseObject1()	Получить свойство (**)
Object.SetDiameterBaseObject1(DiameterBaseObject1)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterBaseObject1(&DiameterBaseObject1)	Получить свойство
Object.put_DiameterBaseObject1(DiameterBaseObject1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать указатель на объект, задающий начальный диаметр спирали.

DiameterBaseObject2 – Объект, задающий конечный диаметр

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DiameterBaseObject2 = Object.DiameterBaseObject2	Получить свойство (*)
Object.DiameterBaseObject2 = DiameterBaseObject2	Установить свойство (*)
DiameterBaseObject2= Object.GetDiameterBaseObject2()	Получить свойство (**)
Object.SetDiameterBaseObject2(DiameterBaseObject2)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterBaseObject2(&DiameterBaseObject2)	Получить свойство
Object.put_DiameterBaseObject2(DiameterBaseObject2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать указатель на объект, задающий конечный диаметр спирали.

DiameterType1 – Способ задания начального диаметра

Интерфейс...

Тип данных: из перечисления ksSpline3DDiameterTypeEnum

Синтаксис Automation:

DiameterType1 = Object.DiameterType1	Получить свойство (*)
Object.DiameterType1 = DiameterType1	Установить свойство (*)
DiameterType1 = Object.GetDiameterType1()	Получить свойство (**)
Object.SetDiameterType1(DiameterType1)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterType1(&DiameterType1)	Получить свойство
Object.put_DiameterType1(DiameterType1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ задания диаметра.

Для начального диаметра способ "По наклону образующей" е доступен.

DiameterType2 – Способ задания конечного диаметра

Интерфейс...

Тип данных: из перечисления ksSpline3DDiameterTypeEnum

Синтаксис Automation:

DiameterType2 = Object.DiameterType2	Получить свойство (*)
Object.DiameterType2 = DiameterType2	Установить свойство (*)
DiameterType2 = Object.GetDiameterType2()	Получить свойство (**)
Object.SetDiameterType2(DiameterType2)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterType2(&DiameterType2)	Получить свойство
Object.put_DiameterType2(DiameterType2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ задания конечного диаметра спирали.

HeightCorrectionType – Признак дополнительной высоты. TRUE – продолжить за объект, FALSE – не доходя до объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HeightCorrectionType = Object.HeightCorrectionType	Получить свойство (*)
Object.HeightCorrectionType = HeightCorrectionType	Установить свойство (*)
HeightCorrectionType = Object.GetHeightCorrectionType()	Получить свойство (**)
Object.SetHeightCorrectionType(HeightCorrectionType)	Установить свойство (**)

Синтаксис COM:

Object.get_HeightCorrectionType(&HeightCorrectionType)	Получить свойство
Object.put_HeightCorrectionType(HeightCorrectionType)	Установить свойство

Версия Компас v18.1

GeneratrixTiltAngle – Угол наклона образующей

Интерфейс...

Тип данных: double

Синтаксис Automation:

GeneratrixTiltAngle = Object.GeneratrixTiltAngle	Получить свойство (*)
Object.GeneratrixTiltAngle = GeneratrixTiltAngle	Установить свойство (*)
GeneratrixTiltAngle = Object.GetGeneratrixTiltAngle()	Получить свойство (**)
Object.SetGeneratrixTiltAngle(GeneratrixTiltAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_GeneratrixTiltAngle(&GeneratrixTiltAngle)	Получить свойство
Object.put_GeneratrixTiltAngle(GeneratrixTiltAngle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать угол наклона образующей.

GeneratrixTiltAngleHow – Направление отсчета угла наклона образующей

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

GeneratrixTiltAngleHow = Object.GeneratrixTiltAngleHow	Получить свойство (*)
Object.GeneratrixTiltAngleHow = GeneratrixTiltAngleHow	Установить свойство (*)
GeneratrixTiltAngleHow = Object.GetGeneratrixTiltAngleHow()	Получить свойство (**)
Object.SetGeneratrixTiltAngleHow(GeneratrixTiltAngleHow)	Установить свойство (**)

Синтаксис COM:

Object.get_GeneratrixTiltAngleHow(&GeneratrixTiltAngleHow)	Получить свойство
Object.put_GeneratrixTiltAngleHow(GeneratrixTiltAngleHow)	Установить свойство

Значения свойства:

TRUE	- наружу,
FALSE	- внутрь (к оси спирали).

Примечание:

Свойство позволяет устанавливать и получать направление отсчета угла наклона образующей.

TurningAngle – Начальный угол (или угол поворота спирали вокруг своей оси)

Интерфейс...

Тип данных: double

Синтаксис Automation:

TurningAngle = Object.TurningAngle	Получить свойство (*)
Object.TurningAngle = TurningAngle	Установить свойство (*)
TurningAngle = Object.GetTurningAngle()	Получить свойство (**)
Object.SetTurningAngle(TurningAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_TurningAngle(&TurningAngle)	Получить свойство
Object.put_TurningAngle(TurningAngle)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать начальный угол.

Интерфейс IConnectCurve

Справка системы КОМПАС...

kompas.chm::/CM_CURVEOPER_CONNECT.htm

Интерфейс операции соединения кривых.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IConnectCurve

Интерфейс позволяет получить и изменить параметры операции соединения кривых.

Интерфейс можно получить в коллекции Операция соединения кривых IConnectCurves.

IConnectCurve – свойства

Curve1 – Кривая 1 – первая из соединяемых кривых

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve1 = Object.Curve1	Получить свойство (*)
Object.Curve1 = Curve1	Установить свойство (*)
Curve1 = Object.GetCurve1()	Получить свойство (**)
Object.SetCurve1(Curve1)	Установить свойство (**)

Синтаксис COM:

Object.get_Curve1(&Curve1)	Получить свойство
Object.put_Curve1(Curve1)	Установить свойство

Curve2 – Кривая 2 – вторая из соединяемых кривых

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve2 = Object.Curve2	Получить свойство (*)
Object.Curve2 = Curve2	Установить свойство (*)
Curve2 = Object.GetCurve2()	Получить свойство (**)
Object.SetCurve2(Curve2)	Установить свойство (**)

Синтаксис COM:

Object.get_Curve2(&Curve2)	Получить свойство
Object.put_Curve2(Curve2)	Установить свойство

Curve1ConnectType – Условие сопряжения соединяющей кривой с первой из соединяемых кривых

Интерфейс...

Тип данных: из перечисления ksConnectTypeEnum

Синтаксис Automation:

Curve1ConnectType = Object.Curve1ConnectType	=	Получить свойство (*)
Object.Curve1ConnectType = Curve1ConnectType	=	Установить свойство (*)
Curve1ConnectType = Object.GetCurve1ConnectType()	=	Получить свойство (**)
Object.SetCurve1ConnectType(Curve1ConnectType)		Установить свойство (**)

Синтаксис COM:

Object.get_Curve1ConnectType(&Curve1ConnectType)	Получить свойство
Object.put_Curve1ConnectType(Curve1ConnectType)	Установить свойство

Curve2ConnectType– Условие сопряжения соединяющей кривой со второй из соединяемых кривых

Интерфейс...

Тип данных: из перечисления ksConnectTypeEnum.

Синтаксис Automation:

Curve2ConnectType = Object.Curve2ConnectType	=	Получить свойство (*)
Object.Curve2ConnectType = Curve2ConnectType	=	Установить свойство (*)
Curve2ConnectType = Object.GetCurve2ConnectType()	=	Получить свойство (**)
Object.SetCurve2ConnectType(Curve2ConnectType)		Установить свойство (**)

Синтаксис COM:

Object.get_Curve2ConnectType(&Curve1ConnectType)	Получить свойство
Object.put_Curve2ConnectType(Curve1ConnectType)	Установить свойство

Curve1ConnectVertex – Вершина сопряжения первой из соединяемых кривых

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Curve1ConnectVertex = Object.Curve1ConnectVertex	=	Получить свойство (*)
Object.Curve1ConnectVertex = Curve1ConnectVertex	=	Установить свойство (*)
Curve1ConnectVertex = Object.GetCurve1ConnectVertex()	=	Получить свойство (**)
Object.SetCurve1ConnectVertex(Curve1ConnectVertex)		Установить свойство (**)

Синтаксис COM:

Object.get_Curve1ConnectVertex(&Curve1ConnectVertex)	Получить свойство
Object.put_Curve1ConnectVertex(Curve1ConnectVertex)	Установить свойство

Значения свойства:

TRUE	- начальная точка,
FALSE	- конечная точка.

Curve2ConnectVertex – Вершина сопряжения второй из соединяемых кривых

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Curve2ConnectVertex = Object.Curve2ConnectVertex	=	Получить свойство (*)
Object.Curve2ConnectVertex = Curve2ConnectVertex	=	Установить свойство (*)

Curve2ConnectVertex	=	Получить свойство (**)
Object.GetCurve2ConnectVertex()		
Object.SetCurve2ConnectVertex(Curve2ConnectVertex)		Установить свойство (**)

Синтаксис COM:

Object.get_Curve2ConnectVertex(&Curve2ConnectVertex)	Получить свойство
Object.put_Curve2ConnectVertex(Curve2ConnectVertex)	Установить свойство

Значения свойства:

TRUE	- начальная точка,
FALSE	- конечная точка.

Tension – Натяжение соединяющей кривой (%)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Tension = Object.Tension	Получить свойство (*)
Object.Tension = Tension	Установить свойство (*)
Tension = Object.GetTension()	Получить свойство (**)
Object.SetTension(Tension)	Установить свойство (**)

Синтаксис COM:

Object.get_Tension(&Tension)	Получить свойство
Object.put_Tension(Tension)	Установить свойство

Интерфейс IContour3D

Справка системы КОМПАС...

`kompas.chm::/604_Glava59_Kontur.htm`

Контур 3D

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IContour3D

Примечание:

Интерфейс можно получить у коллекции контуров 3D с помощью свойства IContours3D::Contour3D и метода IContours3D::Add.

IContour3D – свойства

BuildingType – Тип построения Контура 3D

Интерфейс...

Тип данных: Тип построения Контура из перечисления ksContour3DBuildingTypeTypeEnum.

Синтаксис Automation:

```
BuildingType = Object.BuildingType          Получить свойство (* )
Object.BuildingType = BuildingType          Установить свойство (* )
BuildingType = Object.GetBuildingType()     Получить свойство (**)
Object.SetBuildingType( BuildingType )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType( &BuildingType )    Получить свойство
Object.put_BuildingType(  BuildingType )    Установить свойство
```

Contour3DType – Тип контура

Интерфейс...

Тип данных: из перечисления ksContour3DTypeEnum

Синтаксис Automation:

```
Contour3DType = Object.Contour3DType          Получить свойство (* )
Object.Contour3DType = Contour3DType          Установить свойство (* )
Contour3DType = Object.GetContour3DType()     Получить свойство (**)
Object.SetContour3DType( Contour3DType )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Contour3DType( &Contour3DType )    Получить свойство
Object.put_Contour3DType(  Contour3DType )    Установить свойство
```

CutMode – Обход углов

Интерфейс...

Тип данных: из перечисления ksEquidistant3DCutModeEnum

Синтаксис Automation:

```
CutMode = Object.CutMode          Получить свойство (* )
Object.CutMode = CutMode          Установить свойство (* )
```

```
CutMode = Object.GetCutMode()          Получить свойство (**)
Object.SetCutMode( CutMode )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CutMode( &CutMode )         Получить свойство
Object.put_CutMode(  CutMode )         Установить свойство
```

Edges – Список объектов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Edges = Object.Edges                   Получить свойство (* )
Object.Edges = Edges                   Установить свойство (* )
Edges = Object.GetEdges()              Получить свойство (**)
Object.SetEdges( Edges )               Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Edges( &Edges )             Получить свойство
Object.put_Edges(  Edges )             Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать массив VT_ARRAY | VT_DISPATCH ребер, составляющих контур.

EdgesCount – Количество объектов в контуре

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
EdgesCount = Object.EdgesCount         Получить свойство (* )
EdgesCount = Object.GetEdgesCount()    Получить свойство (**)
```

Синтаксис COM:

```
Object.get_EdgesCount( &EdgesCount )   Получить свойство
```

Примечание:

Свойство доступно только для чтения.

EdgesVisibility – Видимость исходных объектов в Дереве модели

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EdgesVisibility = Object.EdgesVisibility	Получить свойство (*)
Object.EdgesVisibility = EdgesVisibility	Установить свойство (*)
EdgesVisibility = Object.GetEdgesVisibility()	Получить свойство (**)
Object.SetEdgesVisibility(EdgesVisibility)	Установить свойство (**)

Синтаксис COM:

Object.get_EdgesVisibility(&EdgesVisibility)	Получить свойство
Object.put_EdgesVisibility(EdgesVisibility)	Установить свойство

Equidistant – Получить Эквидистанту

Интерфейс...

Тип данных: Указатель на интерфейс IEquidistant3D

Синтаксис Automation:

Equidistant = Object.Equidistant	Получить свойство (*)
Equidistant = Object.GetEquidistant()	Получить свойство (**)

Синтаксис COM:

Object.get_Equidistant(&Equidistant)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PointsCount – Количество точек в контуре

Интерфейс...

Тип данных:long

Синтаксис Automation:

PointsCount = Object.PointsCount	Получить свойство (*)
PointsCount = Object.GetPointsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_PointsCount(&PointsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Radius – Радиус скругления

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство (*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство
Object.put_Radius(Radius)	Установить свойство

VarRadius – Переменный радиус скругления

Интерфейс...

Тип данных: double

Синтаксис Automation:

VarRadius = Object.VarRadius(PointNum)	Получить свойство (*)
Object.VarRadius(PointNum) = VarRadius	Установить свойство (*)
VarRadius = Object.GetVarRadius(PointNum)	Получить свойство (**)
Object.SetVarRadius(PointNum, VarRadius)	Установить свойство (**)

Синтаксис COM:

Object.get_VarRadius(PointNum, &VarRadius)	Получить свойство
Object.put_VarRadius(PointNum, VarRadius)	Установить свойство

Входной параметр:

PointNum	- Индекс точки.

IContour3D – методы

DelVarRadius – Установить цвета фона текстов с Alpha каналом

Интерфейс...

Синтаксис Automation:

BOOL DelVarRadius(long PointNum);

Синтаксис COM:

HRESULT DelVarRadius(long PointNum, BOOL * Result);

Входной параметр:

PointNum	- Индекс точки.

Возвращаемое значение:

TRUE	- в случае удачи,
FALSE	- в случае ошибки.

Интерфейс ICopyGeometry

Интерфейс копии геометрии

Иерархия:

IDispatch
  IKompasAPIObject
    IModelObject
      ICopyGeometry

Интерфейс можно получить у интерфейса коллекции копий геометрии ICopiesGeometry с помощью свойства ICopiesGeometry::CopyGeometry или метода ICopiesGeometry::Add.

ICopyGeometry – свойства

AutoUpdate – Автоматическое обновление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoUpdate = Object.AutoUpdate	Получить свойство (*)

Object.AutoUpdate = AutoUpdate	Установить свойство (*)
AutoUpdate = Object.GetAutoUpdate()	Получить свойство (**)
Object.SetAutoUpdate(AutoUpdate)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoUpdate(&AutoUpdate)	Получить свойство
Object.put_AutoUpdate(AutoUpdate)	Установить свойство

BuildingType – Способ построения операции

Интерфейс...

Тип данных: из перечисления ksCopyGeometryBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

ByCollectionGeometry – Режим копирования коллекции геометрии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ByCollectionGeometry = Object.ByCollectionGeometry	=	Получить свойство (*)
Object.ByCollectionGeometry = ByCollectionGeometry	=	Установить свойство (*)
ByCollectionGeometry = Object.GetByCollectionGeometry()	=	Получить свойство (**)

Object.SetByCollectionGeometry(ByCollectionGeometry) Установить свойство (**)

Синтаксис COM:

Object.get_ByCollectionGeometry(&ByCollectionGeometry) Получить свойство
Object.put_ByCollectionGeometry(ByCollectionGeometry) Установить свойство

CollectionGeometry – Копируемая коллекция геометрии

Интерфейс...

Тип данных: Указатель на интерфейс ICollectionGeometry

Синтаксис Automation:

CollectionGeometry = Получить свойство (*)
Object.CollectionGeometry
Object.CollectionGeometry = Установить свойство (*)
CollectionGeometry
CollectionGeometry = Получить свойство (**)
Object.GetCollectionGeometry()
Object.SetCollectionGeometry(CollectionGeometry) Установить свойство (**)

Синтаксис COM:

Object.get_CollectionGeometry(&CollectionGeometry) Получить свойство
Object.put_CollectionGeometry(CollectionGeometry) Установить свойство

ContextObjects – Работа в контексте

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ContextObjects = Получить свойство (*)
Object.ContextObjects

Object.ContextObjects ContextObjects	=	Установить свойство (*)
ContextObjects Object.GetContextObjects()	=	Получить свойство (**)
Object.SetContextObjects(ContextObjects)		Установить свойство (**)

Синтаксис COM:

Object.get_ContextObjects(&ContextObjects)	Получить свойство
Object.put_ContextObjects(ContextObjects)	Установить свойство

Значения свойства:

TRUE	- работа в контексте,
FALSE	- по внешнему документу.

DocumentFileName – Имя файла внешнего документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DocumentFileName = Object.DocumentFileName		Получить свойство (*)
DocumentFileName Object.GetDocumentFileName()	=	Получить свойство (**)

Синтаксис COM:

Object.get_DocumentFileName(&DocumentFileName)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ExternalLocalCS – Система координат внешнего документа

Интерфейс...

Тип данных: Указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

ExternalLocalCS Object.ExternalLocalCS	=	Получить свойство (*)
Object.ExternalLocalCS ExternalLocalCS	=	Установить свойство (*)
ExternalLocalCS Object.GetExternalLocalCS()	=	Получить свойство (**)
Object.SetExternalLocalCS(ExternalLocalCS)		Установить свойство (**)

Синтаксис COM:

Object.get_ExternalLocalCS(&ExternalLocalCS)	Получить свойство
Object.put_ExternalLocalCS(ExternalLocalCS)	Установить свойство

InitialObjects – Исходные объекты массива SafeArray VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

InitialObjects = Object.InitialObjects		Получить свойство (*)
Object.InitialObjects = InitialObjects		Установить свойство (*)
InitialObjects Object.GetInitialObjects()	=	Получить свойство (**)
Object.SetInitialObjects(InitialObjects)		Установить свойство (**)

Синтаксис COM:

Object.get_InitialObjects(&InitialObjects)	Получить свойство
Object.put_InitialObjects(InitialObjects)	Установить свойство

LocalCS – Система координат

Интерфейс...

Тип данных: Указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

```
LocalCS = Object.LocalCS                Получить свойство (* )
Object.LocalCS = LocalCS                Установить свойство (* )
LocalCS = Object.GetLocalCS()           Получить свойство (**)
Object.SetLocalCS( LocalCS )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LocalCS( &LocalCS )          Получить свойство
Object.put_LocalCS(  LocalCS )          Установить свойство
```

MirrorCopy – Зеркальное копирование

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
MirrorCopy = Object.MirrorCopy          Получить свойство (* )
Object.MirrorCopy = MirrorCopy          Установить свойство (* )
MirrorCopy = Object.GetMirrorCopy()     Получить свойство (**)
Object.SetMirrorCopy( MirrorCopy )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MirrorCopy(  &MirrorCopy     Получить свойство
)
Object.put_MirrorCopy(  MirrorCopy )    Установить свойство
```

WatchForSourceChange – Следить за источником

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
WatchForSourceChange          =  Получить свойство (* )
Object.WatchForSourceChange
Object.WatchForSourceChange   =  Установить свойство (* )
WatchForSourceChange
WatchForSourceChange          =  Получить свойство (**)
Object.GetWatchForSourceChange()
Object.SetWatchForSourceChange(  Установить свойство (**)
WatchForSourceChange )
```

Синтаксис COM:

Object.get_WatchForSourceChange(&WatchForSourceChange)	Получить свойство
Object.put_WatchForSourceChange(WatchForSourceChange)	Установить свойство

ICopyGeometry – методы

AddInitialObjects – Добавить объекты в массив копируемых объектов

Интерфейс...

Синтаксис Automation:

BOOL AddInitialObjects(VARIANT Objects);

Синтаксис COM:

HRESULT AddInitialObjects(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- массив SafeArray VT_ARRAY | VT_DISPATCH копируемых объектов.

AddInitialObjectsFromExternalDocument – Добавить объекты из файла в массив копируемых объектов

Интерфейс...

Синтаксис Automation:

BOOL AddInitialObjectsFromExternalDocument(IKompasDocument3D * ExternalDocument, ILocalCoordinateSystem * ExternalLocalCS, VARIANT Objects);

Синтаксис COM:

HRESULT AddInitialObjectsFromExternalDocument(IKompasDocument3D * ExternalDocument, ILocalCoordinateSystem * ExternalLocalCS, VARIANT Objects, BOOL * Result);

Входные параметры:

ExternalDocument	- указатель на внешний документ,
ExternalLocalCS	- указатель на систему координат внешнего документа,
Objects	- массив SafeArray VT_ARRAY | VT_DISPATCH копируемых объектов.

Clear – Очистить список исходных объектов массива

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Destroy – Разрушить массив

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

IsInitialObject – Проверка – является ли объект исходным для массива

Интерфейс...

Синтаксис Automation:

BOOL IsInitialObject(IModelObject * Object);

Синтаксис COM:

HRESULT IsInitialObject(IModelObject * Object, BOOL * Result);

Возвращаемое значение:

TRUE - объект является исходным,
FALSE - объект не является исходным.

Входные параметры:

Object - указатель на объект.

IsSuitableObject – Проверка пригодности объекта для операции

Интерфейс...

Синтаксис Automation:

BOOL IsSuitableObject(IModelObject * Object);

Синтаксис COM:

HRESULT IsSuitableObject(IModelObject * Object, BOOL * Result);

Возвращаемое значение:

TRUE - объект пригоден,
FALSE - объект не пригоден.

Входные параметры:

Object - указатель на объект.

OpenDocument – Открывает документ-источник на редактирование

Интерфейс...

Синтаксис Automation:

LPDISPATCH OpenDocument(BOOL Visible, BOOL ReadOnly);

Синтаксис COM:

HRESULT OpenDocument(BOOL Visible, BOOL ReadOnly, IKompasDocument3D ** Result);

Возвращаемое значение:

- указатель на интерфейс документа-источника IKompasDocument3D.

Входные параметры:

Visible	- открыть в видимом режиме,
ReadOnly	- открыть только для чтения.

Synhronise – Обновить по источнику

Интерфейс...

Синтаксис Automation:

BOOL Synhronise();

Синтаксис COM :

HRESULT Synhronise(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс ICurveByLaw

Справка системы КОМПАС...

`kompas.chm::/582_46_10_Krivay_po_zakonu.htm`

Интерфейс параметров кривой по закону

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICurveByLaw

Примечание:

Интерфейс можно получить у коллекции кривых по закону с помощью свойства ICurveByLaws::CurveByLaw и метода ICurveByLaws::Add.

ICurveByLaw – свойства

Expression – Выражение

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Expression = Object.Expression(coord)	Получить свойство (*)

Object.Expression(coord) = Expression	Установить свойство (*)
Expression = Object.GetExpression(coord)	Получить свойство (**)
Object.SetExpression(coord, Expression)	Установить свойство (**)

Синтаксис COM:

Object.get_Expression(coord, &Expression)	Получить свойство
Object.put_Expression(coord, Expression)	Установить свойство

Входные параметры:

coord - порядок законов из перечисления ksCoordLawEnum.

IntervalExpression – Выражение для интервальной переменной

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

IntervalExpression = Object.IntervalExpression(coord)	Получить свойство (*)
Object.IntervalExpression(coord) = IntervalExpression	Установить свойство (*)
IntervalExpression = Object.GetIntervalExpression(coord)	Получить свойство (**)
Object.SetIntervalExpression(coord, IntervalExpression)	Установить свойство (**)

Синтаксис COM:

Object.get_IntervalExpression(coord, &IntervalExpression)	Получить свойство
Object.put_IntervalExpression(coord, IntervalExpression)	Установить свойство

Входные параметры:

coord - порядок законов из перечисления ksCoordLawEnum.

LawType – Тип функции

Интерфейс...

Тип данных: из перечисления ksLawTypeEnum

Синтаксис Automation:

LawType = Object.LawType(coord)	Получить свойство (*)
Object.LawType(coord) = LawType	Установить свойство (*)
LawType = Object.GetLawType(coord)	Получить свойство (**)
Object.SetLawType(coord, LawType)	Установить свойство (**)

Синтаксис COM:

Object.get_LawType(coord, &LawType)	Получить свойство
Object.put_LawType(coord, LawType)	Установить свойство

Входные параметры:

coord - порядок законов из перечисления ksCoordLawEnum.

TMin – Минимальный параметр кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

TMin = Object.TMin(coord)	Получить свойство (*)
Object.TMin(coord) = TMin	Установить свойство (*)
TMin = Object.GetTMin(coord)	Получить свойство (**)
Object.SetTMin(coord, TMin)	Установить свойство (**)

Синтаксис COM:

Object.get_TMin(coord, &TMin)	Получить свойство
Object.put_TMin(coord, TMin)	Установить свойство

Входные параметры:

coord - порядок законов из перечисления ksCoordLawEnum.

TMax – Максимальный параметр кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

TMax = Object.TMax(coord)	Получить свойство (*)
Object.TMax(coord) = TMax	Установить свойство (*)
TMax = Object.GetTMax(coord)	Получить свойство (**)
Object.SetTMax(coord, TMax)	Установить свойство (**)

Синтаксис COM:

Object.get_TMax(coord, &TMax)	Получить свойство
Object.put_TMax(coord, TMax)	Установить свойство

Входные параметры:

coord - порядок законов из перечисления ksCoordLawEnum.

PointsType – Тип координат

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointsType = Object.PointsType	Получить свойство (*)
Object.PointsType = PointsType	Установить свойство (*)
PointsType = Object.GetPointsType()	Получить свойство (**)
Object.SetPointsType(PointsType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointsType(&PointsType)	Получить свойство
Object.put_PointsType(PointsType)	Установить свойство

UserVariables – Пользовательские переменные

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

UserVariables = Object.UserVariables	Получить свойство (*)
UserVariables = Object.GetUserVariables()	Получить свойство (**)

Синтаксис COM:

Object.get_UserVariables(&UserVariables)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ICurveBy2Projections

Кривая по двум проекциям.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICurveBy2Projections

Примечание:

Интерфейс можно получить у коллекции кривых по двум проекциям ICurvesBy2Projectionses с помощью свойства ICurvesBy2Projectionses::CurveBy2Projections и метода ICurvesBy2Projectionses::Add.

ICurveBy2Projections – свойства

AutoCheck – Включение/выключение автоматического выбора всех массивов ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoCheck = Object.AutoCheck	Получить свойство (*)
Object.AutoCheck = AutoCheck	Установить свойство (*)
AutoCheck = Object.GetAutoCheck()	Получить свойство (**)
Object.SetAutoCheck(AutoCheck)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoCheck(&AutoCheck)	Получить свойство
Object.put_AutoCheck(AutoCheck)	Установить свойство

Edges – Получить указанный по индексу массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges(Index)	Получить свойство (*)
Edges = Object.GetEdges(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Edges(Index, &Edges)	Получить свойство

Примечание

Свойство доступно только для чтения.

EdgesArraysCount – Количество массивов ребер в кривой пересечения поверхностей

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesArraysCount = Object.EdgesArraysCount	Получить свойство (*)

EdgesArraysCount = Object.GetEdgesArraysCount() Получить свойство (**)

Синтаксис COM:

Object.get_EdgesArraysCount(&EdgesArraysCount) Получить свойство

Примечание

Свойство доступно только для чтения.

EdgesCheck – Получить признак выбора указанного по индексу массива ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EdgesCheck = Object.EdgesCheck Получить свойство (*)
Object.EdgesCheck = EdgesCheck Установить свойство (*)
EdgesCheck = Object.GetEdgesCheck() Получить свойство (**)
Object.SetEdgesCheck(EdgesCheck) Установить свойство (**)

Синтаксис COM:

Object.get_EdgesCheck(&EdgesCheck) Получить свойство
Object.put_EdgesCheck(EdgesCheck) Установить свойство

Входные параметры:

long EdgesArrayIndex - индекс массива ребер.

EdgesChecks – Задать признаки выбора массивов ребер – массив признаков типа SAFEARRAY VARIANT_BOOL – VT_ARRAY | VT_BOOL

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

EdgesChecks = Object.EdgesChecks Получить свойство (*)
Object.EdgesChecks = EdgesChecks Установить свойство (*)
EdgesChecks = Object.GetEdgesChecks() Получить свойство (**)
Object.SetEdgesChecks(EdgesChecks) Установить свойство (**)

Синтаксис COM:

Object.get_EdgesChecks(&EdgesChecks) Получить свойство

Object.put_EdgesChecks(EdgesChecks)	Установить свойство

EdgesChecksCount – Количество доступных для создания контуров

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesChecksCount = Object.EdgesChecksCount	Получить свойство (*)
EdgesChecksCount = Object.GetEdgesChecksCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesChecksCount(&EdgesChecksCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Projection1 – Проекция 1 в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Projection1 = Object.Projection1	Получить свойство (*)
Object.Projection1 = Projection1	Установить свойство (*)
Projection1 = Object.GetProjection1()	Получить свойство (**)
Object.SetProjection1(Projection1)	Установить свойство (**)

Синтаксис COM:

Object.get_Projection1(&Projection1)	Получить свойство
Object.put_Projection1(Projection1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив ребер, составляющих проекцию 1

Projection2 – Проекция 2 в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Projection2 = Object.Projection2	Получить свойство (*)
Object.Projection2 = Projection2	Установить свойство (*)
Projection2 = Object.GetProjection2()	Получить свойство (**)
Object.SetProjection2(Projection2)	Установить свойство (**)

Синтаксис COM:

Object.get_Projection2(&Projection2)	Получить свойство
Object.put_Projection2(Projection2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив ребер, составляющих проекцию 2

Интерфейс ICylindricSpiral3D

Цилиндрическая спираль.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
ISpiral3D
ICylindricSpiral3D

Примечание:

Интерфейс можно получить у коллекции спиралей с помощью свойства ISpirals3D::Spiral3D и метода ISpirals3D::Add.

ICylindricSpiral3D – свойства

Diameter – Диаметр

Интерфейс...

Тип данных: double

Синтаксис Automation:

Diameter = Object.Diameter	Получить свойство (*)
Object.Diameter = Diameter	Установить свойство (*)
Diameter = Object.GetDiameter()	Получить свойство (**)
Object.SetDiameter(Diameter)	Установить свойство (**)

Синтаксис COM:

Object.get_Diameter(&Diameter)	Получить свойство

Object.put_Diameter(Diameter)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать диаметр спирали.

DiameterBaseObject – Объект, задающий диаметр

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

DiameterBaseObject = Object.DiameterBaseObject	Получить свойство (*)
Object.DiameterBaseObject = DiameterBaseObject	Установить свойство (*)
DiameterBaseObject = Object.GetDiameterBaseObject()	Получить свойство (**)
Object.SetDiameterBaseObject(DiameterBaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterBaseObject(&DiameterBaseObject)	Получить свойство
Object.put_DiameterBaseObject(DiameterBaseObject)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать указатель на объект, задающий диаметр спирали.

DiameterType – Способ задания диаметра

Интерфейс...

Тип данных: из перечисления ksSpline3DDiameterTypeEnum

Синтаксис Automation:

DiameterType = Object.DiameterType	Получить свойство (*)
Object.DiameterType = DiameterType	Установить свойство (*)
DiameterType = Object.GetDiameterType()	Получить свойство (**)
Object.SetDiameterType(DiameterType)	Установить свойство (**)

Синтаксис COM:

Object.get_DiameterType(&DiameterType)	Получить свойство
Object.put_DiameterType(DiameterType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ задания диаметра.

HeightCorrectionType – Признак дополнительной высоты. TRUE – продолжить за объект, FALSE – не доходя до объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HeightCorrectionType Object.HeightCorrectionType	=	Получить свойство (*)
Object.HeightCorrectionType HeightCorrectionType	=	Установить свойство (*)
HeightCorrectionType Object.GetHeightCorrectionType()	=	Получить свойство (**)
Object.SetHeightCorrectionType(HeightCorrectionType)		Установить свойство (**)

Синтаксис COM:

Object.get_HeightCorrectionType(&HeightCorrectionType)	Получить свойство
Object.put_HeightCorrectionType(HeightCorrectionType)	Установить свойство

Версия Компас v18.1

Интерфейс ICurveOutLine

Справка системы КОМПАС...

`kompas.chm::/516_43_12_Liniy_ocherka.htm`

Интерфейс параметров линии очерка.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICurveOutLine

Примечание:

Интерфейс можно получить у коллекции линий очерка с помощью свойства ICurveOutLines::CurveOutLine и метода ICurveOutLines::Add.

ICurveOutLine – свойства

AutoCheck – Включение/выключение автоматического выбора всех массивов ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoCheck = Object.AutoCheck	Получить свойство (*)
Object.AutoCheck = AutoCheck	Установить свойство (*)
AutoCheck = Object.GetAutoCheck()	Получить свойство (**)
Object.SetAutoCheck(AutoCheck)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoCheck(&AutoCheck)	Получить свойство
Object.put_AutoCheck(AutoCheck)	Установить свойство

Edges – Получить указанный по индексу массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges(Index)	Получить свойство (*)
Edges = Object.GetEdges(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Edges(Index, &Edges)	Получить свойство

Входные параметры:

Index - индекс массива ребер

Примечание:

Свойство доступно только для чтения.

EdgesArraysCount – Количество массивов ребер в линии очерка

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesArraysCount = Object.EdgesArraysCount	Получить свойство (*)
EdgesArraysCount = Object.GetEdgesArraysCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesArraysCount(&EdgesArraysCount)	Получить свойство

Примечание

Свойство доступно только для чтения.

EdgesChecks – Задать признаки выбора массивов ребер – массив признаков типа SAFEARRAY VARIANT_BOOL – VT_ARRAY | VT_BOOL

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

EdgesChecks = Object.EdgesChecks	Получить свойство (*)
Object.EdgesChecks = EdgesChecks	Установить свойство (*)
EdgesChecks = Object.GetEdgesChecks()	Получить свойство (**)
Object.SetEdgesChecks(EdgesChecks)	Установить свойство (**)

Синтаксис COM:

Object.get_EdgesChecks(&EdgesChecks)	Получить свойство
Object.put_EdgesChecks(EdgesChecks)	Установить свойство

EdgesCheck – Получить признак выбора, указанного по индексу массива ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EdgesCheck = Object.EdgesCheck(EdgesArrayIndex)	Получить свойство (*)
Object.EdgesCheck(EdgesArrayIndex) = EdgesCheck	Установить свойство (*)
EdgesCheck = Object.GetEdgesCheck(EdgesArrayIndex)	Получить свойство (**)
Object.SetEdgesCheck(EdgesArrayIndex, EdgesCheck)	Установить свойство (**)

Синтаксис COM:

Object.get_EdgesCheck(EdgesArrayIndex, &EdgesCheck)	Получить свойство
Object.put_EdgesCheck(EdgesArrayIndex, EdgesCheck)	Установить свойство

Входные параметры:

EdgesArrayIndex - индекс массива ребер.

EdgesChecksCount – Количество доступных для создания контуров

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesChecksCount Object.EdgesChecksCount	=	Получить свойство (*)
EdgesChecksCount Object.GetEdgesChecksCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesChecksCount(&EdgesChecksCount)	Получить свойство

Faces – Список граней

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces(Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces)	Получить свойство
Object.put_Faces(Faces)	Установить свойство

Vector3D – Параметры вектора

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание

Свойство доступно только для чтения.

Интерфейс IEquidistant3D

Справка системы КОМПАС...

kompas.chm::/1115_117_13_Ekvidistanta.htm

Эквидистанта 3D.

Иерархия:

IKompasAPIObject

IModelObject

IEquidistant3D

Описание:

Позволяет задавать параметры эквидистанты в 3D документе.

Примечание:

1. Интерфейс можно получить у коллекции 3D-эквидистант, используя свойство IEquidistants3D::Equidistant3D или метод IEquidistants3D::Add.
2. После задания параметров эквидистанты требуется вызвать метод IModelObject::Update.

IEquidistant3D – свойства

Angle – Угол смещения

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Свойство позволяет устанавливать и получать угол смещения.

BaseObject – Объект, задающий базовое направление смещения или поверхность, которой должны принадлежать базовая кривая

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Свойство позволяет устанавливать и получать объект, задающий базовое направление смещения.

CutMode – Тип построения

Интерфейс...

Тип данных: из перечисления ksEquidistant3DCutModeEnum

Синтаксис Automation:

CutMode = Object.CutMode	Получить свойство (*)
Object.CutMode = CutMode	Установить свойство (*)
CutMode = Object.GetCutMode()	Получить свойство (**)
Object.SetCutMode(CutMode)	Установить свойство (**)

Синтаксис COM:

Object.get_CutMode(&CutMode)	Получить свойство
Object.put_CutMode(CutMode)	Установить свойство

Свойство позволяет устанавливать и получать тип построения эквидистанты.

DirFromBegin – Смещение отсчитывать от начальной вершины пути

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DirFromBegin Object.DirFromBegin	=	Получить свойство (*)
Object.DirFromBegin DirFromBegin	=	Установить свойство (*)
DirFromBegin Object.GetDirFromBegin()	=	Получить свойство (**)
Object.SetDirFromBegin(DirFromBegin)		Установить свойство (**)

Синтаксис COM:

Object.get_DirFromBegin(&DirFromBegin)	Получить свойство
Object.put_DirFromBegin(DirFromBegin)	Установить свойство

Свойство позволяет устанавливать и получать способ отсчета смещения.

Distance – Расстояние

Интерфейс...

Тип данных: double

Синтаксис Automation:

Distance = Object.Distance	Получить свойство (*)
Object.Distance = Distance	Установить свойство (*)
Distance = Object.GetDistance()	Получить свойство (**)
Object.SetDistance(Distance)	Установить свойство (**)

Синтаксис COM:

Object.get_Distance(&Distance)	Получить свойство
Object.put_Distance(Distance)	Установить свойство

Свойство позволяет устанавливать и получать расстояние между объектом и эквидистантой.

Edges – Массив ребер

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Object.Edges = Edges	Установить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)
Object.SetEdges(Edges)	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство
Object.put_Edges(Edges)	Установить свойство

Свойство позволяет устанавливать и получать массив ребер эквидистанты.

Примечание

Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

EdgesCount – Количество объектов в эквидистанте

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesCount = Object.EdgesCount	Получить свойство (*)
EdgesCount = Object.GetEdgesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesCount(&EdgesCount)	Получить свойство

Примечание

Свойство доступно только для чтения.

KeepRadius – Сохранять радиусы скруглений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

KeepRadius = Object.KeepRadius	Получить свойство (*)
Object.KeepRadius = KeepRadius	Установить свойство (*)
KeepRadius = Object.GetKeepRadius()	Получить свойство (**)
Object.SetKeepRadius(KeepRadius)	Установить свойство (**)

Синтаксис COM:

Object.get_KeepRadius(&KeepRadius)	Получить свойство
Object.put_KeepRadius(KeepRadius)	Установить свойство

Свойство позволяет включать и отключать сохранение радиусов скруглений.

OnFace – Способ построения эквидистанта: вдоль поверхности или по направлению от вершины

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- вдоль поверхности,
FALSE	- по направлению от вершины.

Синтаксис Automation:

OnFace = Object.OnFace	Получить свойство (*)
Object.OnFace = OnFace	Установить свойство (*)
OnFace = Object.GetOnFace()	Получить свойство (**)
Object.SetOnFace(OnFace)	Установить свойство (**)

Синтаксис COM:

Object.get_OnFace(&OnFace)	Получить свойство
Object.put_OnFace(OnFace)	Установить свойство

Свойство позволяет устанавливать и получать способ построения эквидистанты.

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс вектора, задающего направление построения эквидистанты.
2. Свойство доступно только для чтения.

IEquidistant3D – методы

AddEdge – Добавить объект в эквидистанту

Интерфейс...

Синтаксис Automation:

BOOL AddEdge(IModelObject * Object);

Синтаксис COM:

HRESULT AddEdge(IModelObject * Object, BOOL * Result);

Входные параметры:

Object - указатель на добавляемый объект.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

RemoveEdge – Удалить объект из списка объектов эквидистанты

Интерфейс...

Синтаксис Automation:

BOOL RemoveEdge(long Index);

Синтаксис COM:

HRESULT RemoveEdge(long Index, BOOL * Result);

Входные параметры:

Index - индекс удаляемого объекта.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Интерфейс IFilletCurve

Справка системы КОМПАС...

kompas.chm::/CM_CURVEOPER_CORNER.htm

Интерфейс операции скругления кривых.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFilletCurve

Позволяет скруглить угол, образованный двумя кривыми. В зависимости от взаимного расположения кривых, скругление выполняется дугой окружности определенного радиуса или сплайном, лежащим на цилиндрической поверхности определенного радиуса.

В качестве исходных кривых могут использоваться:

- пространственные кривые;
- ребра тел и поверхностей;
- контуры в эскизе.

IFilletCurve – свойства

Curve1 – Кривая 1 (первая из скругляемых кривых)

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve1 = Object.Curve1	Получить свойство (*)
Object.Curve1 = Curve1	Установить свойство (*)
Curve1 = Object.GetCurve1()	Получить свойство (**)
Object.SetCurve1(Curve1)	Установить свойство (**)

Синтаксис COM:

Object.get_Curve1(&Curve1)	Получить свойство
Object.put_Curve1(Curve1)	Установить свойство

Свойство позволяет устанавливать и получать первую исходную кривую.

Curve2 – Кривая 2 (вторая из скругляемых кривых)

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
Curve2 = Object.Curve2                  Получить свойство (* )
Object.Curve2 = Curve2                  Установить свойство (* )
Curve2 = Object.GetCurve2()             Получить свойство (**)
Object.SetCurve2( Curve2 )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Curve2( &Curve2 )            Получить свойство
Object.put_Curve2(  Curve2 )            Установить свойство
```

Свойство позволяет устанавливать и получать вторую исходную кривую.

Direction – Направление дуги скругления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction            Получить свойство (* )
Object.Direction = Direction            Установить свойство (* )
Direction = Object.GetDirection()       Получить свойство (**)
Object.SetDirection( Direction )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction )      Получить свойство
Object.put_Direction(  Direction )      Установить свойство
```

Значения свойства:

```
TRUE                                    - прямое направление,
FALSE                                   - обратное направление.
```

Свойство позволяет устанавливать и получать направление дуги скругления.

OnSurface – По поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
OnSurface = Object.OnSurface            Получить свойство (* )
Object.OnSurface = OnSurface            Установить свойство (* )
OnSurface = Object.GetOnSurface()       Получить свойство (**)
Object.SetOnSurface( OnSurface )        Установить свойство (**)
```

Синтаксис COM:

Object.get_OnSurface(&OnSurface)	Получить свойство
Object.put_OnSurface(OnSurface)	Установить свойство

Radius – Радиус дуги скругления

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = Object.Radius	Получить свойство (*)
Object.Radius = Radius	Установить свойство (*)
Radius = Object.GetRadius()	Получить свойство (**)
Object.SetRadius(Radius)	Установить свойство (**)

Синтаксис COM:

Object.get_Radius(&Radius)	Получить свойство
Object.put_Radius(Radius)	Установить свойство

Свойство позволяет устанавливать и получать радиус дуги скругления.

TrimCurve1 – Усекать кривую 1

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TrimCurve1 = Object.TrimCurve1	Получить свойство (*)
Object.TrimCurve1 = TrimCurve1	Установить свойство (*)
TrimCurve1 = Object.GetTrimCurve1()	Получить свойство (**)
Object.SetTrimCurve1(TrimCurve1)	Установить свойство (**)

Синтаксис COM:

Object.get_TrimCurve1(&TrimCurve1)	Получить свойство
Object.put_TrimCurve1(TrimCurve1)	Установить свойство

Свойство позволяет устанавливать и получать признак усечения первой кривой. Если усечение включено, для исходной кривой создается копия, усеченная в точке ее касания с кривой скругления; если отключено, то создается только кривая скругления.

TrimCurve2 – Усекать кривую 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TrimCurve2 = Object.TrimCurve2	Получить свойство (*)
Object.TrimCurve2 = TrimCurve2	Установить свойство (*)
TrimCurve2 = Object.GetTrimCurve2()	Получить свойство (**)
Object.SetTrimCurve2(TrimCurve2)	Установить свойство (**)

Синтаксис COM:

Object.get_TrimCurve2(&TrimCurve2)	Получить свойство
Object.put_TrimCurve2(TrimCurve2)	Установить свойство

Свойство позволяет устанавливать и получать признак усечения второй кривой. Если усечение включено, для исходной кривой создается копия, усеченная в точке ее касания с кривой скругления; если отключено, то создается только кривая скругления.

IFilletCurve – методы

GetCurve1CutPoint – Приближенная точка начала скругления на кривой 1

Интерфейс...

Синтаксис Automation:

BOOL GetCurve1CutPoint(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetCurve1CutPoint(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры

X, Y, Z - координаты точки.

GetCurve2CutPoint – Приближенная точка начала скругления на кривой 2

Интерфейс...

Синтаксис Automation:

BOOL GetCurve2CutPoint(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetCurve2CutPoint(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X, Y, Z - координаты точки.

SetCurve1CutPoint – Приближенная точка начала скругления на кривой 1

Интерфейс...

Синтаксис Automation:

BOOL SetCurve1CutPoint(double X, double Y, double Z);

Синтаксис COM:

HRESULT SetCurve1CutPoint(double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

X, Y, Z - координаты точки.

SetCurve2CutPoint – Приближенная точка начала скругления на кривой 2

Интерфейс...

Синтаксис Automation:

BOOL SetCurve2CutPoint(double X, double Y, double Z);

Синтаксис COM:

HRESULT SetCurve2CutPoint(double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X, Y, Z - координаты точки.

Интерфейс IIsoparametricCurve

Справка системы КОМПАС...

kompas.chm::/462_Isoparametricheskie_krivye.htm#izoparam_curve

Изопараметрическая кривая.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IIsoparametricCurve

Примечание:

Интерфейс можно получить у коллекции изопараметрических кривых с помощью свойства IIsoparametricCurves::IsoparametricCurve и метода IIsoparametricCurves::Add.

IIsoparametricCurve – свойства

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject	Получить свойство (*)
Object.AssociationObject = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject()	Получить свойство (**)
Object.SetAssociationObject(AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject)	Получить свойство
Object.put_AssociationObject(AssociationObject)	Установить свойство

SurfaceObject – Установить поверхность для изопараметрической кривой

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

SurfaceObject = Object.SurfaceObject	Получить свойство (*)
Object.SurfaceObject = SurfaceObject	Установить свойство (*)
SurfaceObject = Object.GetSurfaceObject()	Получить свойство (**)
Object.SetSurfaceObject(SurfaceObject)	Установить свойство (**)

Синтаксис COM:

```
Object.get_SurfaceObject( &SurfaceObject )    Получить свойство
Object.put_SurfaceObject(  SurfaceObject )    Установить свойство
```

V – Значение параметра V в %

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
V = Object.V              Получить свойство (* )
Object.V = V              Установить свойство (* )
V = Object.GetV()         Получить свойство (**)
Object.SetV( V )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_V( &V )        Получить свойство
Object.put_V(  V )        Установить свойство
```

U – Значение параметра U в %

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
U = Object.U              Получить свойство (* )
Object.U = U              Установить свойство (* )
U = Object.GetU()         Получить свойство (**)
Object.SetU( U )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_U( &U )        Получить свойство
Object.put_U(  U )        Установить свойство
```

UDirection – Направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
UDirection = Object.UDirection    Получить свойство (* )
Object.UDirection = UDirection    Установить свойство (* )
```

UDirection	=	Получить свойство (**)
Object.GetUDirection()		
Object.SetUDirection(UDirection)		Установить свойство (**)

Синтаксис COM:

Object.get_UDirection(&UDirection)	Получить свойство
Object.put_UDirection(UDirection)	Установить свойство

Значения свойства:

TRUE	- Направление по U,
FALSE	- Направление по V.

IIsoparametricCurve – методы

GetPoint – Получить координату точки на поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X	- координата точки по X,
Y	- координата точки по Y,
Z	- координата точки по Z.

SetPoint – Установить координату точки на поверхности

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X	- координата точки по X,
Y	- координата точки по Y,
Z	- координата точки по Z.

Интерфейс IIsoparametricCurvesSet

Справка системы КОМПАС...

kompas.chm::/462_Isoparametricheskie_krivye.htm#set_izo_curves

Группа изопараметрических кривых.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IIsoparametricCurvesSet

Примечание:

Интерфейс можно получить у коллекции групп изопараметрических кривых с помощью свойства IIsoparametricCurvesSets::IsoparametricCurvesSet и метода IIsoparametricCurvesSets::Add.

IIsoparametricCurvesSet – свойства

SaveBoundaries – Оставлять границы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveBoundaries = Object.SaveBoundaries	Получить свойство (*)
Object.SaveBoundaries = SaveBoundaries	Установить свойство (*)
SaveBoundaries = Object.GetSaveBoundaries()	Получить свойство (**)
Object.SetSaveBoundaries(SaveBoundaries)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveBoundaries(&SaveBoundaries)	Получить свойство
Object.put_SaveBoundaries(SaveBoundaries)	Установить свойство

SurfaceObject – Установить поверхность для группы изопараметрических кривых

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
SurfaceObject = Object.SurfaceObject          Получить свойство (* )
Object.SurfaceObject = SurfaceObject          Установить свойство (* )
SurfaceObject = Object.GetSurfaceObject()     Получить свойство (**)
Object.SetSurfaceObject( SurfaceObject )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SurfaceObject( &SurfaceObject )    Получить свойство
Object.put_SurfaceObject(  SurfaceObject )    Установить свойство
```

UCount – Количество кривых по U

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
UCount = Object.UCount          Получить свойство (* )
Object.UCount = UCount          Установить свойство (* )
UCount                    =     Получить свойство (**)
Object.GetUCount()
Object.SetUCount( UCount )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_UCount(              Получить свойство
&UCount )
Object.put_UCount(  UCount      Установить свойство
)
```

VCount – Количество кривых по V

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
VCount = Object.VCount          Получить свойство (* )
Object.VCount = VCount          Установить свойство (* )
VCount = Object.GetVCount()     Получить свойство (**)
Object.SetVCount( VCount )      Установить свойство (**)
```

Синтаксис COM:

Object.get_VCount(&VCount)	Получить свойство
Object.put_VCount(VCount)	Установить свойство

IIsoparametricCurvesSet – методы

Destroy – Рассыпать

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс ILineSegment3D

Справка системы КОМПАС...

`kompas.chm::/454_48_3_otrezok.htm`

Отрезок 3D

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ILineSegment3D

Примечание:

Интерфейс можно получить у коллекции отрезков с помощью свойства ILineSegments3D::ILineSegment3D и метода ILineSegments3D::Add.

ILineSegment3D – свойства

Angle – Угол отрезка

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)

```
Object.Angle = Angle                    Установить свойство (* )
Angle = Object.GetAngle()               Получить свойство (**)
Object.SetAngle( Angle )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )              Получить свойство
Object.put_Angle(  Angle )              Установить свойство
```

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
AssociationObject = Object.AssociationObject( Begin )          Получить свойство (* )
Object.AssociationObject( Begin ) = AssociationObject          Установить свойство (* )
AssociationObject = Object.GetAssociationObject( Begin )       Получить свойство (**)
Object.SetAssociationObject( Begin, AssociationObject )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AssociationObject( Begin, &AssociationObject )      Получить свойство
Object.put_AssociationObject(  Begin, AssociationObject )      Установить свойство
```

Входные параметры:

BOOL Begin - начальная точка

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
BasePlane = Object.BasePlane            Получить свойство (* )
Object.BasePlane = BasePlane            Установить свойство (* )
BasePlane = Object.GetBasePlane()       Получить свойство (**)
Object.SetBasePlane( BasePlane )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BasePlane( &BasePlane )      Получить свойство
Object.put_BasePlane(  BasePlane )      Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать плоскость, в которой строится отрезок.

Синтаксис COM:

```
Object.get_Lenght( &Lenght )          Получить свойство
Object.put_Lenght(  Lenght )          Установить свойство
```

BuildingType – Тип построения отрезка

Интерфейс...

Тип данных: из перечисления ksLineSegment3DTypeEnum

Синтаксис Automation:

```
BuildingType = Object.BuildingType                  Получить свойство (* )
Object.BuildingType = BuildingType                  Установить свойство (* )
BuildingType = Object.GetBuildingType()             Получить свойство (**)
Object.SetBuildingType( BuildingType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType( &BuildingType )      Получить свойство
Object.put_BuildingType(  BuildingType )      Установить свойство
```

Lenght – Длина отрезка

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lenght = Object.Lenght                  Получить свойство (* )
Object.Lenght = Length                  Установить свойство (* )
Lenght = Object.GetLenght()             Получить свойство (**)
Object.SetLenght( Lenght )              Установить свойство (**)
```

PointParameters – Получить интерфейс параметров точки

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
PointParameters = Object.PointParameters( Begin )       Получить свойство (* )
PointParameters = Object.GetPointParameters( Begin )    Получить свойство (**)
```

Синтаксис COM:

Object.get_PointParameters(Begin, &PointParameters) Получить свойство

Примечание:

Свойство доступно только для чтения.

PointType – Тип параметров построения точки

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointType = Object.PointType(Begin) Получить свойство (*)
Object.PointType(Begin) = PointType Установить свойство (*)
PointType = Object.GetPointType(Begin) Получить свойство (**)
Object.SetPointType(Begin, PointType) Установить свойство (**)

Синтаксис COM:

Object.get_PointType(Begin, &PointType) Получить свойство
Object.put_PointType(Begin, PointType) Установить свойство

Входные параметры:

BOOL Begin - начальная точка

ILineSegment3D – методы

GetPoint – Получить координату точки

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(BOOL Begin, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(BOOL Begin, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Begin - начальная точка.

Выходные параметры:

X	- координата точки,
Y	- координата точки,
Z	- координата точки.

SetPoint – Установить координату точки

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(BOOL Begin, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(BOOL Begin, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Begin	- начальная точка,
X	- координата точки,
Y	- координата точки,
Z	- координата точки.

Интерфейс IMeshAroundPointParam

Параметры сетки для группы точек по поверхности вокруг точки.

Иерархия:

IDispatch

IMeshAroundPointParam

Примечание:

Интерфейс можно получить у интерфейса IPointsArrOnSurface с помощью метода IUnknown::QueryInterface.

IMeshAroundPointParam – свойства

AssociationObject – Ассоциированный объект

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
AssociationObject = Object.AssociationObject          Получить свойство (* )
Object.AssociationObject = AssociationObject          Установить свойство (* )
AssociationObject = Object.GetAssociationObject()     Получить свойство (**)
Object.SetAssociationObject( AssociationObject )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AssociationObject( &AssociationObject )    Получить свойство
Object.put_AssociationObject( AssociationObject )     Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать указатель на ассоциированный объект.

MeshAngle – Угол поворота сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
MeshAngle = Object.MeshAngle                          Получить свойство (* )
Object.MeshAngle = MeshAngle                          Установить свойство (* )
MeshAngle = Object.GetMeshAngle()                     Получить свойство (**)
Object.SetMeshAngle( MeshAngle )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_MeshAngle( &MeshAngle )                    Получить свойство
Object.put_MeshAngle(  MeshAngle )                    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угол поворота сетки.

MeshType – Параметры сетки для группы точек по поверхности вокруг точки

Интерфейс...

Тип данных: из перечисления ksMeshAroundPointTypeEnum

Синтаксис Automation:

```
MeshType = Object.MeshType                            Получить свойство (* )
Object.MeshType = MeshType                            Установить свойство (* )
MeshType = Object.GetMeshType()                       Получить свойство (**)
Object.SetMeshType( MeshType )                        Установить свойство (**)
```

Синтаксис COM:

Object.get_MeshType(&MeshType)	Получить свойство
Object.put_MeshType(MeshType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ построения группы точек.

OffsetU – Смещение точки по U

Интерфейс...

Тип данных: double

Синтаксис Automation:

OffsetU = Object.OffsetU	Получить свойство (*)
Object.OffsetU = OffsetU	Установить свойство (*)
OffsetU = Object.GetOffsetU()	Получить свойство (**)
Object.SetOffsetU(OffsetU)	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetU(&OffsetU)	Получить свойство
Object.put_OffsetU(OffsetU)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать смещение точки по U.

OffsetV – Смещение точки по V

Интерфейс...

Тип данных: double

Синтаксис Automation:

OffsetV = Object.OffsetV	Получить свойство (*)
Object.OffsetV = OffsetV	Установить свойство (*)
OffsetV = Object.GetOffsetV()	Получить свойство (**)
Object.SetOffsetV(OffsetV)	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetV(&OffsetV)	Получить свойство
Object.put_OffsetV(OffsetV)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать смещение точки по V.

RadialStep – Шаг радиальный

Интерфейс...

Тип данных: double

Синтаксис Automation:

RadialStep = Object.RadialStep	Получить свойство (*)
Object.RadialStep = RadialStep	Установить свойство (*)
RadialStep = Object.GetRadialStep()	Получить свойство (**)
Object.SetRadialStep(RadialStep)	Установить свойство (**)

Синтаксис COM:

Object.get_RadialStep(&RadialStep)	Получить свойство
Object.put_RadialStep(RadialStep)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать радиальный шаг.

RayCount – Количество радиальных лучей

Интерфейс...

Тип данных: double

Синтаксис Automation:

RayCount = Object.RayCount	Получить свойство (*)
Object.RayCount = RayCount	Установить свойство (*)
RayCount = Object.GetRayCount()	Получить свойство (**)
Object.SetRayCount(RayCount)	Установить свойство (**)

Синтаксис COM:

Object.get_RayCount(&RayCount)	Получить свойство
Object.put_RayCount(RayCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество радиальных лучей.

Step – Шаг между узлами

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг между узлами.

Step1 – Шаг по оси 1

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step1 = Object.Step1	Получить свойство (*)
Object.Step1 = Step1	Установить свойство (*)
Step1 = Object.GetStep1()	Получить свойство (**)
Object.SetStep1(Step1)	Установить свойство (**)

Синтаксис COM:

Object.get_Step1(&Step1)	Получить свойство
Object.put_Step1(Step1)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг по оси 1.

Step2 – Шаг по оси 2

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step2 = Object.Step2	Получить свойство (*)
Object.Step2 = Step2	Установить свойство (*)
Step2 = Object.GetStep2()	Получить свойство (**)
Object.SetStep2(Step2)	Установить свойство (**)

Синтаксис COM:

Object.get_Step2(&Step2)	Получить свойство
Object.put_Step2(Step2)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг по оси 2.

StepU – Шаг по U

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
StepU = Object.StepU                    Получить свойство (* )
Object.StepU = StepU                    Установить свойство (* )
StepU = Object.GetStepU()               Получить свойство (**)
Object.SetStepU( StepU )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_StepU( &StepU )              Получить свойство
Object.put_StepU( StepU )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать шаг по U.

StepV – Шаг по V

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
StepV = Object.StepV                    Получить свойство (* )
Object.StepV = StepV                    Установить свойство (* )
StepV = Object.GetStepV()               Получить свойство (**)
Object.SetStepV( StepV )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_StepV( &StepV )              Получить свойство
Object.put_StepV( StepV )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать шаг по V.

IMeshAroundPointParam – методы

GetOffsetPoint – Получить точку смещения

Интерфейс...

Синтаксис Automation:

BOOL GetOffsetPoint(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetOffsetPoint(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X, Y, Z	- координаты точки смещения.

Примечание:

Метод возвращает координаты точки смещения.

SetOffsetPoint – Задать смещение по точке

Интерфейс...

Синтаксис Automation:

BOOL SetOffsetPoint(double X, double Y, double Z);

Синтаксис COM:

HRESULT SetOffsetPoint(double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X, Y, Z	- координаты точки смещения.

Примечание:

Метод задает координаты точки смещения.

Интерфейс IPointsArrFromFile

Группа точек из файла.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPointsArrFromFile

Примечание:

Интерфейс можно получить у коллекции групп точек из файла с помощью свойства IPointsArrsFromFiles::PointsArrFromFile и метода IPointsArrsFromFiles::Add.

IPointsArrFromFile – свойства

FileName – Тип точек в файле

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя файла.

PointsType – Тип точек в файле

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointsType = Object.PointsType	Получить свойство (*)
Object.PointsType = PointsType	Установить свойство (*)
PointsType = Object.GetPointsType()	Получить свойство (**)
Object.SetPointsType(PointsType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointsType(&PointsType)	Получить свойство
Object.put_PointsType(PointsType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать тип точек.

Symbol – Стиль отображения

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

Symbol = Object.Symbol	Получить свойство (*)
Object.Symbol = Symbol	Установить свойство (*)
Symbol = Object.GetSymbol()	Получить свойство (**)
Object.SetSymbol(Symbol)	Установить свойство (**)

Синтаксис COM:

Object.get_Symbol(&Symbol)	Получить свойство
Object.put_Symbol(Symbol)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль отображения.

IPointsArrFromFile – методы

Destroy – Разрушить массив

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет разрушить группу точек.

Интерфейс IPointsArrOnCurve

Группа точек по кривой.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPointsArrOnCurve

Примечание:

Интерфейс можно получить у коллекции групп точек по кривой с помощью свойства IPointsArrsOnCurves::PointsArrOnCurve и метода IPointsArrsOnCurves::Add.

IPointsArrOnCurve – свойства

BuildingType – Способ построения точек группы

Интерфейс...

Тип данных: из перечисления ksPointsArrOnCurveTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ построения группы точек.

ByStep – Способ построения точек

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ByStep = Object.ByStep	Получить свойство (*)
Object.ByStep = ByStep	Установить свойство (*)
ByStep = Object.GetByStep()	Получить свойство (**)
Object.SetByStep(ByStep)	Установить свойство (**)

Синтаксис COM:

Object.get_ByStep(&ByStep)	Получить свойство
Object.put_ByStep(ByStep)	Установить свойство

Значение свойства:

TRUE	- по шагу,
FALSE	- вдоль всей кривой.

Примечание:

Свойство позволяет устанавливать и получать способ построения точек.

Count – Количество точек

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Object.Count = Count	Установить свойство (*)
Count = Object.GetCount()	Получить свойство (**)
Object.SetCount(Count)	Установить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство
Object.put_Count(Count)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество точек.

Curve – Кривая

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve = Object.Curve	Получить свойство (*)
Object.Curve = Curve	Установить свойство (*)
Curve = Object.GetCurve()	Получить свойство (**)
Object.SetCurve(Curve)	Установить свойство (**)

Синтаксис COM:

Object.get_Curve(&Curve)	Получить свойство
Object.put_Curve(Curve)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать указатель на кривую для построения группы точек.

Offset – Значение отступа%

Интерфейс...

Тип данных: double

Синтаксис Automation:

Offset = Object.Offset(FirstPoint)	Получить свойство (*)
Object.Offset(FirstPoint) = Offset	Установить свойство (*)

Offset = Object.GetOffset(FirstPoint)	Получить свойство (**)
Object.SetOffset(FirstPoint, Offset)	Установить свойство (**)

Синтаксис COM:

Object.get_Offset(FirstPoint, &Offset)	Получить свойство
Object.put_Offset(FirstPoint, Offset)	Установить свойство

Входные параметры:

BOOL FirstPoint	- для первой точки.

Примечание:

Свойство позволяет устанавливать и получать значение отступа в процентах.

OffsetDirection – Направление границы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

OffsetDirection = Object.OffsetDirection(FirstPoint)	Получить свойство (*)
Object.OffsetDirection(FirstPoint) = OffsetDirection	Установить свойство (*)
OffsetDirection = Object.GetOffsetDirection(FirstPoint)	Получить свойство (**)
Object.SetOffsetDirection(FirstPoint, OffsetDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetDirection(FirstPoint, &OffsetDirection)	Получить свойство
Object.put_OffsetDirection(FirstPoint, OffsetDirection)	Установить свойство

Входные параметры:

BOOL FirstPoint	- для первой точки.

Примечание:

Свойство позволяет устанавливать и получать направление границы.

OffsetType – Получить способ задания границ

Интерфейс...

Тип данных: из перечисления ksPoint3DCurveParamTypeEnum

Синтаксис Automation:

OffsetType = Object.OffsetType(FirstPoint)	Получить свойство (*)
Object.OffsetType(FirstPoint) = OffsetType	Установить свойство (*)
OffsetType = Object.GetOffsetType(FirstPoint)	Получить свойство (**)
Object.SetOffsetType(FirstPoint, OffsetType)	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetType(FirstPoint, &OffsetType)	Получить свойство
Object.put_OffsetType(FirstPoint, OffsetType)	Установить свойство

Входные параметры:

BOOL FirstPoint	- для первой точки.

Примечание:

Свойство позволяет устанавливать и получать способ задания границ.

OnOffsets – Задать/отменить отступы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

OnOffsets = Object.OnOffsets	Получить свойство (*)
Object.OnOffsets = OnOffsets	Установить свойство (*)
OnOffsets = Object.GetOnOffsets()	Получить свойство (**)
Object.SetOnOffsets(OnOffsets)	Установить свойство (**)

Синтаксис COM:

Object.get_OnOffsets(&OnOffsets)	Получить свойство
Object.put_OnOffsets(OnOffsets)	Установить свойство

Примечание:

Свойство позволяет задавать и отменять отступы.

ReverseDirection – Направление построения массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReverseDirection = Object.ReverseDirection	Получить свойство (*)
Object.ReverseDirection = ReverseDirection	Установить свойство (*)
ReverseDirection = Object.GetReverseDirection()	Получить свойство (**)
Object.SetReverseDirection(ReverseDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ReverseDirection(&ReverseDirection)	Получить свойство
Object.put_ReverseDirection(ReverseDirection)	Установить свойство

Значение свойства:

TRUE	- обратное направление,
FALSE	- прямое направление.

Примечание:

Свойство позволяет устанавливать и получать направление построения массива.

Step – Шаг

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг построения точек.

Symbol – Стиль отображения

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

Symbol = Object.Symbol	Получить свойство (*)
Object.Symbol = Symbol	Установить свойство (*)
Symbol = Object.GetSymbol()	Получить свойство (**)
Object.SetSymbol(Symbol)	Установить свойство (**)

Синтаксис COM:

Object.get_Symbol(&Symbol)	Получить свойство
Object.put_Symbol(Symbol)	Установить свойство

Входные параметры:

BOOL FirstPoint	- для первой точки.

Примечание:

Свойство позволяет устанавливать и получать направление границы.

IPointsArrOnCurve – методы

Destroy – Разрушить массив

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет разрушить группу точек.

Интерфейс IPointsArrOnSurface

Группа точек по поверхности.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPointsArrOnSurface

Примечание:

Интерфейс можно получить у коллекции групп точек по поверхности с помощью свойства IPointsArrsOnSurfaces::PointsArrOnSurface и метода IPointsArrsOnSurfaces::Add.

IPointsArrOnSurface – свойства

AllowBoundaries – Учитывать границы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AllowBoundaries = Object.AllowBoundaries	Получить свойство (*)
Object.AllowBoundaries = AllowBoundaries	Установить свойство (*)
AllowBoundaries = Object.GetAllowBoundaries()	Получить свойство (**)
Object.SetAllowBoundaries(AllowBoundaries)	Установить свойство (**)

Синтаксис COM:

```
Object.get_AllowBoundaries( &AllowBoundaries )      Получить свойство
Object.put_AllowBoundaries( AllowBoundaries )       Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угловое отклонение.

AngularDeflection – Угловое отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
AngularDeflection = Object.AngularDeflection        Получить свойство (* )
Object.AngularDeflection = AngularDeflection        Установить свойство (* )
AngularDeflection = Object.GetAngularDeflection()   Получить свойство (**)
Object.SetAngularDeflection( AngularDeflection )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AngularDeflection( &AngularDeflection )  Получить свойство
Object.put_AngularDeflection( AngularDeflection )   Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать угловое отклонение.

BuildingType – Способ построения точек группы

Интерфейс...

Тип данных: из перечисления ksPointsArrOnSurfaceTypeEnum

Синтаксис Automation:

```
BuildingType = Object.BuildingType                  Получить свойство (* )
Object.BuildingType = BuildingType                  Установить свойство (* )
BuildingType = Object.GetBuildingType()             Получить свойство (**)
Object.SetBuildingType( BuildingType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType( &BuildingType )            Получить свойство
Object.put_BuildingType( BuildingType )             Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать способ построения точек.

Face – Грань поверхности или тела

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Face = Object.Face	Получить свойство (*)
Object.Face = Face	Установить свойство (*)
Face = Object.GetFace()	Получить свойство (**)
Object.SetFace(Face)	Установить свойство (**)

Синтаксис COM:

Object.get_Face(&Face)	Получить свойство
Object.put_Face(Face)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать грань для построения группы точек.

LinearDeflection – Линейное отклонение

Интерфейс...

Тип данных: double

Синтаксис Automation:

LinearDeflection = Object.LinearDeflection	Получить свойство (*)
Object.LinearDeflection = LinearDeflection	Установить свойство (*)
LinearDeflection = Object.GetLinearDeflection()	Получить свойство (**)
Object.SetLinearDeflection(LinearDeflection)	Установить свойство (**)

Синтаксис COM:

Object.get_LinearDeflection(&LinearDeflection)	Получить свойство
Object.put_LinearDeflection(LinearDeflection)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать линейное отклонение.

Symbol – Стиль отображения

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

Symbol = Object.Symbol	Получить свойство (*)
Object.Symbol = Symbol	Установить свойство (*)
Symbol = Object.GetSymbol()	Получить свойство (**)

Object.SetSymbol(Symbol)	Установить свойство (**)

Синтаксис COM:

Object.get_Symbol(&Symbol)	Получить свойство
Object.put_Symbol(Symbol)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать стиль отображения.

UCount – Количество точек по U

Интерфейс...

Тип данных: long

Синтаксис Automation:

UCount = Object.UCount	Получить свойство (*)
Object.UCount = UCount	Установить свойство (*)
UCount = Object.GetUCount()	Получить свойство (**)
Object.SetUCount(UCount)	Установить свойство (**)

Синтаксис COM:

Object.get_UCount(&UCount)	Получить свойство
Object.put_UCount(UCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество точек по U.

VCount – Количество точек по V

Интерфейс...

Тип данных: long

Синтаксис Automation:

VCount = Object.VCount	Получить свойство (*)
Object.VCount = VCount	Установить свойство (*)
VCount = Object.GetVCount()	Получить свойство (**)
Object.SetVCount(VCount)	Установить свойство (**)

Синтаксис COM:

Object.get_VCount(&VCount)	Получить свойство
Object.put_VCount(VCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество точек по V.

IPointsArrOnSurface – методы

Destroy – Разрушить массив

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет разрушить группу точек.

Интерфейс IPolyLine

Справка системы КОМПАС: Команда Ломаная

`kompas.chm::/CM_CCURVE_POLYLINE_BY_VERTEX.htm`

Интерфейс пространственной ломаной.

Иерархия:

IKompasAPIObject

IModelObject

IPolyLine

ILocalCSObject

Примечание:

1. Интерфейс можно получить у коллекции пространственных ломаных, используя свойство IPolyLines::PolyLine или метод IPolyLines::Add.
2. После задания параметров объекта требуется вызвать метод IModelObject::Update.
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IPolyLine – свойства

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - пространственная ломаная замкнута,
FALSE - пространственная ломаная разомкнута.

Синтаксис Automation:

```
Closed = iObject.Closed;              Получить свойство (* )
iObject.Closed = Closed;              Установить свойство (* )
Closed = iObject.GetClosed();         Получить свойство (**)
iObject.SetClosed(Closed);            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Closed (&Closed)     Получить свойство
iObject->put_Closed (Closed)      Установить свойство
```

Примечание.

Свойство позволяет получить/установить разомкнутую или замкнутую ломаную.

Edges – Массив ребер 3D ломаной

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Edges = iObject.Edges;                Получить свойство (* )
iObject.Edges = Edges;                Установить свойство (* )
Edges = iObject.GetEdges();           Получить свойство (**)
iObject.SetEdges( Edges );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Edges( &Edges );     Получить свойство
iObject->put_Edges( Edges );      Установить свойство
```

Примечание.

Свойство позволяет получать массив ребер 3D ломаной. Массив возвращается в виде массива SAFEARRAY рёбер LPDISPATCH (VT_ARRAY | VT_DISPATCH).

VertexCount – Количество вершин

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
VertexCount =                Получить свойство (* )
iObject.VertexCount;
VertexCount =                Получить свойство (**)
iObject.GetVertexCount( );
```

Синтаксис COM:

```
iObject->get_VertexCount(    Получить свойство
&VertexCount )
```

Примечание.

Свойство доступно только для чтения.

VertexParams – Параметры вершины ломаной

Интерфейс...

Тип данных: указатель на интерфейс параметров вершины ломанной ICurveVertexParam

Синтаксис Automation:

```
VertexParams = iObject.VertexParams( Index );          Получить свойство (* )
VertexParams = iObject.GetVertexParams( Index );       Получить свойство (**)
```

Синтаксис COM:

```
iObject-                     Получить свойство
>get_VertexParams(
Index, &VertexParams )
```

Входные параметры:

```
Index                        - индекс вершины.
```

Примечание.

Свойство доступно только для чтения.

VertexParamsArray – Массив параметров вершин в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Значения свойства:

```
Массив SafeArray типа VT_ARRAY |
VT_DISPATCH.
```

Синтаксис Automation:

VertexParamsArray = iObject.VertexParamsArray;	Получить свойство (*)
VertexParamsArray = iObject.GetVertexParamsArray();	Получить свойство (**)

Синтаксис COM:

iObject->get_VertexParamsArray(&VertexParamsArray)	Получить свойство

Примечание.

1. Свойство доступно только для чтения.
2. Свойство позволяет получить массив SAFEARRAY параметров всех вершин. Массив VT_DISPATCH, которые можно преобразовать в интерфейсы ICurveVertexParam.

VertexVisible – Видимость свободных вершин 3D ломаной

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- вершины видимые,
FALSE	- вершины невидимые.

Синтаксис Automation:

VertexVisible = iObject.VertexVisible;	Получить свойство (*)
iObject.VertexVisible = VertexVisible;	Установить свойство (*)
VertexVisible = iObject.GetVertexVisible();	Получить свойство (**)
iObject.SetVertexVisible(VertexVisible);	Установить свойство (**)

Синтаксис COM:

iObject->get_VertexVisible (&VertexVisible)	Получить свойство
iObject->put_VertexVisible (VertexVisible)	Установить свойство

Примечание.

Свойство позволяет получить/установить видимость свободных вершин 3D ломаной.

IPolyLine – методы

AddVertex – Создать новую вершину

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddVertex(long Index);

Синтаксис COM:

HRESULT AddVertex([in] long Index,
[out, retval] ICurveVertexParam ** Result);

Возвращаемое значение:

ICurveVertexParam	- указатель на интерфейс параметров вершины ломаной.

DeleteVertex – Удалить вершину с указанным индексом

Интерфейс...

Синтаксис Automation:

BOOL DeleteVertex(long Index);

Синтаксис COM:

HRESULT DeleteVertex([in] long Index,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

Index	- индекс вершины.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

ErrorIndexes – Массив не скругленных вершин SAFEARRAY – VT_ARRAY | VT_I4

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ErrorIndexes = Object.ErrorIndexes;	Получить свойство (*)
ErrorIndexes = Object.GetErrorIndexes();	Получить свойство (**)

Синтаксис COM:

Object.get_ErrorIndexes(&ErrorIndexes) Получить свойство

Примечание.

Свойство доступно только для чтения.

Flush – Очистить массив вершин

Интерфейс...

Синтаксис Automation:

BOOL Flush();

Синтаксис COM:

HRESULT Flush([out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetVertexCoordinatesArray – Получить массив координат и радиусов вершин в виде SAFEARRAY double – VT_ARRAY | VT_R8

Интерфейс...

Синтаксис Automation:

BOOL GetVertexCoordinatesArray(VARIANT * Coordinates,
VARIANT * Radiuses);

Синтаксис COM:

HRESULT GetVertexCoordinatesArray([out] VARIANT * Coordinates,
[out] VARIANT * Radiuses,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

Coordinates	- массив SafeArray типа VT_ARRAY | VT_R8 координат точек всех вершин,
Radiuses	- массив SafeArray типа VT_ARRAY | VT_R8 радиусов при вершинах кривой.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

1. Параметры Coordinates и Radiuses не являются обязательными. В функцию достаточно передать один из указателей на VARIANT.
2. Массивы являются согласованными:

Координаты точек в массиве Coordinates лежат в следующей последовательности:

- x0, y0, z0, x1, y1, z1 ...xi, yi, zi,

Радиусы при вершинах кривой в массиве Radiuses лежат в последовательности:

- radius0, radius1, ...radiusi.

ReadFromFile – Прочитать параметры вершин из текстового файла

Интерфейс...

Синтаксис Automation:

BOOL ReadFromFile(BSTR fileName);

Синтаксис COM:

HRESULT ReadFromFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

fileName	- имя файла.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

См. также IPolyLine::WriteToFile.

WriteToFile – Записать параметры вершин в текстовый файл

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(BSTR fileName);

Синтаксис COM:

HRESULT WriteToFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

fileName	- имя файла.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

См. также IPolyLine::ReadFromFile.

Интерфейс IProjectionCurve

Проекционная кривая.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IProjectionCurve

Примечание:

Интерфейс можно получить у коллекции проекционных кривых IProjectionCurves с помощью свойства IProjectionCurves::ProjectionCurve и метода IProjectionCurves::Add.

IProjectionCurve – свойства

AutoCheck – Включение/выключение автоматического выбора всех массивов ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoCheck= Object.AutoCheck	Получить свойство (*)
Object.AutoCheck= AutoCheck	Установить свойство (*)
AutoCheck= Object.GetAutoCheck()	Получить свойство (**)
Object.SetAutoCheck(AutoCheck)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoCheck(&AutoCheck)	Получить свойство
Object.put_AutoCheck(AutoCheck)	Установить свойство

Curves – Проекция 2 в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Curves = Object.Curves	Получить свойство (*)
Object.Curves = Curves	Установить свойство (*)
Curves = Object.GetCurves()	Получить свойство (**)
Object.SetCurves(Curves)	Установить свойство (**)

Синтаксис COM:

Object.get_Curves(&Curves)	Получить свойство
Object.put_Curves(Curves)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив кривых, составляющих базовую кривую.

Edges – Получить указанный по индексу массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges;	Получить свойство (*)
Edges = Object.GetEdges();	Получить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство

Входные параметры:

long Index	- индекс массива ребер.

Примечание.

Свойство доступно только для чтения.

EdgesArraysCount – Количество массивов ребер в кривой пересечения поверхностей

Интерфейс...

Тип данных: long.

Синтаксис Automation:

EdgesArraysCount = Получить свойство (*)
Object.EdgesArraysCount;
EdgesArraysCount = Получить свойство (**)
Object.GetEdgesArraysCount();

Синтаксис COM:

Object.get_EdgesArraysCount(&EdgesArraysCount) — Получить свойство

Примечание.

Свойство доступно только для чтения.

EdgesCheck – Получить признак выбора указанного по индексу массива ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EdgesChecks= Object.EdgesChecks — Получить свойство (*)
Object.EdgesChecks= EdgesChecks — Установить свойство (*)
EdgesChecks= Object.GetEdgesChecks() — Получить свойство (**)
Object.SetEdgesChecks(EdgesChecks) — Установить свойство (**)

Синтаксис COM:

Object.get_EdgesChecks(&EdgesChecks) — Получить свойство
Object.put_EdgesChecks(EdgesChecks) — Установить свойство

Входные параметры:

long EdgesArrayIndex — - индекс массива ребер.

Примечание.

Свойство доступно только для чтения.

EdgesChecks – Задать признаки выбора массивов ребер – массив признаков типа SAFEARRAY VARIANT_BOOL – VT_ARRAY | VT_BOOL

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

EdgesChecks= Object.EdgesChecks	Получить свойство (*)
Object.EdgesChecks= EdgesChecks	Установить свойство (*)
EdgesChecks= Object.GetEdgesChecks()	Получить свойство (**)
Object.SetEdgesChecks(EdgesChecks)	Установить свойство (**)

Синтаксис COM:

Object.get_EdgesChecks(&EdgesChecks)	Получить свойство
Object.put_EdgesChecks(EdgesChecks)	Установить свойство

EdgesChecksCount – Количество доступных для создания контуров

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesChecksCount = Object.EdgesChecksCount	Получить свойство (*)
EdgesChecksCount = Object.GetEdgesChecksCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesChecksCount(&EdgesChecksCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Faces – Проекция 1 в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces(Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces)	Получить свойство
Object.put_Faces(Faces)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив граней, составляющих поверхность проецирования.

ProjectionType – Тип проекции

Интерфейс...

Тип данных: из перечисления ksCurveProjectionTypeEnum

Синтаксис Automation:

ProjectionType= Object.ProjectionType	Получить свойство (*)
Object.ProjectionType= ProjectionType	Установить свойство (*)
ProjectionType= Object.GetProjectionType()	Получить свойство (**)
Object.SetProjectionType(ProjectionType)	Установить свойство (**)

Синтаксис COM:

Object.get_ProjectionType(&ProjectionType)	Получить свойство
Object.put_ProjectionType(ProjectionType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать массив кривых, составляющих базовую кривую.

TruncationByBounds – Усечение по границам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TruncationByBounds= Object.TruncationByBounds	Получить свойство (*)
Object.TruncationByBounds= TruncationByBounds	Установить свойство (*)
TruncationByBounds= Object.GetTruncationByBounds()	Получить свойство (**)
Object.SetTruncationByBounds(TruncationByBounds)	Установить свойство (**)

Синтаксис COM:

Object.get_TruncationByBounds(&TruncationByBounds)	Получить свойство
Object.put_TruncationByBounds(TruncationByBounds)	Установить свойство

Vector3D – Параметры вектора

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D;	Получить свойство (*)
Vector3D = Object.GetVector3D();	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание.

Свойство доступно только для чтения.

Интерфейс ISpiral3D

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISpiral3D

Примечание:

Интерфейс можно получить у коллекции спиралей с помощью свойства ISpirals3D::Spiral3D и метода ISpirals3D::Add, а также у интерфейсов IConicSpiral3D и ICylindricSpiral3D с помощью метода IUnknown::QueryInterface.

ISpiral3D – свойства

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
BasePlane = Object.BasePlane                    Получить свойство (* )
Object.BasePlane = BasePlane                    Установить свойство (* )
BasePlane = Object.GetBasePlane()               Получить свойство (**)
Object.SetBasePlane( BasePlane )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BasePlane( &BasePlane )              Получить свойство
Object.put_BasePlane( BasePlane )               Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать базовую плоскость или плоскую грань, на которой располагается спираль.

BuildingDirection – Направление построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
BuildingDirection = Object.BuildingDirection                Получить свойство (* )
Object.BuildingDirection = BuildingDirection                Установить свойство (* )
BuildingDirection = Object.GetBuildingDirection()           Получить свойство (**)
Object.SetBuildingDirection( BuildingDirection )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingDirection( &BuildingDirection            Получить свойство
)
Object.put_BuildingDirection( BuildingDirection )           Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать направление построения спирали.

BuildingType – Способ построения

Интерфейс...

Тип данных: из перечисления ksSpline3DBuildingTypeEnum

Синтаксис Automation:

```
BuildingType = Object.BuildingType              Получить свойство (* )
Object.BuildingType = BuildingType              Установить свойство (* )
BuildingType = Object.GetBuildingType()         Получить свойство (**)
Object.SetBuildingType( BuildingType )          Установить свойство (**)
```

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ построения спирали.

Heigh – Высота

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height	Получить свойство (*)
Object.Height = Height	Установить свойство (*)
Height = Object.GetHeight()	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height)	Получить свойство
Object.put_Height(Height)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать высоту спирали.

HeightBaseObject – Объект, задающий высоту

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

HeightBaseObject = Object.HeightBaseObject	Получить свойство (*)
Object.HeightBaseObject = HeightBaseObject	Установить свойство (*)
HeightBaseObject = Object.GetHeightBaseObject()	Получить свойство (**)
Object.SetHeightBaseObject(HeightBaseObject)	Установить свойство (**)

Синтаксис COM:

bject.get_HeightBaseObject(&HeightBaseObject)	Получить свойство
Object.put_HeightBaseObject(HeightBaseObject)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать указатель на объект, задающий высоту спирали.

HeightCorrection – Коррекция высоты, заданной по объекту

Интерфейс...

Тип данных: double

Синтаксис Automation:

HeightCorrection = Object.HeightCorrection	Получить свойство (*)
Object.HeightCorrection = HeightCorrection	Установить свойство (*)
HeightCorrection = Object.GetHeightCorrection()	Получить свойство (**)
Object.SetHeightCorrection(HeightCorrection)	Установить свойство (**)

Синтаксис COM:

Object.get_HeightCorrection(&HeightCorrection)	Получить свойство
Object.put_HeightCorrection(HeightCorrection)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать коррекцию высоты, заданной по объекту.

HeightType – Способ задания высоты

Интерфейс...

Тип данных: из перечисления ksSpiral3DHeightTypeEnum

Синтаксис Automation:

HeightType = Object.HeightType	Получить свойство (*)
Object.HeightType = HeightType	Установить свойство (*)
HeightType = Object.GetHeightType()	Получить свойство (**)
Object.SetHeightType(HeightType)	Установить свойство (**)

Синтаксис COM:

Object.get_HeightType(&HeightType)	Получить свойство
Object.put_HeightType(HeightType)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать способ задания высоты спирали.

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс ISketch

Синтаксис Automation:

Sketch = Object.Sketch	Получить свойство (*)

Sketch = Object.GetSketch()	Получить свойство (**)

Синтаксис COM:

Object.get_Sketch(&Sketch)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на эскиз спирали.

Step – Шаг

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать шаг витков спирали.

TurningAngle – Начальный угол (или угол поворота спирали вокруг своей оси)

Интерфейс...

Тип данных: double

Синтаксис Automation:

TurningAngle = Object.TurningAngle	Получить свойство (*)
Object.TurningAngle = TurningAngle	Установить свойство (*)
TurningAngle = Object.GetTurningAngle()	Получить свойство (**)
Object.SetTurningAngle(TurningAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_TurningAngle(&TurningAngle)	Получить свойство

Object.put_TurningAngle(TurningAngle)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать угол поворота спирали вокруг перпендикуляра к опорной грани.
2. Угол измеряется относительно оси абсцисс системы координат опорной грани.

TurnsCount – Количество витков

Интерфейс...

Тип данных: double

Синтаксис Automation:

TurnsCount = Object.TurnsCount	Получить свойство (*)
Object.TurnsCount = TurnsCount	Установить свойство (*)
TurnsCount = Object.GetTurnsCount()	Получить свойство (**)
Object.SetTurnsCount(TurnsCount)	Установить свойство (**)

Синтаксис COM:

Object.get_TurnsCount(&TurnsCount)	Получить свойство
Object.put_TurnsCount(TurnsCount)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество витков спирали.

TurnDirection – Направление навивки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TurnDirection = Object.TurnDirection	Получить свойство (*)
Object.TurnDirection = TurnDirection	Установить свойство (*)
TurnDirection = Object.GetTurnDirection()	Получить свойство (**)
Object.SetTurnDirection(TurnDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_TurnDirection(&TurnDirection)	Получить свойство
Object.put_TurnDirection(TurnDirection)	Установить свойство

Значение свойства:

TRUE	- прямое,
FALSE	- обратное.

Примечание:

Свойство позволяет устанавливать и получать направление навивки.

ISpiral3D – методы

GetBasePoint – Получить координаты точки привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Интерфейс...

Синтаксис Automation:

BOOL GetBasePoint(double * X, double * Y);

Синтаксис COM:

HRESULT GetBasePoint(double * X, double * Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X, Y - координаты точки привязки спирали.

Примечание:

Метод возвращает координаты точки привязки спирали.

SetBasePoint – Установить координаты точки привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Интерфейс...

Синтаксис Automation:

BOOL SetBasePoint(double X, double Y);

Синтаксис COM:

HRESULT SetBasePoint(double X, double Y, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X, Y - координаты точки привязки спирали.

Примечание:

Метод устанавливает координаты точки привязки спирали.

Интерфейс ISpline3D

Справка системы КОМПАС:

kompas.chm::/1091_116_6_Splajn.htm

Интерфейс сплайна.

Иерархия:

IKompasAPIObject

IModelObject

ISpline3D

ILocalCSObject

Примечание:

1. Интерфейс можно получить у коллекции сплайнов, используя свойство ISplines3D::Spline3D или метод ISplines3D::Add.
2. После задания параметров сплайна требуется вызвать метод IModelObject::Update.
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

ISpline3D – свойства

CenterPointAssociationObject – Точка привязки центра кривизны

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

CenterPointAssociationObject	=	Получить свойство (*)
Object.CenterPointAssociationObject(PointIndex);		
Object.CenterPointAssociationObject(PointIndex)	=	Установить свойство (*)
CenterPointAssociationObject;		
CenterPointAssociationObject	=	Получить свойство (**)
Object.GetCenterPointAssociationObject(PointIndex);		
Object.SetCenterPointAssociationObject(PointIndex,		Установить свойство (**)
CenterPointAssociationObject)		

Синтаксис COM:

Object.get_CenterPointAssociationObject(	Получить свойство
PointIndex, &CenterPointAssociationObject)	
Object.put_CenterPointAssociationObject(	Установить свойство
PointIndex, CenterPointAssociationObject)	

Входные параметры:

long PointIndex - индекс точки.

CenterPointParams – Получение параметров точки центра кривизны

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject.

Синтаксис Automation:

CenterPointParams = Получить свойство (*)
Object.CenterPointParams(PointIndex);
CenterPointParams = Получить свойство (**)
Object.GetCenterPointParams(PointIndex);

Синтаксис COM:

Object.get_CenterPointParams(PointIndex, Получить свойство
&CenterPointParams)

Входные параметры:

long PointIndex - индекс точки.

Примечание:

1. Свойство доступно только для чтения.
2. В зависимости от типа ISpline3D::CenterPointType, интерфейс параметров должен приводиться к одному из следующих вариантов:

- ksPDisplace - IPoint3DParamDisplace (интерфейс параметров пространственной точки, заданной по смещению от опорного объекта),
- ksPIntersect - IPoint3DParamIntersect (интерфейс параметров пространственной точки, заданной на пересечении опорных объектов),
- ksPCenter - IPoint3DParamCenter (интерфейс параметров пространственной точки, заданной в центре опорного объекта),
- ksPCurve - IPoint3DParamCurve (интерфейс параметров пространственной точки, заданной на кривой со смещением),
- ksPSurface - IPoint3DparamSurface (интерфейс параметров пространственной точки, заданной на поверхности),
- ksPProjection - IPoint3DParamProjection (интерфейс параметров пространственной точки, заданной проецированием).

CenterPointType – Установить способ построения точки

Интерфейс...

Тип данных: Значение из перечисления ksPoint3DTypeEnum.

Синтаксис Automation:

```
CenterPointType = Object.CenterPointType( PointIndex ); Получить свойство (* )
Object.CenterPointType( PointIndex ) = CenterPointType; Установить свойство (* )
CenterPointType = Object.GetCenterPointType( PointIndex ); Получить свойство (**)
Object.SetCenterPointType( PointIndex, CenterPointType ) Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CenterPointType( PointIndex, &CenterPointType ) Получить свойство
Object.put_CenterPointType( PointIndex, CenterPointType ) Установить свойство
```

Входные параметры:

```
long PointIndex - индекс точки.
```

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE - сплайн замкнутый,
FALSE - сплайн разомкнутый.
```

Синтаксис Automation:

```
Closed = iObject.Closed; Получить свойство (* )
iObject.Closed = Closed; Установить свойство (* )
Closed = iObject.GetClosed(); Получить свойство (**)
iObject.SetClosed(Closed); Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Closed (&Closed) Получить свойство
iObject->put_Closed (Closed) Установить свойство
```

Примечание.

Свойство позволяет получить/установить разомкнутый или замкнутый сплайн.

ConstraintLimitObject – Граница

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

ConstraintLimitObject = Object.ConstraintLimitObject(PointIndex);	=	Получить свойство (*)
Object.ConstraintLimitObject(PointIndex) = ConstraintLimitObject;	=	Установить свойство (*)
ConstraintLimitObject = Object.GetConstraintLimitObject(PointIndex);	=	Получить свойство (**)
Object.SetConstraintLimitObject(PointIndex, ConstraintLimitObject);		Установить свойство (**)

Синтаксис COM:

Object.get_ConstraintLimitObject(PointIndex, &ConstraintLimitObject)	Получить свойство
Object.put_ConstraintLimitObject(PointIndex, ConstraintLimitObject)	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

ConstraintObject – Объект сопряжения

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

ConstraintObject = Object.ConstraintObject(PointIndex);		Получить свойство (*)
Object.ConstraintObject(PointIndex) = ConstraintObject;	=	Установить свойство (*)
ConstraintObject = Object.GetConstraintObject(PointIndex);		Получить свойство (**)

Object.SetConstraintObject(ConstraintObject);	PointIndex,	Установить свойство (**)

Синтаксис COM:

Object.get_ConstraintObject(&ConstraintObject)	PointIndex,	Получить свойство
Object.put_ConstraintObject(ConstraintObject)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

ConstraintReverse – Противоположное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ConstraintReverse = Object.ConstraintReverse(PointIndex);		Получить свойство (*)
Object.ConstraintReverse(PointIndex) = ConstraintReverse;		Установить свойство (*)
ConstraintReverse = Object.GetConstraintReverse(PointIndex);		Получить свойство (**)
Object.SetConstraintReverse(ConstraintReverse);	PointIndex,	Установить свойство (**)

Синтаксис COM:

Object.get_ConstraintReverse(&ConstraintReverse)	PointIndex,	Получить свойство
Object.put_ConstraintReverse(ConstraintReverse)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

ConstraintType – Тип сопряжения в заданной вершине. Управление кривизной

Интерфейс...

Тип данных: Значение из перечисления ksNurbsByPointsPointConstraintsEnum.

Синтаксис Automation:

ConstraintType = Object.ConstraintType(PointIndex);	Получить свойство (*)
Object.ConstraintType(PointIndex) = ConstraintType;	Установить свойство (*)
ConstraintType = Object.GetConstraintType(PointIndex);	Получить свойство (**)
Object.SetConstraintType(PointIndex, ConstraintType);	Установить свойство (**)

Синтаксис COM:

Object.get_ConstraintType(PointIndex, &ConstraintType)	Получить свойство
Object.put_ConstraintType(PointIndex, ConstraintType)	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

Curvature – Величина кривизны в указанной точке

Интерфейс...

Тип данных: double

Синтаксис Automation:

Curvature = Object.Curvature(PointIndex);	Получить свойство (*)
Object.Curvature(PointIndex) = Curvature;	Установить свойство (*)
Curvature = Object.GetCurvature(PointIndex);	Получить свойство (**)
Object.SetCurvature(PointIndex, Curvature);	Установить свойство (**)

Синтаксис COM:

Object.get_Curvature(PointIndex, &Curvature)	Получить свойство
Object.put_Curvature(PointIndex, Curvature)	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

CurvatureRadius – Радиус кривизны

Интерфейс...

Тип данных: double

Синтаксис Automation:

CurvatureRadius = Object.CurvatureRadius(PointIndex);	Получить свойство (*)
Object.CurvatureRadius(PointIndex) = CurvatureRadius;	Установить свойство (*)
CurvatureRadius = Object.GetCurvatureRadius(PointIndex);	Получить свойство (**)
Object.SetCurvatureRadius(PointIndex, CurvatureRadius);	Установить свойство (**)

Синтаксис COM:

Object.get_CurvatureRadius(&CurvatureRadius)	PointIndex,	Получить свойство
Object.put_CurvatureRadius(CurvatureRadius)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

DirectionVector – Направляющий вектор в точке сопряжения

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D.

Синтаксис Automation:

DirectionVector = Object.DirectionVector(PointIndex, VectorIndex, PVal);	Получить свойство (*)
DirectionVector = Object.GetDirectionVector(PointIndex, VectorIndex, PVal);	Получить свойство (**)

Синтаксис COM:

Object.get_DirectionVector(VectorIndex, PVal, &DirectionVector)	PointIndex,	Получить свойство

Входные параметры:

long PointIndex	- индекс точки.
VectorIndex	- способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.

Примечание:

Свойство доступно только для чтения.

Knots – Массив SAFEARRAY узлов кривой NURBS

Интерфейс...

Синтаксис Automation:

Knots = Object.Knots	Получить свойство (*)
Object.Knots = Knots	Установить свойство (*)
Knots = Object.GetKnots()	Получить свойство (**)
Object.SetKnots(Knots)	Установить свойство (**)

Синтаксис COM:

Object.get_Knots(&Knots)	Получить свойство
Object.put_Knots(Knots)	Установить свойство

Свойство позволяет получить и изменить узлы кривой NURBS.

Reverse – Противоположное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Reverse = Object.Reverse(Index)	Получить свойство (*)
Object.Reverse(Index) = Reverse	Установить свойство (*)
Reverse = Object.GetReverse(Index)	Получить свойство (**)
Object.SetReverse(Index, Reverse))	Установить свойство (**)

Синтаксис COM:

Object.get_Reverse(Index, &Reverse)	Получить свойство
Object.put_Reverse(Index, Reverse)	Установить свойство

Входные параметры:

Index	- индекс касательного вектора.

SplineOnPoles – Тип построения

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- сплайн по полюсам,
FALSE	- сплайн по точкам.

Синтаксис Automation:

SplineOnPoles = iObject.SplineOnPoles;	Получить свойство (*)
iObject.SplineOnPoles = SplineOnPoles;	Установить свойство (*)
SplineOnPoles = iObject.GetSplineOnPoles();	Получить свойство (**)
iObject.SetSplineOnPoles(SplineOnPoles);	Установить свойство (**)

Синтаксис COM:

iObject->get_SplineOnPoles (&SplineOnPoles)	Получить свойство
iObject->put_SplineOnPoles (SplineOnPoles)	Установить свойство

Примечание.

Свойство позволяет получить/установить сплайн по точкам/полюсам.

SplineOrder – Порядок сплайна

Интерфейс...

Тип данных: long

Значения свойства:

TRUE	- сплайн по полюсам,
FALSE	- сплайн по точкам.

Синтаксис Automation:

SplineOrder = iObject.SplineOrder;	Получить свойство (*)
iObject.SplineOrder = SplineOrder;	Установить свойство (*)

SplineOrder = iObject.GetSplineOrder();	Получить свойство (**)
iObject.SetSplineOrder(SplineOrder);	Установить свойство (**)

Синтаксис COM:

iObject->get_SplineOrder (&SplineOrder)	Получить свойство
iObject->put_SplineOrder (SplineOrder)	Установить свойство

Примечание.

Свойство позволяет получить/установить порядок сплайна. Значение порядка кривой должно принадлежать диапазону 3 ... 10.

SplineTransitionType – Способ создания сопряжения в заданной вершине

Интерфейс...

Тип данных: Значение из перечисления ksSplineTransitionTypeEnum.

Синтаксис Automation:

SplineTransitionType = Object.SplineTransitionType(PointIndex);	Получить свойство (*)
Object.SplineTransitionType(PointIndex) = SplineTransitionType;	Установить свойство (*)
SplineTransitionType = Object.GetSplineTransitionType(PointIndex);	Получить свойство (**)
Object.SetSplineTransitionType(PointIndex, SplineTransitionType);	Установить свойство (**)

Синтаксис COM:

Object.get_SplineTransitionType(&SplineTransitionType)	PointIndex,	Получить свойство
Object.put_SplineTransitionType(SplineTransitionType)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

TangentVectorLenght – Длина касательного вектора

Интерфейс...

Тип данных: double

Синтаксис Automation:

TangentVectorLenght = Object.TangentVectorLenght(PointIndex);		Получить свойство (*)
Object.TangentVectorLenght(PointIndex) TangentVectorLenght;	=	Установить свойство (*)
TangentVectorLenght Object.GetTangentVectorLenght(PointIndex);	=	Получить свойство (**)
Object.SetTangentVectorLenght(PointIndex, TangentVectorLenght);		Установить свойство (**)

Синтаксис COM:

Object.get_TangentVectorLenght(&TangentVectorLenght)	PointIndex,	Получить свойство
Object.put_TangentVectorLenght(TangentVectorLenght)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.

VertexCount – Количество вершин

Интерфейс...

Тип данных: long

Синтаксис Automation:

VertexCount iObject.VertexCount;	=	Получить свойство (*)
VertexCount iObject.GetVertexCount( );	=	Получить свойство (**)

Синтаксис COM:

iObject->get_VertexCount(&VertexCount)	Получить свойство

Примечание.

Свойство доступно только для чтения.

VertexParams – Получение параметров вершины ломаной

Интерфейс...

Тип данных: Указатель на интерфейс параметров вершины ломаной ICurveVertexParam

Синтаксис Automation:

VertexParams = iObject.VertexParams(Index);	Получить свойство (*)
VertexParams = iObject.GetVertexParams(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_VertexParams(Index, &VertexParams)	Получить свойство

Входные параметры:

Index	- индекс вершины.

Примечание.

Свойство доступно только для чтения.

VertexParamsArray – Массив параметров вершин в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Значения свойства:

Массив SafeArray типа VT_ARRAY | VT_DISPATCH.

Синтаксис Automation:

VertexParamsArray = iObject.VertexParamsArray;	Получить свойство (*)
VertexParamsArray = iObject.GetVertexParamsArray();	Получить свойство (**)

Синтаксис COM:

iObject->get_VertexParamsArray(&VertexParamsArray)	Получить свойство

Примечание.

1. Свойство доступно только для чтения.
2. Свойство позволяет получить массив SAFEARRAY параметров всех вершин. Это массив VT_DISPATCH, которые можно преобразовать в интерфейсы ICurveVertexParam.

VectorDirectionObject – Направляющий объект вектора в точке сопряжения

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

VectorDirectionObject = Object.VectorDirectionObject(VectorIndex);	PointIndex,	Получить свойство (*)
Object.VectorDirectionObject(VectorIndex) = VectorDirectionObject;	PointIndex,	Установить свойство (*)
VectorDirectionObject = Object.GetVectorDirectionObject(VectorIndex);	PointIndex,	Получить свойство (**)
Object.SetVectorDirectionObject(VectorIndex, VectorDirectionObject);	PointIndex,	Установить свойство (**)

Синтаксис COM:

Object.get_VectorDirectionObject(VectorIndex, &VectorDirectionObject)	PointIndex,	Получить свойство
Object.put_VectorDirectionObject(VectorIndex, VectorDirectionObject)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.
VectorIndex	- способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.

VectorOrientation – Направление вектора в точке сопряжения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

VectorOrientation = Object.VectorOrientation(PointIndex, VectorIndex);	Получить свойство (*)
Object.VectorOrientation(PointIndex, VectorIndex) = VectorOrientation;	Установить свойство (*)
VectorOrientation = Object.GetVectorOrientation(PointIndex, VectorIndex);	Получить свойство (**)
Object.SetVectorOrientation(PointIndex, VectorIndex, VectorOrientation);	Установить свойство (**)

Синтаксис COM:

Object.get_VectorOrientation(VectorIndex, &VectorOrientation)	PointIndex,	Получить свойство
Object.put_VectorOrientation(VectorIndex, VectorOrientation)	PointIndex,	Установить свойство

Входные параметры:

long PointIndex	- индекс точки.
VectorIndex	- способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.

В Компас 18.1 удалены свойства:

- SplineTangent
- TangentCurve
- VectorLenght

ISpline3D – методы

AddVertex – Создать новую вершину

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddVertex(long Index);

Синтаксис COM:

HRESULT AddVertex([in] long Index,
[out, retval] ICurveVertexParam ** Result);

Возвращаемое значение:

ICurveVertexParam	- указатель на интерфейс параметров вершины ломаной.

ClearTangentParameters – Удалить параметры управления во всех точках

Интерфейс...

Синтаксис Automation:

BOOL ClearTangentParameters();

Синтаксис COM:

HRESULT ClearTangentParameters(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

DeleteCenterPointParams – Удалить параметры для заданной точки

Интерфейс...

Синтаксис Automation:

BOOL DeleteCenterPointParams(long PointIndex);

Синтаксис COM:

HRESULT DeleteCenterPointParams(long PointIndex, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входные параметры:

PointIndex	- индекс точки.

DeleteVertex – Удалить вершину с указанным индексом

Интерфейс...

Синтаксис Automation:

BOOL DeleteVertex(long Index);

Синтаксис COM:

HRESULT DeleteVertex([in] long Index,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

Index - индекс вершины.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Flush – Очистить массив вершин

Интерфейс...

Синтаксис Automation:

BOOL Flush();

Синтаксис COM:

HRESULT Flush([out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

GetCenterPoint – Угол вектора производной в указанной точке

Интерфейс...

Синтаксис Automation:

BOOL GetCenterPoint(long PointIndex, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetCenterPoint(long PointIndex, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

PointIndex - индекс точки.

Выходные параметры:

X, Y, Z - координаты точки.

GetVector – Получить направление касательного вектора (в сопаряжении)

Интерфейс...

Синтаксис Automation:

BOOL GetVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения.

Входные параметры:

PointIndex - индекс точки,
VectorIndex - способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.

Выходные параметры:

X, Y, Z - координаты направления вектора.

GetVertexCoordinatesArray – Получить массив координат и весов вершин в виде SAFEARRAY double – VT_ARRAY | VT_R8

Интерфейс...

Синтаксис Automation:

BOOL GetVertexCoordinatesArray(VARIANT * Coordinates,
VARIANT * Weights);

Синтаксис COM:

HRESULT GetVertexCoordinatesArray([out] VARIANT * Coordinates,
[out] VARIANT * Weights,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

Coordinates - массив SafeArray типа VT_ARRAY | VT_R8 координат точек всех вершин,

Weights	- массив SafeArray типа VT_ARRAY I VT_R8 весов каждой из всех вершин.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

1. Параметры Coordinates и Weights не являются обязательными. В функцию достаточно передать один из указателей на VARIANT.
2. Массивы являются согласованными:

Координаты точек в массиве Coordinates лежат в следующей последовательности:

- x0, y0, z0, x1, y1, z1 ...xi, yi, zi,

Веса точек в массиве Weights лежат в последовательности:

- weight0, weight1, ...weighti.

InvertVector – Сменить направление касательного вектора на обратное (в сопряжении)

Интерфейс...

Синтаксис Automation:

BOOL InvertVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex);

Синтаксис COM:

HRESULT InvertVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входные параметры:

PointIndex	- индекс точки,
VectorIndex	- способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.

ReadFromFile – Прочитать параметры вершин из текстового файла

Интерфейс...

Синтаксис Automation:

BOOL ReadFromFile(BSTR fileName);

Синтаксис COM:

HRESULT ReadFromFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

fileName	- имя файла.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

См. также ISpline3D::WriteToFile.

SetCenterPoint – Угол вектора производной в указанной точке

Интерфейс...

Синтаксис Automation:

BOOL SetCenterPoint(long PointIndex, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetCenterPoint(long PointIndex, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входные параметры:

PointIndex	- индекс точки,
X, Y, Z	- координаты точки.

SetVector – Установить направление касательного вектора

Интерфейс...

Синтаксис Automation:

BOOL SetVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetVector(long PointIndex, ksTransitionVectorIndexEnum VectorIndex, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входные параметры:

PointIndex	- индекс точки,
VectorIndex	- способ создания сопряжения в заданной вершине сплайна из перечисления ksTransitionVectorIndexEnum.
X, Y, Z	- координаты направления вектора.

WriteToFile – Записать параметры вершин в текстовый файл

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(BSTR fileName);

Синтаксис COM:

HRESULT WriteToFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL * Result);

Входные параметры:

fileName	- имя файла.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание.

См. также ISpline3D::ReadFromFile.

В Компас 18.1 удалены методы:

- SetTangentVector
- GetTangentVector
- Invert

Интерфейс ISplineOnSurface

Справка системы КОМПАС...

kompas.chm::/CM_SPLINE_ON_SURFACE.htm

Сплайн на поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISplineOnSurface

Примечание:

Интерфейс можно получить у коллекции сплайнов по поверхностям с помощью свойства ISplinesOnSurfaces::SplineOnSurface и метода ISplinesOnSurfaces::Add.

ISplineOnSurface – свойства

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject(Index)	Получить свойство (*)
Object.AssociationObject(Index) = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject(Index)	Получить свойство (**)
Object.SetAssociationObject(Index, AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(Index, &AssociationObject)	Получить свойство
Object.put_AssociationObject(Index, AssociationObject)	Установить свойство

Входные параметры:

long Index - индекс вершины.

Closed – Замкнутость

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Closed = Object.Closed	Получить свойство (*)
Object.Closed = Closed	Установить свойство (*)

```
Closed                    =   Получить свойство (**)
Object.GetClosed()
Object.SetClosed( Closed )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Closed(                Получить свойство
&Closed )
Object.put_Closed(                Установить свойство
Closed )
```

Reverse – Противоположное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Reverse = Object.Reverse( Index )          Получить свойство (* )
Object.Reverse( Index ) = Reverse          Установить свойство (* )
Reverse = Object.GetReverse( Index )       Получить свойство (**)
Object.SetReverse( Index, Reverse )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Reverse( Index, &Reverse )      Получить свойство
Object.put_Reverset(  Index, Reverse )     Установить свойство
```

Входные параметры:

long Index - индекс вершины.

Свойство позволяет устанавливать и получать признак того, что направление изменено на противоположное.

SplineOnPoles – Тип построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
SplineOnPoles = Object.SplineOnPoles          Получить свойство (* )
Object.SplineOnPoles = SplineOnPoles          Установить свойство (* )
SplineOnPoles = Object.GetSplineOnPoles()     Получить свойство (**)
Object.SetSplineOnPoles( SplineOnPoles )      Установить свойство (**)
```

Синтаксис COM:

Object.get_SplineOnPoles(&SplineOnPoles)	Получить свойство
Object.put_SplineOnPoles(SplineOnPoles)	Установить свойство

TRUE - по полюсам
FALSE - по точкам.

SplineOrder – Порядок сплайна

Интерфейс...

Тип данных: long

Синтаксис Automation:

SplineOrder = Object.SplineOrder	Получить свойство (*)
Object.SplineOrder = SplineOrder	Установить свойство (*)
SplineOrder = Object.GetSplineOrder()	Получить свойство (**)
Object.SetSplineOrder(SplineOrder)	Установить свойство (**)

Синтаксис COM:

Object.get_SplineOrder(&SplineOrder)	Получить свойство
Object.put_SplineOrder(SplineOrder)	Установить свойство

SurfaceObject – Установить поверхность

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

SurfaceObject = Object.SurfaceObject	Получить свойство (*)
Object.SurfaceObject = SurfaceObject	Установить свойство (*)
SurfaceObject = Object.GetSurfaceObject()	Получить свойство (**)
Object.SetSurfaceObject(SurfaceObject)	Установить свойство (**)

Синтаксис COM:

Object.get_SurfaceObject(&SurfaceObject)	Получить свойство
Object.put_SurfaceObject(SurfaceObject)	Установить свойство

SplineTangent – Тип касательного вектора в вершине

Интерфейс...

Тип данных: из перечисления ksSplineTangentEnum

Синтаксис Automation:

SplineTangent = Object.SplineTangent(Index)	Получить свойство (*)
Object.SplineTangent(Index) = SplineTangent	Установить свойство (*)
SplineTangent = Object.GetSplineTangent(Index)	Получить свойство (**)
Object.SetSplineTangent(Index, SplineTangent)	Установить свойство (**)

Синтаксис COM:

Object.get_SplineTangent(Index, &SplineTangent)	Получить свойство
Object.put_SplineTangent(Index, SplineTangent)	Установить свойство

Входные параметры:

long Index - индекс вершины.

TangentCurve – Касательная кривая

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

TangentCurve = Object.TangentCurve(Index)	Получить свойство (*)
Object.TangentCurve(Index) = TangentCurve	Установить свойство (*)
TangentCurve = Object.GetTangentCurve(Index)	Получить свойство (**)
Object.SetTangentCurve(Index, TangentCurve)	Установить свойство (**)

Синтаксис COM:

Object.get_TangentCurve(Index, &TangentCurve)	Получить свойство
Object.put_TangentCurve(Index, TangentCurve)	Установить свойство

Входные параметры:

long Index - индекс вершины.

VertexCount – Получить количество вершин

Интерфейс...

Тип данных: long

Синтаксис Automation:

VertexCount = Object.VertexCount	Получить свойство (*)
VertexCount = Object.GetVertexCount()	Получить свойство (**)

Синтаксис COM:

Object.get_VertexCount(&VertexCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

VectorLenght – Длина вектора

Интерфейс...

Тип данных: double

Синтаксис Automation:

VectorLenght = Object.VectorLenght(Index)	Получить свойство (*)
Object.VectorLenght(Index) = VectorLenght	Установить свойство (*)
VectorLenght = Object.GetVectorLenght(Index)	Получить свойство (**)
Object.SetVectorLenght(Index, VectorLenght)	Установить свойство (**)

Синтаксис COM:

Object.get_VectorLenght(Index, &VectorLenght)	Получить свойство
Object.put_VectorLenght(Index, VectorLenght)	Установить свойство

Входные параметры:

long Index - индекс вершины.

ISplineOnSurface – методы

AddPoint – Создание новой вершины

Интерфейс...

Синтаксис Automation:

BOOL AddPoint(long IndexAt, double X, double Y, double Z, double W, IModelObject * AssociationObject);

Синтаксис COM:

HRESULT AddPoint(long IndexAt, double X, double Y, double Z, double W, IModelObject * AssociationObject, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

IndexAt	- индекс вершины,
X	- координата по X,
Y	- координата по Y,
Z	- координата по Z
W	- вес,
Association Object	- ассоциативная вершина.

AddVertex – Создание новой вершины

Интерфейс...

Синтаксис Automation:

BOOL AddVertex(long IndexAt,
double U,
double V,
double W,
IModelObject * AssociationObject);

Синтаксис COM:

HRESULT AddVertex(long IndexAt,
double U,
double V,
double W,
IModelObject * AssociationObject,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

IndexAt	- индекс вершины, перед которой нужно добавить новую,
U	- координата по U в %,
V	- координата по V в %,
W	- вес,
Association Object	- ассоциативная вершина.

Clear – Очистить массив вершин

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

ClearTangentParameters – Удалить параметры управления во всех точках

Интерфейс...

Синтаксис Automation:

BOOL ClearTangentParameters();

Синтаксис COM:

HRESULT ClearTangentParameters(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DeleteVertex – Удалить вершину с указанным индексом

Интерфейс...

Синтаксис Automation:

BOOL DeleteVertex(long Index);

Синтаксис COM:

HRESULT DeleteVertex(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины.

GetPoint – Получить координаты и вес вершины

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long Index, double * X, double * Y, double * Z, double * W);

Синтаксис COM:

HRESULT GetPoint(long Index, double * X, double * Y, double * Z, double * W, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины.

Выходные параметры:

X	- координата по X,
Y	- координата по Y,
Z	- координата по Z,
W	- вес.

GetTangentVector – Получить направление касательного вектора

Интерфейс...

Синтаксис Automation:

BOOL GetTangentVector(long Index, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetTangentVector(long Index, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины.

Выходные параметры:

X	- координата по X,
Y	- координата по Y,
Z	- координата по Z.

GetVertex – Получить координаты UV и вес вершины

Интерфейс...

Синтаксис Automation:

BOOL GetVertex(long Index, double * U, double * V, double * W);

Синтаксис COM:

HRESULT GetVertex(long Index, double * U, double * V, double * W, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины.

Выходные параметры:

U	- координата по U в %,
V	- координата по V в %,
W	- вес.

GetVertexParams – Массив параметров UV и весов вершин в виде SAFEARRAY double – VT_ARRAY | VT_R8

Интерфейс...

Синтаксис Automation:

BOOL GetVertexParams(VARIANT * UV, VARIANT * Points, VARIANT * Weights);

Синтаксис COM:

HRESULT GetVertexParams(VARIANT * UV, VARIANT * Points, VARIANT * Weights, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

UV	- массив VT_ARRAY I VT_R8 параметров UV,
Points	- массив VT_ARRAY I VT_R8 координат вершин,
Weights	- массив VT_ARRAY I VT_R8 весов вершин.

Invert – Сменить направление касательного вектора на обратное

Интерфейс...

Синтаксис Automation:

BOOL Invert(long Index);

Синтаксис COM:

HRESULT Invert(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины.

SetPoint – Установить координаты и вес вершины

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(long Index, double X, double Y, double Z, double W);

Синтаксис COM:

HRESULT SetPoint(long Index, double X, double Y, double Z, double W, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X	- координата по X,
Y	- координата по Y,
Z	- координата по Z,
W	- вес.

SetTangentVector – Установить направление касательного вектора

Интерфейс...

Синтаксис Automation:

BOOL SetTangentVector(long Index, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetTangentVector(long Index, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины,
X	- координата по X,
Y	- координата по Y,
Z	- координата по Z.

SetVertex – Установить координаты UV и вес вершины

Интерфейс...

Синтаксис Automation:

BOOL SetVertex(long Index, double U, double V, double W);

Синтаксис COM:

HRESULT SetVertex(long Index, double U, double V, double W, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс вершины,
U	- координата по U в %,
V	- координата по V в %,
W	- вес.

Интерфейс ISurfacesIntersectionCurve

Справка системы КОМПАС...

`kompas.chm::/1021_108_13_Krivay_peresech_poverchnostey.htm`

Интерфейс кривых пересечений поверхностей.

Иерархия:

IKompasAPIObject

IModelObject

ISurfacesIntersectionCurve

Описание:

Интерфейс позволяет создать кривую пересечения двух поверхностей, или двух наборов поверхностей, или поверхности и набора.

Примечание:

Интерфейс можно получить с помощью свойства ISurfacesIntersectionCurves::SurfacesIntersectionCurve и метода ISurfacesIntersectionCurves::Add.

ISurfacesIntersectionCurve – свойства

AutoCheck – Включение/выключение автоматического выбора всех массивов ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AutoCheck = Object.AutoCheck      Получить свойство (* )
Object.AutoCheck = AutoCheck      Установить свойство (* )
AutoCheck                   =     Получить свойство (**)
Object.GetAutoCheck()
Object.SetAutoCheck(              Установить свойство (**)
AutoCheck )
```

Синтаксис COM:

```
Object.get_AutoCheck(                    Получить свойство
&AutoCheck )
Object.put_AutoCheck(                    Установить свойство
AutoCheck )
```

Свойство позволяет включать и отключать автоматический выбор всех массивов ребер.

EdgesArraysCount – Количество массивов ребер в кривой пересечения поверхностей

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
EdgesArraysCount            =     Получить свойство (* )
Object.EdgesArraysCount
EdgesArraysCount            =     Получить свойство (**)
Object.GetEdgesArraysCount()
```

Синтаксис COM:

```
Object.get_EdgesArraysCount(             Получить свойство
&EdgesArraysCount )
```

Свойство позволяет получить количество массивов ребер в кривой пересечения поверхностей.

Примечание:

Свойство доступно только для чтения.

Edges – Количество массивов ребер в кривой пересечения поверхностей

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges(EdgesArrayIndex)	Получить свойство (*)
Edges = Object.GetEdges(EdgesArrayIndex)	Получить свойство (**)

Синтаксис COM:

Object.get_Edges(EdgesArrayIndex, &Edges)	Получить свойство

Входные параметры:

EdgesArrayIndex	- индекс массива ребер

Примечание:

Свойство доступно только для чтения.

В зависимости от количества ребер возвращается:

- нет ребер - VT_EMPTY (если указан неправильный индекс или ребра не созданы),
- одно ребро - VT_DISPATCH,
- более одного ребра - VT_ARRAY | VT_DISPATCH.

Индекс должен быть в диапазоне 0...ISurfacesIntersectionCurve::EdgesArraysCount-1.

EdgesCheck – Признак выбора указанного по индексу массива ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EdgesCheck = Object.EdgesCheck(EdgesArrayIndex)	Получить свойство (*)
Object.EdgesCheck(EdgesArrayIndex) = EdgesCheck	Установить свойство (*)

```
EdgesCheck = Object.GetEdgesCheck( EdgesArrayIndex )    Получить свойство (**)
Object.SetEdgesCheck( EdgesArrayIndex, EdgesCheck)      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EdgesCheck(                    Получить свойство
EdgesArrayIndex,   &EdgesCheck
)
Object.put_EdgesCheck(                    Установить свойство
EdgesArrayIndex, EdgesCheck )
```

Входные параметры:

```
EdgesArrayIndex                 - индекс массива ребер.
```

Примечание:

Индекс должен быть в диапазоне 0...ISurfacesIntersectionCurve::EdgesArraysCount-1.

EdgesChecks – Признаки выбора массивов ребер

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_BOOL

Синтаксис Automation:

```
EdgesChecks = Object.EdgesChecks                Получить свойство (* )
Object.EdgesChecks = EdgesChecks                Установить свойство (* )
EdgesChecks = Object.GetEdgesChecks()           Получить свойство (**)
Object.SetEdgesChecks( EdgesChecks )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EdgesChecks(                   Получить свойство
&EdgesChecks )
Object.put_EdgesChecks(                   Установить свойство
EdgesChecks )
```

Свойство позволяет устанавливать и получать признаки выбора массива ребер.

EdgesChecksCount – Количество доступных для создания контуров

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
EdgesChecksCount            =   Получить свойство (* )
Object.EdgesChecksCount
```

EdgesChecksCount Object.GetEdgesChecksCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesChecksCount(&EdgesChecksCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ObjectsCount – Получить количество объектов в указанном наборе

Интерфейс...

Тип данных: long

Синтаксис Automation:

ObjectsCount Object.ObjectsCount(ForArray1)	=	Получить свойство (*)
ObjectsCount Object.GetObjectsCount(ForArray1)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ObjectsCount(ForArray1, &ObjectsCount)	Получить свойство

Входные параметры:

ForArray1	- TRUE - первый набор, - FALSE - второй набор.

Свойство позволяет получить количество объектов в указанном наборе.

Примечание:

Свойство доступно только для чтения.

ISurfacesIntersectionCurve – методы

AddObjects – Добавить элемент(ы) в указанный набор

Интерфейс...

Синтаксис Automation:

BOOL AddObjects(BOOL ForArray1, VARIANT * Objects);

Синтаксис COM:

HRESULT AddObjects(BOOL ForArray1, VARIANT * Objects, BOOL * Res);

Выходные параметры:

ForArray1	- TRUE - первый набор, - FALSE - второй набор,
Objects	- набор граней.

Возвращаемое значение:

TRUE	- в случае удачи.

Примечание:

Если в набор добавляется одна грань, то ее можно передать как VT_DISPATCH.

Если в набор добавляется несколько граней, то нужно передать массив VT_ARRAY | VT_DISPATCH.

Clear – Очистить указанный набор объектов

Интерфейс...

Синтаксис Automation:

BOOL Clear(BOOL ForArray1);

Синтаксис COM:

HRESULT Clear(BOOL ForArray1, BOOL * Res);

Выходные параметры:

ForArray1	- TRUE - первый набор, - FALSE - второй набор.

Возвращаемое значение:

TRUE	- в случае удачи.

GetObject – Получить объект из указанного набора по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObject(BOOL ForArray1, long Index);

Синтаксис COM:

HRESULT GetObject(BOOL ForArray1, long Index, IModelObject **Object);

Выходные параметры:

ForArray1	- TRUE - первый набор, - FALSE - второй набор,
Index	- индекс объекта в наборе.

Возвращаемое значение:

- указатель на интерфейс IModelObject.

GetObjects – Получить 1-й и 2-й наборы поверхностей

Интерфейс...

Синтаксис Automation:

BOOL GetObjects(VARIANT * ObjectsArray1, VARIANT * ObjectsArray2);

Синтаксис COM:

HRESULT GetObjects(VARIANT * ObjectsArray1, VARIANT * ObjectsArray2, BOOL * Res);

Выходные параметры:

ObjectsArray1	- первый набор граней,
ObjectsArray2	- первый набор граней.

Возвращаемое значение:

TRUE	- в случае удачи.

Примечание:

Если в наборе одна грань, то она возвращается как VT_DISPATCH.

Если в наборе несколько граней, то возвращается массив VT_ARRAY | VT_DISPATCH.

SetObjects – Задать 1-й и 2-й наборы поверхностей

Интерфейс...

Синтаксис Automation:

BOOL SetObjects(VARIANT ObjectsArray1, VARIANT ObjectsArray2)

Синтаксис COM:

HRESULT SetObjects(VARIANT ObjectsArray1, VARIANT ObjectsArray2, BOOL * Res);

Входные параметры:

ObjectsArray1	- первый набор граней,
ObjectsArray2	- первый набор граней.

Возвращаемое значение:

TRUE	- в случае удачи.

Примечание:

Если требуется передать одну грань в наборе, ее можно передать как VT_DISPATCH.

Если требуется передать несколько граней в наборе, то нужно передать его как массив VT_ARRAY | VT_DISPATCH.

Интерфейс ITrimmedCurve

Справка системы КОМПАС...

kompas.chm::/CM_CURVEOPER_CUT.htm

Интерфейс параметров операции усечения кривой

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ITrimmedCurve

ITrimmedCurve – свойства

BeginParameter – Начальный параметр кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

BeginParameter = Object.BeginParameter	Получить свойство (*)
Object.BeginParameter = BeginParameter	Установить свойство (*)
BeginParameter = Object.GetBeginParameter()	Получить свойство (**)
Object.SetBeginParameter(BeginParameter)	Установить свойство (**)

Синтаксис COM:

Object.get_BeginParameter(&BeginParameter)	Получить свойство
Object.put_BeginParameter(BeginParameter)	Установить свойство

CutObject1 – Первый секущий объект для усечения кривой

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

CutObject1 = Object.CutObject1	Получить свойство (*)
Object.CutObject1 = CutObject1	Установить свойство (*)
CutObject1 = Object.GetCutObject1()	Получить свойство (**)
Object.SetCutObject1(CutObject1)	Установить свойство (**)

Синтаксис COM:

Object.get_CutObject1(&CutObject1)	Получить свойство
Object.put_CutObject1(CutObject1)	Установить свойство

Свойство позволяет устанавливать и получать первый секущий объект.

CutObject2 – Второй секущий объект для усечения кривой

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

CutObject2 = Object.CutObject2	Получить свойство (*)
Object.CutObject2 = CutObject2	Установить свойство (*)
CutObject2 = Object.GetCutObject2()	Получить свойство (**)
Object.SetCutObject2(CutObject2)	Установить свойство (**)

Синтаксис COM:

Object.get_CutObject2(&CutObject2)	Получить свойство
Object.put_CutObject2(CutObject2)	Установить свойство

Свойство позволяет устанавливать и получать второй секущий объект.

Curve – Усекаемая кривая

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve = Object.Curve	Получить свойство (*)
Object.Curve = Curve	Установить свойство (*)
Curve = Object.GetCurve()	Получить свойство (**)
Object.SetCurve(Curve)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Curve( &Curve )                    Получить свойство
Object.put_Curve(  Curve )                    Установить свойство
```

Свойство позволяет устанавливать и получать усекаемую кривую.

EndParameter – Конечный параметр кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
EndParameter = Object.EndParameter                    Получить свойство (* )
Object.EndParameter = EndParameter                    Установить свойство (* )
EndParameter = Object.GetEndParameter()               Получить свойство (**)
Object.SetEndParameter( EndParameter )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EndParameter( &EndParameter )      Получить свойство
Object.put_EndParameter(  EndParameter )      Установить свойство
```

Sense – Сменить направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Sense = Object.Sense                          Получить свойство (* )
Object.Sense = Sense                          Установить свойство (* )
Sense = Object.GetSense()                     Получить свойство (**)
Object.SetSense( Sense )                      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sense( &Sense )                    Получить свойство
Object.put_Sense(  Sense )                    Установить свойство
```

Свойство позволяет менять направление усечения.

Примечание:

Свойство работает только при UseTwoCutObjecs = FALSE.

UseTwoCutObjecs – Использовать два секущих объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseTwoCutObjecs = Object.UseTwoCutObjecs	Получить свойство (*)
Object.UseTwoCutObjecs = UseTwoCutObjecs	Установить свойство (*)
UseTwoCutObjecs = Object.GetUseTwoCutObjecs()	Получить свойство (**)
Object.SetUseTwoCutObjecs(UseTwoCutObjecs)	Установить свойство (**)

Синтаксис COM:

Object.get_UseTwoCutObjecs(&UseTwoCutObjecs)	Получить свойство
Object.put_UseTwoCutObjecs(UseTwoCutObjecs)	Установить свойство

Свойство позволяет устанавливать и получать признак использования двух секущих объектов.

ITrimmedCurve – методы

GetIntersectParameters – Получить массивы точек пересечения кривой с секущими объектами

Интерфейс...

Синтаксис Automation:

BOOL GetIntersectParameters(VARIANT * CutPoints1, VARIANT * CutPoints2);

Синтаксис COM:

HRESULT GetIntersectParameters(VARIANT * CutPoints1, VARIANT * CutPoints2, BOOL * Result);

Выходные параметры

CutPoints1 - Массив VT_ARRAY | VT_R8 параметров в точках пересечения с первым секущим объектом.

CutPoints2 - Массив VT_ARRAY | VT_R8 параметров в точках пересечения со вторым секущим объектом.

Интерфейс IUnhistoredCurve3D

Интерфейс кривой без истории.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IUnhistoredCurve3D

Интерфейс можно получить у коллекции кривых без истории с помощью свойства IUnhistoredCurves3D::UnhistoredCurve3D и методов IUnhistoredCurves3D::Add и IUnhistoredCurves3D::Load.

IUnhistoredCurve3D – методы

Replace – Изменить кривую

Интерфейс...

Синтаксис Automation:

BOOL Replace(IModelObject * Curve, BOOL DeleteSource);

Синтаксис COM:

HRESULT Replace(IModelObject * Curve, BOOL DeleteSource, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

Curve - указатель на кривую,
DeleteSource - удалить исходную кривую.

Метод заменяет математику в кривой без истории на математику с переданной кривой.

Unwrap – Построить развертку

Интерфейс...

Синтаксис Automation:

BOOL Unwrap(VARIANT Placement, double X, double Y, double Z, double Accuracy);

Синтаксис COM:

HRESULT Unwrap(VARIANT Placement, double X, double Y, double Z, double Accuracy, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

Placement - положение плоскости, на которой требуется построить развертку,
X, Y, Z - координаты начала построения развертки,
Accuracy - точность.

Интерфейс IUserFolder

Интерфейс пользовательской директории.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IUserFolder

Данный интерфейс можно получить с помощью метода IUserFolders::Add.

Версия: КОМПАС v18.

IUserFolder – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM

HRESULT Add(ksObj3dTypeEnum Type, IModelObject * * Result);

Возвращаемое значение:

-указатель на интерфейс IModelObject.

Входные параметры:

Type	- тип объекта из перечисления ksObj3dTypeEnum

GetObjects – Получить набор объектов в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Синтаксис Automation:

VARIANT GetObjects();

Синтаксис COM:

HRESULT GetObjects(VARIANT * Result);

Возвращаемое значение:

Коллекция объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

SetObjects – Задать объекты

Интерфейс...

Синтаксис Automation:

BOOL SetObjects(VARIANT Objects);

Синтаксис COM:

HRESULT SetObjects(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Objects	- набор объектов.

Примечание:

- Если требуется передать один объект в наборе, его можно передать как VT_DISPATCH.
- Если требуется передать несколько объектов в наборе, то нужно передать их как массив VT_ARRAY | VT_DISPATCH.

Интерфейс IUserObject3D

Интерфейс пользовательского объекта 3D.

Иерархия:

IDispatch
  IKompasAPIObject
    IModelObject
      IUserObject3D

Имеет дополнительный интерфейс IUserParameters, который можно получить с помощью метода IUnknown::QueryInterface.

IUserObject3D – свойства

AssociationObject – Опорный объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject Object.AssociationObject(Index);	=	Получить свойство (*)
Object.AssociationObject(Index) AssociationObject;	=	Установить свойство (*)

AssociationObject = Object.GetAssociationObject(Index);		Получить свойство (**)
Object.SetAssociationObject(Index, AssociationObject);		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(Index, &AssociationObject)	Получить свойство
Object.put_AssociationObject(Index, AssociationObject)	Установить свойство

Входные параметры:

long Index	- индекс объекта.

AssociationObjectCount – Количество опорных объектов

Интерфейс...

Тип данных: Указатель на интерфейс long

Синтаксис Automation:

AssociationObjectCount = Object.AssociationObjectCount	=	Получить свойство (*)
AssociationObjectCount = Object.GetAssociationObjectCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_AssociationObjectCount(&AssociationObjectCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ObjectID – Идентификатор объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ObjectID = Object.ObjectID	Получить свойство (*)
Object.ObjectID = ObjectID	Установить свойство (*)
ObjectID = Object.GetObjectID()	Получить свойство (**)
Object.SetObjectID(ObjectID)	Установить свойство (**)

Синтаксис COM:

Object.get_ObjectID(&ObjectID)	Получить свойство
Object.put_ObjectID(ObjectID)	Установить свойство

PropertyObjectEditable – Поддерживается интерфейс внешних свойств объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PropertyObjectEditable = Object.PropertyObjectEditable	=	Получить свойство (*)
Object.PropertyObjectEditable PropertyObjectEditable	=	Установить свойство (*)
PropertyObjectEditable = Object.GetPropertyObjectEditable()	=	Получить свойство (**)
Object.SetPropertyObjectEditable(PropertyObjectEditable)		Установить свойство (**)

Синтаксис COM:

Object.get_PropertyObjectEditable(&PropertyObjectEditable)	Получить свойство
Object.put_PropertyObjectEditable(PropertyObjectEditable)	Установить свойство

UserParameters – Получить интерфейс параметров

Интерфейс...

Тип данных: Указатель на интерфейс IUnknown

Синтаксис Automation:

UserParameters = Object.UserParameters	=	Получить свойство (*)
UserParameters = Object.GetUserParameters()	=	Получить свойство (**)

Синтаксис COM:

Object.get_UserParameters(&UserParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IUserObject3D – методы

ClearAssociationObject – Удалить все опорные объекты

Интерфейс...

Синтаксис Automation:

BOOL ClearAssociationObject();

Синтаксис COM:

HRESULT ClearAssociationObject(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IEdge

Интерфейс Ребро 3D.

Иерархия:

IDispatch
  IKompasAPIObject
    IModelObject
      IEdge

КОМПАС версия v18

IEdge – свойства

AdjacentFace – Получить указатель на интерфейс грани, в цикл которой входит ребро

Интерфейс...

Тип данных: Указатель на интерфейс IFace.

Синтаксис Automation:

BaseAdjacentFace Object.BaseAdjacentFace	=	Получить свойство (*)
BaseAdjacentFace Object.GetBaseAdjacentFace()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BaseAdjacentFace(&BaseAdjacentFace) — Получить свойство

Примечание:

Свойство доступно только для чтения.

BaseCurve3DType – Тип базовой кривой

Интерфейс...

Тип данных: Значение из перечисления ksMathCurve3DTypeEnum.

Синтаксис Automation:

BaseCurve3DType = Object.BaseCurve3DType — Получить свойство (*)

BaseCurve3DType = Object.GetBaseCurve3DType() — Получить свойство (**)

Синтаксис COM:

Object.get_BaseCurve3DType(&BaseCurve3DType) — Получить свойство

Примечание:

Свойство доступно только для чтения.

Curve3DType – Тип кривой

Интерфейс...

Тип данных: Значение из перечисления ksMathCurve3DTypeEnum.

Синтаксис Automation:

Curve3DType = Object.Curve3DType — Получить свойство (*)

Curve3DType = Object.GetCurve3DType() — Получить свойство (**)

Синтаксис COM:

Object.get_Curve3DType(&Curve3DType) — Получить свойство

Примечание:

Свойство доступно только для чтения.

Edges – Получить указатель на интерфейс массива ребер, стыкующихся с данным ребром

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Edges = Object.Edges( Start )        Получить свойство (* )
Edges = Object.GetEdges( Start )     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Edges( Start, &Edges )        Получить свойство
```

Примечание:

Свойство доступно только для чтения.

IsPeriodic – Является ли ребро периодичным

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsPeriodic = Object.IsPeriodic       Получить свойство (* )
IsPeriodic = Object.GetIsPeriodic()  Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsPeriodic( &IsPeriodic )     Получить свойство
```

Примечание:

Свойство доступно только для чтения.

IsSketchEdge – Является ли ребро кривой эскиза

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsSketchEdge = Object.IsSketchEdge		Получить свойство (*)
IsSketchEdge Object.GetIsSketchEdge()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsSketchEdge(&IsSketchEdge)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IsStraight – Является ли ребро прямым

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsStraight = Object.IsStraight		Получить свойство (*)
IsStraight Object.GetIsStraight()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsStraight(&IsStraight)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MathCurve – Получить указатель на интерфейс математической кривой

Интерфейс...

Тип данных: Указатель на интерфейс IMathCurve3D.

Синтаксис Automation:

MathCurve = Object.MathCurve		Получить свойство (*)
MathCurve Object.GetMathCurve()	=	Получить свойство (**)

Синтаксис COM:

Object.get_MathCurve(&MathCurve) — Получить свойство

Примечание:

Свойство доступно только для чтения.

OrientedEdges – Получить указатель на массив ориентированных ребер

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

OrientedEdges = Object.OrientedEdges — Получить свойство (*)

OrientedEdges = Object.GetOrientedEdges() — Получить свойство (**)

Синтаксис COM:

Object.get_OrientedEdges(&OrientedEdges) — Получить свойство

Примечание:

Свойство доступно только для чтения.

Vertex – Получить указатель на интерфейс вершин: начальной и конечной

Интерфейс...

Тип данных: Указатель на интерфейс IVertex.

Синтаксис Automation:

Vertex = Object.Vertex(Start) — Получить свойство (*)

Vertex = Object.GetVertex(Start) — Получить свойство (**)

Синтаксис COM:

Object.get_Vertex(Start, &Vertex) — Получить свойство

Примечание:

Свойство доступно только для чтения.

IEdge – методы

GetLength – Получить длину ребра

Интерфейс...

Синтаксис Automation:

double GetLength(ksLengthUnitsEnum Unit);

Синтаксис COM:

HRESULT GetLength(ksLengthUnitsEnum Unit, double * Result);

Возвращаемое значение:

-длина ребра.

Входные параметры:

Unit - единицы измерения длины из перечисления ksLengthUnitsEnum.

Интерфейс IFace

Интерфейс грани.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFace

КОМПАС версия v18

IFace – свойства

BaseSurface3DType – Тип базовой поверхности

Интерфейс...

Тип данных: из перечисления ksMathSurface3DTypeEnum.

Синтаксис Automation:

BaseSurface3DType Object.BaseSurface3DType	=	Получить свойство (*)
BaseSurface3DType Object.GetBaseSurface3DType()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BaseSurface3DType(&BaseSurface3DType)		Получить свойство

Примечание:

Свойство доступно только для чтения.

ConnectedFaces – Массив граней, стыкующихся с данной гранью в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ConnectedFaces Object.ConnectedFaces	=	Получить свойство (*)
ConnectedFaces Object.GetConnectedFaces()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ConnectedFaces(&ConnectedFaces)		Получить свойство

Примечание:

Свойство доступно только для чтения.

LimitingEdges – Массив ребер, ограничивающих грань в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

LimitingEdges Object.LimitingEdges	=	Получить свойство (*)
LimitingEdges Object.GetLimitingEdges()	=	Получить свойство (**)

Синтаксис COM:

Object.get_LimitingEdges(&LimitingEdges)		Получить свойство

Примечание:

Свойство доступно только для чтения.

Loops – Массив циклов в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Loops = Object.Loops              Получить свойство (* )
Loops = Object.GetLoops()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Loops( &Loops )                Получить свойство
```

Примечание:

Свойство доступно только для чтения.

MathSurface – Получить интерфейс математической поверхности

Интерфейс...

Тип данных: Указатель на интерфейс IMathSurface3D.

Синтаксис Automation:

```
MathSurface = Object. Получить свойство (* )
MathSurface
MathSurface = Object.Get Получить свойство (**)
MathSurface()
```

Синтаксис COM:

```
Object.get_ MathSurface( & Получить свойство
MathSurface )
```

Примечание:

Свойство доступно только для чтения.

NormalOrientation – Признак ориентации нормали

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
NormalOrientation = Object.    Получить свойство (* )
NormalOrientation
NormalOrientation = Object.Get    Получить свойство (**)
NormalOrientation()
```

Синтаксис COM:

```
Object.get_ NormalOrientation(                    Получить свойство
& NormalOrientation )
```

Примечание:

Свойство доступно только для чтения.

Radius – Дать максимальный радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius              Получить свойство (* )
Radius = Object.GetRadius()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius )                    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Surface3DType – Тип поверхности

Интерфейс...

Тип данных: из перечисления ksMathSurface3DTypeEnum.

Синтаксис Automation:

```
Surface3DType                 =    Получить свойство (* )
Object.Surface3DType
Surface3DType                 =    Получить свойство (**)
Object.GetSurface3DType()
```

Синтаксис COM:

```
Object.get_Surface3DType(                    Получить свойство
&Surface3DType )
```

Примечание:

Свойство доступно только для чтения.

Tessellation – Параметры триангуляционной сетки

Интерфейс...

Тип данных: Указатель на интерфейс ITessellation7.

Синтаксис Automation:

Tessellation Object.Tessellation	=	Получить свойство (*)
Tessellation Object.GetTessellation()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Tessellation(&Tessellation)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IFace – методы

GetConeParam – Получить параметры конической грани

Интерфейс...

Синтаксис Automation:

BOOL GetConeParam(double * Height, double * Angle, double * Radius);

Синтаксис COM:

HRESULT GetConeParam(double * Height, double * Angle, double * Radius, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

Height	- высота,
Angle	- угол,
Radius	- радиус.

GetArea – Получить площадь грани

Интерфейс...

Синтаксис Automation:

double GetArea(ksLengthUnitsEnum Unit);

Синтаксис COM:

HRESULT GetArea(ksLengthUnitsEnum Unit, double * Result);

Возвращаемое значение:

- площадь грани.

Входные параметры:

Unit	- единицы измерения длины из перечисления ksLengthUnitsEnum.

Интерфейс ILoop7

Интерфейс цикла (содержит информацию о связях с другими гранями).

Иерархия:

IDispatch

IKompasAPIObject

ILoop7

Данный интерфейс можно получить с помощью метода IFace::Loops.

КОМПАС версия v18

ILoop7 – свойства

Edges – Получить массив ребер

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить массив SafeArray (VT_APPAY|VT_DISPATCH) ребер грани IEdge.

IsOuter – TRUE – цикл внешний, FALSE – цикл внутренний

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsOuter = Object.IsOuter          Получить свойство (* )
IsOuter = Object.GetIsOuter()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsOuter( &IsOuter )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

OrientedEdges – Получить массив ориентированных ребер

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
OrientedEdges                 =   Получить свойство (* )
Object.OrientedEdges
OrientedEdges                 =   Получить свойство (**)
Object.GetOrientedEdges()
```

Синтаксис COM:

```
      Object.get_OrientedEdges(          Получить свойство
&OrientedEdges )
```

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить массив SafeArray (VT_APPAY|VT_DISPATCH) ориентированных ребер IOrientedEdge7.

ILoop7 – методы

GetLength – Получить общую длину ребер. bitVector = ST_MIX_MM..ST_MIX_M единицы измерения

Интерфейс...

Синтаксис Automation:

double GetLength(ksLengthUnitsEnum Unit);

Синтаксис COM:

HRESULT GetLength(ksLengthUnitsEnum Unit, double * Result);

Возвращаемое значение:

- общая длина ребер цикла.

Входные параметры:

Unit	- единицы измерения длины из перечисления ksLengthUnitsEnum.

Интерфейс IOrientedEdge7

Интерфейс ориентированного ребра.

Иерархия:

IDispatch

IKompasAPIObject

IOrientedEdge7

КОМПАС версия v18

IOrientedEdge7 – свойства

AdjacentFace – Получить грань, в которой ребро входит в цикл

Интерфейс...

Тип данных: Указатель на интерфейс IFace.

Синтаксис Automation:

AdjacentFace Object.AdjacentFace(FacePlus)	=	Получить свойство (*)
AdjacentFace Object.GetAdjacentFace(FacePlus)	=	Получить свойство (**)

Синтаксис COM:

Object.get_AdjacentFace(FacePlus, &AdjacentFace)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TRUE - ориентация плюс.

Edge – Получить интерфейс базового ребра

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Edge = Object.Edge	Получить свойство (*)
Edge = Object.GetEdge()	Получить свойство (**)

Синтаксис COM:

Object.get_Edge(&Edge)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Next – Получить следующее ориентированное ребро

Интерфейс...

Тип данных: Указатель на интерфейс IOrientedEdge7.

Next = Object.Next	Получить свойство (*)
Next = Object.GetNext()	Получить свойство (**)

Синтаксис COM:

Object.get_Next(&Next)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Orientation – Получить направление относительно базового ребра

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Orientation = Object.Orientation      Получить свойство (* )
Orientation                    =      Получить свойство (**)
Object.GetOrientation()
```

Синтаксис COM:

```
        Object.get_Orientation(              Получить свойство
&Orientation )
```

Примечание:

Свойство доступно только для чтения.

TRUE - направления совпадают.

SameSense – Получить направление относительно кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
SameSense = Object.SameSense    Получить свойство (* )
SameSense                  =    Получить свойство (**)
Object.GetSameSense()
```

Синтаксис COM:

```
        Object.get_SameSense(              Получить свойство
&SameSense )
```

Примечание:

Свойство доступно только для чтения.

TRUE - направления совпадают.

Интерфейс ITessellation7

Интерфейс триангуляции грани.

Иерархия:

IDispatch

ITessellation7

Данный интерфейс можно получить, используя свойство грани IFace::Tessellation.

КОМПАС версия v18

ITessellation7 – свойства

FacetAngle – Ограничение углового отклонения триангуляционной пластины (Если 0, обычная триангуляция)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
FacetAngle = Object.FacetAngle;                 Получить свойство (* )
Object.FacetAngle = FacetAngle;                 Установить свойство (* )
FacetAngle = Object.GetFacetAngle();            Получить свойство (**)
Object.SetFacetAngle( FacetAngle );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_FacetAngle( &FacetAngle )            Получить свойство
Object.put_FacetAngle( FacetAngle )             Установить свойство
```

FacetSag – Ограничение прогиба поверхности (Если 0, обычная триангуляция)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
FacetSag = Object.FacetSag;                     Получить свойство (* )
Object.FacetSag = FacetSag;                     Установить свойство (* )
FacetSag = Object.GetFacetSag();                Получить свойство (**)
Object.SetFacetSag( FacetSag );                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_FacetSag( &FacetSag )                Получить свойство
Object.put_FacetSag(  FacetSag )                Установить свойство
```

FacetSize – Ограничение размера ребра (Если 0, обычная триангуляция)

Интерфейс...

Тип данных: double

Синтаксис Automation:

FacetSize = Object.FacetSize;	Получить свойство (*)
Object.FacetSize = FacetSize;	Установить свойство (*)
FacetSize = Object.GetFacetSize();	Получить свойство (**)
Object.SetFacetSize(FacetSize);	Установить свойство (**)

Синтаксис COM:

Object.get_FacetSize(&FacetSize)	Получить свойство
Object.put_FacetSize(FacetSize)	Установить свойство

NeedParams – Необходимость заполнения параметров вершин

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NeedParams = Object.NeedParams ;		Получить свойство (*)
Object.NeedParams = NeedParams ;		Установить свойство (*)
NeedParams = Object.GetNeedParams ();		Получить свойство (**)
Object.SetNeedParams (NeedParams);		Установить свойство (**)

Синтаксис COM:

Object.get_NeedParams (&NeedParams)		Получить свойство
Object.put_NeedParams (NeedParams)		Установить свойство

ITessellation7 – методы

RebuildTessellation – Перестроить после изменения параметров сетки

Интерфейс...

Синтаксис Automation:

BOOL RebuildTessellation();

Синтаксис COM:

HRESULT RebuildTessellation(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetFacetPoints – Получить параметры вершин треагуляционной сетки

Интерфейс...

Синтаксис Automation:

BOOL GetFacetPoints(VARIANT * Points, VARIANT * Indexes, VARIANT * Normals);

Синтаксис COM:

HRESULT GetFacetPoints(VARIANT * Points, VARIANT * Indexes, VARIANT * Normals, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

Points	- массив SafeArray вещественных чисел - координат вершин триангуляционной сетки,
Indexes	- массив SafeArray целых чисел - индексов вершин триангуляционной сетки,
Normals	- массив SafeArray вещественных чисел - нормалей вершин триангуляционной сетки.

Примечание:

1. В массиве индексов вершин триангуляционной сетки хранятся индексы точек из массива координат точек points. Для получения параметров вершины триангуляционной сетки нужно получить индекс точки из массива indexes.
2. Координаты точек в массиве points лежат в последовательности x, y, z первой точки, x, y, z второй точки и т.д.

3. В массиве нормалей вершин триангуляционной сетки координаты нормалей лежат в последовательности x, y, z - нормаль для первой точки, x, y, z- нормаль второй точки и т.д.

GetFacetParams – Получить параметры вершин триангуляционной сетки

Интерфейс...

Синтаксис Automation:

VARIANT GetFacetParams();

Синтаксис COM:

HRESULT GetFacetParams(VARIANT * Result);

Возвращаемое значение:

SafeArray	- массив вещественных чисел - параметрических координат вершин триангуляционных сетки. Для формирование параметрических координат триангуляционной сетки требуется установить признак ITessellation7::NeedParams.

Интерфейс IVertex

Интерфейс вершины.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IVertex

Данный интерфейс можно получить с помощью метода IEdge::Vertex.

КОМПАС версия v18

IVertex – свойства

IsFreeVertex – Является ли вершина свободной

Интерфейс...

Синтаксис Automation:

IsFreeVertex = Object.IsFreeVertex;	Получить свойство (*)
IsFreeVertex = Object.GetIsFreeVertex();	Получить свойство (**)

Синтаксис COM:

```
Object.get_IsFreeV                Получить свойство
ertex(
&IsFreeVertex )
```

Примечание:

Свойство доступно только для чтения.

IsTopologyVertex – Является ли вершина топологической

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsTopologyVertex = Object.IsTopologyVertex;     Получить свойство (* )
IsTopologyVertex                          =     Получить свойство (**)
Object.GetIsTopologyVertex();
```

Синтаксис COM:

```
Object.get_IsTopologyVertex(                    Получить свойство
&IsTopologyVertex )
```

Примечание:

Свойство доступно только для чтения.

IsSketchVertex – Является ли вершина точкой эскиза

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsSketchVertex                    =    Получить свойство (* )
Object.IsSketchVertex;
IsSketchVertex                    =    Получить свойство (**)
Object.GetIsSketchVertex();
```

Синтаксис COM:

```
Object.get_IsSketchVertex(                Получить свойство
&IsSketchVertex )
```

Примечание:

Свойство доступно только для чтения.

IVertex – методы

GetPoint – Получить координаты вершины

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

Интерфейс ICurveVertexParam

Справка системы КОМПАС...

kompas.chm::/1087_116_5_5_Tablicha_parametrov_vershin.htm

Интерфейс параметров вершины кривой.

Иерархия:

IKompasAPIObject

ICurveVertexParam

Указатель на интерфейс можно получить у сплайнов или ломаных, используя свойство ISpline3D::VertexParams, ISpline3D::VertexParamsArray и метод ISpline3D::AddVertex.

ICurveVertexParam – свойства

AssociationVertex – Ассоциативная вершина

Интерфейс...

Тип данных: Указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

AssociationVertex = iObject.AssociationVertex;	=	Получить свойство (*)
iObject.AssociationVertex = AssociationVertex;	=	Установить свойство (*)

```
AssociationVertex = iObject.GetAssociationVertex();   Получить свойство (**)
iObject.SetAssociationVertex(AssociationVertex);      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_AssociationVertex (&AssociationVertex)   Получить свойство
iObject->put_AssociationVertex (AssociationVertex)    Установить свойство
```

BuildingObject – Объект, относительно которого ведется построение

Интерфейс...

Тип данных: Указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

```
BuildingObject = iObject.BuildingObject;              Получить свойство (* )
iObject.BuildingObject = BuildingObject;              Установить свойство (* )
BuildingObject = iObject.GetBuildingObject();         Получить свойство (**)
iObject.SetBuildingObject(BuildingObject);            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_BuildingObject (&BuildingObject)         Получить свойство
iObject->put_BuildingObject (BuildingObject)          Установить свойство
```

Примечание:

Задать объект, относительно которого ведется построение, можно для типа построения ksLineBuildingType, равного ksLBTParallel и ksLBTPerpendicular.

BuildingType – Способ построения сегмента кривой

Интерфейс...

Тип данных: из перечисления ksLineBuildingType

Синтаксис Automation:

```
BuildingType = iObject.BuildingType;                  Получить свойство (* )
iObject.BuildingType = BuildingType;                  Установить свойство (* )
```

```
BuildingType = Получить свойство (**)
iObject.GetBuildingType();
iObject.SetBuildingType(BuildingType); Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_BuildingType Получить свойство
(&BuildingType)
iObject->put_BuildingType Установить свойство
(BuildingType)
```

Примечание:

1. Для сплайна используются только константы ksLBTByPoint и ksLBTByPoint3DParams.
2. В случае задания ksLBTByPoint3DParams начинают работать свойства PointType и PointParameters.

Index – Индекс вершины в массиве вершин кривой

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Index = iObject.Index; Получить свойство (* )
Index = iObject.GetIndex(); Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Index Получить свойство
(&Index)
```

Примечание:

Свойство доступно только для чтения.

PointParameters – Интерфейс параметров точки

Интерфейс...

Тип данных: Указатель на интерфейс параметров вершины ломанной IKompasAPIObject

Синтаксис Automation:

```
PointParameters = Получить свойство (* )
iObject.PointParameters( Index );
PointParameters = Получить свойство (**)
iObject.GetPointParameters( Index );
```

Синтаксис COM:

iObject->get_PointParameters(Index, &PointParameters)	Получить свойство

Входные параметры:

Index	- индекс вершины.

Примечание:

1. Свойство доступно только для чтения.
2. В зависимости от типа PointType интерфейс параметров должен приводится к одному из следующих вариантов:

   ksPDisplace - IPoint3DParamDisplace - Интерфейс параметров пространственной точки, заданной по смещению от опорного объекта.

   ksPIntersect - IPoint3DParamIntersect - Интерфейс параметров пространственной точки, заданной на пересечении опорных объектов.

   ksPCenter - IPoint3DParamCenter - Интерфейс параметров пространственной точки, заданной в центре опорного объекта,

   ksPCurve - IPoint3DParamCurve - Интерфейс параметров пространственной точки, заданной на кривой со смещением.

   ksPSurface - IPoint3DparamSurface - Интерфейс параметров пространственной точки, заданной на поверхности.

   ksPProjection - IPoint3DParamProjection Интерфейс параметров пространственной точки, заданной проецированием.

PointType – Способ построения точки

Интерфейс...

Тип данных: Из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointType = iObject.PointType;	Получить свойство (*)
iObject.PointType = PointType;	Установить свойство (*)
PointType = iObject.GetPointType();	Получить свойство (**)
iObject.SetPointType(PointType);	Установить свойство (**)

Синтаксис COM:

iObject->get_PointType (&PointType)	Получить свойство
iObject->put_PointType (PointType)	Установить свойство

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс вектора, задающего направление построения сегмента ломаной или сплайна.
2. Свойство доступно только для чтения.

Vertex – Вершина кривой

Интерфейс...

Тип данных: указатель на интерфейс модельного объекта IModelObject.

Синтаксис Automation:

Vertex = iObject.Vertex;	Получить свойство (*)
Vertex = iObject.GetVertex();	Получить свойство (**)

Синтаксис COM:

iObject->get_Vertex	Получить свойство

Примечание:

Свойство доступно только для чтения.

ICurveVertexParam – методы

GetParamByDistance – Получить расстояние и радиус (для ломаной) или вес (для сплайна)

Интерфейс...

Синтаксис Automation:

BOOL GetParamByDistance(double * Distance,
double * Value);

Синтаксис Automation:

HRESULT GetParamByDistance([out] double * Distance,
[out] double * Value,
[out, retval] VARIANT_BOOL * PVal);

Выходные параметры:

Distance	- расстояние до предыдущей вершины,
value	- радиус для точки ломаной и вес для сплайна.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Для вершины с индексом 0 расстояние выдается до вершины с индексом 1. Для остальных вершин расстояние выдается до вершины с индексом на 1 меньше, чем у данной вершины.
2. Получить расстояние можно для любого типа построения.
3. Установить расстояние можно, если тип построения ksLineBuildingType не равен ksLBTByPoint и не равен ksLBTByPoint3DParams.

GetParamVertex – Получить параметры вершины

Интерфейс...

Синтаксис Automation:

BOOL GetParamVertex(double * x,
double * y,
double * z,
double * value);

Синтаксис Automation:

HRESULT GetParamVertex([out] double * x,
[out] double * y,
[out] double * z,
[out] double * value,
[out, retval] VARIANT_BOOL * PVal);

Выходные параметры:

x, y, z	- пространственные координаты вершины,
value	- радиус для точки ломаной и вес для сплайна.

Возвращаемое значение:

TRUE	- в случае удачного завершения,

FALSE - в случае неудачи.

SetParamByDistance – Установить расстояние, радиус (ломаная) или вес (сплайн)

Интерфейс...

Синтаксис Automation:

double Distance,
double Value);

HRESULT SetParamByDistance([in] double Distance,
double Value,
[out, retval] VARIANT_BOOL * PVal);

Distance - расстояние до предыдущей вершины,
value - радиус для точки ломаной и вес для сплайна.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание.

1. Нельзя установить расстояние для вершины с индексом 0.
2. Установить расстояние можно, если тип построения ksLineBuildingType не равен ksLBTByPoint и не равен ksLBTByPoint3DParams.

SetParamByVertex – Установить параметры вершины по указателю на вершину

Интерфейс...

Синтаксис Automation:

BOOL SetParamByVertex(LPDISPATCH Obj,
double Value);

Синтаксис Automation:

HRESULT SetParamByVertex([in] IModelObject * Obj,
[in] double Value,
[out, retval] VARIANT_BOOL * PVal);

Входные параметры:

obj - указатель на вершину,
value - радиус для точки ломаной и вес для сплайна.

Возвращаемое значение:

TRUE	- в случае удачного завершения.
FALSE	- в случае неудачи.

SetParamVertex – Установить параметры вершины

Интерфейс...

Синтаксис Automation:

BOOL SetParamVertex(double x,
double y,
double z,
double value);

Синтаксис Automation:

HRESULT SetParamVertex([in] double x,
[in] double y,
[in] double z,
[in] double value,
[out, retval] VARIANT_BOOL * PVal);

Входные параметры:

x, y, z	- пространственные координаты вершины,
value	- радиус для точки ломаной и вес для сплайна.

Возвращаемое значение:

TRUE	- в случае удачного завершения.
FALSE	- в случае неудачи.

Update – Обновить параметры вершин кривой начиная с этой

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис Automation:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IModelCurve3D

Объект 3D-кривая.

Иерархия:

IDispatch

IModelCurve3D

Описание:

Позволяет получить и изменить общие свойства для 3D-кривых.

Является дополнительным интерфейсом для IModelObject.

IModelCurve3D – свойства

EdgesStyle – Стиль линии

Интерфейс...

Тип данных: из перечисления ksCurveStyleEnum

Синтаксис Automation:

EdgesStyle = Object.EdgesStyle	Получить свойство (*)
Object.EdgesStyle = EdgesStyle	Установить свойство (*)
EdgesStyle = Object.GetEdgesStyle()	Получить свойство (**)
Object.SetEdgesStyle(EdgesStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_EdgesStyle(&EdgesStyle)	Получить свойство
Object.put_EdgesStyle(EdgesStyle)	Установить свойство

Свойство позволяет устанавливать и получать стиль линии.

VertexStyle – Стиль вершин

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

VertexStyle = Object.VertexStyle	Получить свойство (*)
Object.VertexStyle = VertexStyle	Установить свойство (*)
VertexStyle = Object.GetVertexStyle()	Получить свойство (**)
Object.SetVertexStyle(VertexStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_VertexStyle(&VertexStyle)	Получить свойство

Object.put_VertexStyle(VertexStyle)	Установить свойство

Свойство позволяет устанавливать и получать стиль вершин.

VertexVisible – Признак отображения вершин

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

VertexVisible = Object.VertexVisible	Получить свойство (*)
Object.VertexVisible = VertexVisible	Установить свойство (*)
VertexVisible = Object.GetVertexVisible()	Получить свойство (**)
Object.SetVertexVisible(VertexVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_VertexVisible(&EdgesCheck)	Получить свойство
Object.put_VertexVisible(EdgesCheck)	Установить свойство

Свойство позволяет устанавливать и получать признак отображения вершин.

IModelCurve3D – методы

GetVerticesParams – Получить параметры вершин кривой

Интерфейс...

Синтаксис Automation:

VARIANT GetVerticesParams();

Синтаксис COM:

HRESULT GetVerticesParams(VARIANT * Result);

Возвращаемое значение:

- массив параметров вершин - SafeArray вещественных чисел VT_ARRAY | VT_R8.

Интерфейс IMathCurve3D

Интерфейс математической кривой в трехмерном пространстве.

Иерархия:

IDispatch

IKompasAPIObject

IMathCurve3D

IMathCurve3D – свойства

Closed – Замкнутость кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Closed = Object.Closed;                Получить свойство (* )
Closed = Object.GetClosed();           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Closed( &Closed )           Получить свойство
```

Примечание:

Свойство доступно только для чтения.

CurveType – Получить тип кривой

Интерфейс...

Тип данных: из перечисления ksMathCurve3DTypeEnum

Синтаксис Automation:

```
CurveType = Object.CurveType;          Получить свойство (* )
CurveType = Object.GetCurveType();     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_CurveType( &CurveType )     Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Degenerate – Проверка вырожденности кривой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Degenerate = Object.Degenerate;        Получить свойство (* )
Degenerate = Object.GetDegenerate();   Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Degenerate( &Degenerate )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ParamMax – Получить значение параметра конечное

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ParamMax = Object.ParamMax;                   Получить свойство (* )
ParamMax = Object.GetParamMax();              Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ParamMax( &ParamMax)               Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ParamMin – Получить значение параметра начальное

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ParamMin = Object.ParamMin;                   Получить свойство (* )
ParamMin = Object.GetParamMin();              Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ParamMin( &ParamMin )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Periodic – Периодичность замкнутой кривой

Интерфейс...

Тип данных:BOOL

Синтаксис Automation:

```
Periodic = Object.Periodic;                Получить свойство (* )
Periodic = Object.GetPeriodic();           Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Periodic( &Periodic )           Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Radius – Радиус кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius;                    Получить свойство (* )
Radius = Object.GetRadius();               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius )               Получить свойство
```

Примечание:

Свойство доступно только для чтения.

IMathCurve3D – методы

CalculatePolygon – Получить полигон точек на кривой

Интерфейс...

Синтаксис Automation:

VARIANT CalculatePolygon(double Step);

Синтаксис COM:

HRESULT CalculatePolygon(double Step, VARIANT * Result);

Возвращаемое значение:

- массив VT_ARRAY | VT_R8 координат точек на кривой.

Входные параметры:

Step - шаг для расчета полигона.

Примечание:

1. Если шаг задан равным 0, то применяется шаг, используемый в системе КОМПАС, который рассчитывается с учетом габаритов детали.
2. Если шаг задан -1, то применяется шаг, используемый в системе КОМПАС при отрисовке кривых. Он рассчитывается с учетом следующих параметров:

- габариты детали и масштаба отображения модели,
- тип возвращаемого массива VT_ARRAY | VT_R8,
- значения координат в массиве лежат в последовательности x1, y1, z1, x2, y2, z2 ... xn, yn, zn.

GetCentre – Получить центр кривой

Интерфейс...

Синтаксис Automation:

BOOL GetCentre(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetCentre(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X, Y, Z	- координаты центра кривой.

GetWeightCentre – Получить центр тяжести кривой

Интерфейс...

Синтаксис Automation:

BOOL GetWeightCentre(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetWeightCentre(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X, Y, Z	- координаты центра тяжести кривой.

GetDerivativeT – Получить первую производную по T

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeT(double ParamT, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeT(double ParamT, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y, Z	- первая производная по T.

GetDerivativeTT – Получить вторую производную по T

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeTT(double ParamT, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeTT(double ParamT, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y, Z	- вторая производная по T.

GetDerivativeTTT – Получить третью производную по T

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeTTT(double ParamT, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeTTT(double ParamT, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y,Z	- третья производная по T.

GetGabarit – Выдать габарит кривой

Интерфейс...

Синтаксис Automation:

BOOL GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2) ;

Синтаксис COM:

HRESULT GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2, BOOL * Result) ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X1,Y1,Z1,X2,Y2,Z2	- координаты вершин габаритного параллелепипеда.

Примечание:

Координаты вершин параллелепипеда возвращаются в системе координат компонента.

GetLength – Получить длину кривой (ST_MIX_MM..ST_MIX_M единицы измерения)

Интерфейс...

Синтаксис Automation:

double GetLength(ksLengthUnitEnum BitVector) ;

Синтаксис COM:

HRESULT GetLength(ksLengthUnitEnum BitVector, double * Result) ;

Возвращаемое значение:

- длина кривой.

Входные параметры:

BitVector - единица измерения.

GetMetricLength – Метрическая длина кривой

Интерфейс...

Синтаксис Automation:

double GetMetricLength(double StartParam, double EndParam);

Синтаксис COM:

HRESULT GetMetricLength(double StartParam, double EndParam, double * Result) ;

Возвращаемое значение:

- длина кривой.

Входные параметры:

StartParam	- начальное значение параметра T в параметрическом представлении кривой,
EndParam	- конечное значение параметра T в параметрическом представлении кривой.

GetNormal – Получить нормаль

Интерфейс...

Синтаксис Automation:

BOOL GetNormal(double ParamT, double * X, double * Y, double * Z ;

Синтаксис COM:

HRESULT GetNormal(double ParamT, double * X, double * Y, double * Z, BOOL * Result ;

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

ParamT - значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y,Z - вектор нормали.

GetPoint – Получить точку на кривой

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(double ParamT, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(double ParamT, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

ParamT - значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y,Z - координаты точки на кривой.

GetTangentVector – Получить тангенциальный вектор (нормализованный)

Интерфейс...

Синтаксис Automation:

BOOL GetTangentVector(double ParamT, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetTangentVector(double ParamT, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

X, Y,Z	- касательный вектор.

NearPointProjection – Получить ближайшую проекцию точки на кривую

Интерфейс...

Синтаксис Automation:

BOOL NearPointProjection(double X, double Y, double Z, double * T, BOOL Ext);

Синтаксис COM:

HRESULT NearPointProjection(double X, double Y, double Z, double * T, BOOL Ext, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X, Y, Z	- координаты исходной пространственной точки,
Ext	- если проекция точки вне кривой,
TRUE	- проецировать на продолжение кривой,
FALSE	- найти ближайшую граничную точку кривой. Если проекция точки внутри кривой, значение этого параметра не учитывается.

Выходные параметры:

T	- параметрические координаты точки на кривой.

3D-объект

Интерфейс IModelObject

Базовый интерфейс для всех модельных объектов.

Иерархия:

IKompasAPIObject

IModelObject

Примечание:

1. Данный интерфейс является базовым для модельных объектов.
2. Интерфейс можно получить из коллекции модельных объектов IModelObjects с помощью свойства IModelObjects::Item.
3. Интерфейс может быть получен с помощью метода IUnknown QueryInterface от модельного объекта, например, IPart7, IFeature7.
4. Интерфейс может быть получен от интерфейса IModelContainer с помощью свойства IModelContainer::Objects из массива SAFEARRAY типа VT_ARRAY|VT_DISPATCH с преобразованием от интерфейса IDispatch к IModelObject с помощью метода IUnknown QueryInterface.
5. Интерфейс может быть получен от интерфейса IFeature7 с помощью свойства IFeature7::ModelObjects из массива SAFEARRAY типа VT_ARRAY|VT_DISPATCH с преобразованием от интерфейса IDispatch к IModelObject с помощью метода IUnknown QueryInterface.
6. Имеет дополнительный интерфейс IColorParam7, который можно получить с помощью метода IUnknown::QueryInterface.

IModelObject – свойства

Hidden – Состояние видимости объекта

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- объект скрыт,
FALSE	- объект показан.

Синтаксис Automation:

Hidden = iObject.Hidden	Получить свойство(*)
iObject.Hidden = Hidden	Установить свойство (*)

Hidden = iObject.GetHidden( )	Получить свойство (**)
iObject.SetHidden (Hidden)	Установить свойство (**)

Синтаксис COM:

iObject->get_Hidden (&Hidden)	Получить свойство
iObject->put_Hidden (Hidden)	Установить свойство

Примечание:

1. Позволяет получить и установить состояние видимости объекта.
2. Свойство вступает в силу после вызова метода IModelObject::Update.

ModelObjectType – Тип модельного объекта

Интерфейс...

Тип данных: указатель на интерфейс Obj3dType

Синтаксис Automation:

ModelObjectType = iObject.ModelObjectType	Получить свойство(*)
ModelObjectType = iObject.GetModelObjectType( )	Получить свойство (**)

Синтаксис COM:

iObject->get_ModelObjectType (&ModelObjectType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Name – Имя элемента трехмерной модели

Интерфейс...

Тип данных: строка

Синтаксис Automation:

Name = iObject.Name	Получить свойство(*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName( )	Получить свойство (**)
iObject.SetName(Name)	Установить свойство (**)

Синтаксис COM:

iObject->get_Name(&Name)	Получить свойство
iObject->put_Name(Name)	Установить свойство

Примечание:

1. Позволяет получить и установить имя трехмерной модели.
2. Свойство вступает в силу после вызова метода IModelObject::Update.
3. Не все объекты имеют имя, например, ребра.

Owner – Объект дерева, породивший этот объект

Интерфейс...

Тип данных: указатель на интерфейс IFeature7

Синтаксис Automation:

Owner = iObject.Owner	Получить свойство(*)
Owner = iObject.GetOwner( )	Получить свойство (**)

Синтаксис COM:

HRESULT Owner([out, retval] IFeature7** Result);	Получить свойство

Примечание:

Свойство доступно только для чтения.

Part – Компонент, владеющий элементом

Интерфейс...

Тип данных: указатель на интерфейс IPart7.

Синтаксис Automation:

Part = iObject.Part	Получить свойство(*)
Part = iObject.GetPart( )	Получить свойство (**)

Синтаксис COM:

```
HRESULT Part( [out, retval]          Получить свойство
IPart7** Result );
```

Примечание:

1. Возвращает указатель на интерфейс компонента, владеющий объектом.
2. Свойство доступно только для чтения.

Valid – Признак невырожденности объекта

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE                    - объект скрыт,
FALSE                   - объект показан.
```

Синтаксис Automation:

```
Valid = iObject.Valid          Получить свойство(* )
Valid = iObject.GetValid( )    Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Valid ( &Valid )          Получить свойство
```

Примечание:

1. Позволяет получить признак невырожденности объекта.
2. Свойство доступно только для чтения.

IModelObject – методы

Update – Обновить данные объекта

Интерфейс...

Синтаксис Automation:

BOOL Update( );

Синтаксис COM:

HRESULT Update ([out, retval] VARIANT_BOOL* pRes);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод Update необходимо вызвать для объекта, свойства которого были изменены (после вызова Set- свойств или методов интерфейса), чтобы эти изменения вступили в силу.
2. Метод Update вернёт TRUE в случае успеха и FALSE в случае неудачи.
3. Вызов Update сработает как при изменении одного свойства, так и при изменении группы свойств объекта.

Интерфейс IMeshObject3D

Интерфейс пространственного полигонального объекта.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject1

IMeshObject3D

Интерфейс можно получить, выполнив QueryInterface для объекта IModelObject с типом o3d_MeshObject3D

IMeshObject3D – методы

InitByObjects – Создать по объектам

Интерфейс...

Синтаксис Automation:

BOOL InitByObjects(VARIANT Objects);

Синтаксис COM :

HRESULT InitByObjects(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Objects	- объект или массив объектов (VARIANT типа VT_DISPATCH или VT_ARRAY I VT_DISPATCH),

Интерфейс IModelObject1

Интерфейс 3D объекта.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject1

Примечание:

Интерфейс можно получить у интерфейса IModelObject с помощью метода IUnknown::QueryInterface.

IModelObject1 – свойства

Childrens – Потомки объекта

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Childrens = Object.Childrens( RelType );        Получить свойство (* )
Childrens = Object.GetChildrens( RelType );     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Childrens( RelType, &Childrens )                    Получить свойство
```

Параметры:

```
ksRelationTypeEnum RelType        - тип родственных отношений.
```

Примечание:

1. Свойство доступно только для чтения.
2. Свойство возвращает массив VT_ARRAY | VT_DISPATCH объектов-потомков данного объекта.

ConnectedWithInitialEmbodiment – Отменить/ восстановить связь объекта зависимого исполнения с соответствующим объектом исходного исполнения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ConnectedWithInitialEmbodiment = Object.ConnectedWithInitialEmbodiment	Получить свойство (*)
Object.ConnectedWithInitialEmbodiment = ConnectedWithInitialEmbodiment	Установить свойство (*)
ConnectedWithInitialEmbodiment = Object.GetConnectedWithInitialEmbodiment()	Получить свойство (**)
Object.SetConnectedWithInitialEmbodiment(ConnectedWithInitialEmbodiment)	Установить свойство (**)

Синтаксис COM:

Object.get_ConnectedWithInitialEmbodiment(&ConnectedWithInitialEmbodiment)	Получить свойство
Object.put_ConnectedWithInitialEmbodiment(ConnectedWithInitialEmbodiment)	Установить свойство

Editable – Признак редактирования

Интерфейс...

Тип данных: из перечисления ksEditableStateEnum

Синтаксис Automation:

Editable = Object.Editable	Получить свойство (*)
Object.Editable = Editable	Установить свойство (*)
Editable = Object.GetEditable()	Получить свойство (**)
Object.SetEditable(Editable)	Установить свойство (**)

Синтаксис COM:

Object.get_Editable(&Editable)	Получить свойство
Object.put_Editable(Editable)	Установить свойство

HiddenEx – Состояние видимости объекта

Интерфейс...

Тип данных: из перечисления ksVisibleStateEnum

Синтаксис Automation:

```
HiddenEx = Object.HiddenEx                  Получить свойство (* )
Object.HiddenEx = HiddenEx                  Установить свойство (* )
HiddenEx = Object.GetHiddenEx()             Получить свойство (**)
Object.SetHiddenEx( HiddenEx )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HiddenEx( &HiddenEx )            Получить свойство
Object.put_HiddenEx(  HiddenEx )            Установить свойство
```

IsEditableObject – Признак текущей доступности редактирования объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsEditableObject = Object.IsEditableObject;   Получить свойство (* )
IsEditableObject                          =   Получить свойство (**)
Object.GetIsEditableObject();
```

Синтаксис COM:

```
Object.get_IsEditableObject(                  Получить свойство
&IsEditableObject )
```

Примечание:

Свойство доступно только для чтения.

IsExternalObject – Объект является внешним по отношению к текущему редактируемому контексту

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsExternalObject = Object.IsExternalObject    Получить свойство (* )
IsExternalObject                          =   Получить свойство (**)
Object.GetIsExternalObject()
```

Синтаксис COM:

```
Object.get_IsExternalObject(                  Получить свойство
&IsExternalObject )
```

Примечание:

Свойство доступно только для чтения.

LayerNumber – Номер слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

LayerNumber = Object.LayerNumber	Получить свойство (*)
Object.LayerNumber = LayerNumber	Установить свойство (*)
LayerNumber = Object.GetLayerNumber()	Получить свойство (**)
Object.SetLayerNumber(LayerNumber)	Установить свойство (**)

Синтаксис COM:

Object.get_LayerNumber(&LayerNumber)	Получить свойство
Object.put_LayerNumber(LayerNumber)	Установить свойство

Links – Ссылки на опорные примитивы объекта

Интерфейс...

Тип данных: VARIANT массив

Синтаксис Automation:

Links = Object.Links;	Получить свойство (*)
Links = Object.GetLinks();	Получить свойство (**)

Синтаксис COM:

Object.get_Links(&Links)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MathObject – Математический объект

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

MathObject = Object.MathObject;	Получить свойство (*)
MathObject = Object.GetMathObject();	Получить свойство (**)

Синтаксис COM:

Object.get_MathObject(&MathObject)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Parents– Родители объекта

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Parents = Object.Parents(RelType);	Получить свойство (*)
Parents = Object.GetParents(RelType);	Получить свойство (**)

Синтаксис COM:

Object.get_Parents(RelType, &Parents)	Получить свойство

Параметры:

ksRelationTypeEnum RelType	тип родственных отношений

Примечание:

1. Свойство доступно только для чтения.
2. Свойство возвращает массив VT_ARRAY I VT_DISPATCH объектов-родителей данного объекта.

Projected – Признак проецирования

Интерфейс...

Тип данных: из перечисления ksProjectionOptionEnum

Синтаксис Automation:

Projected = Object.Projected	Получить свойство (*)
Object.Projected = Projected	Установить свойство (*)
Projected = Object.GetProjected()	Получить свойство (**)

Object.SetProjected(Projected) Установить свойство (**)

Синтаксис COM:

Object.get_Projected(&Projected) Получить свойство
Object.put_Projected(Projected) Установить свойство

IModelObject1 – методы

IsMyLink – Является ли примитив опорным для объекта

Интерфейс...

Синтаксис Automation:

```
BOOL IsMyLink( ksRelationTypeEnum RelType,
        LPDISPATCH Primitive );
```

Синтаксис COM:

```
HRESULT IsMyLink( ksRelationTypeEnum RelType,
        IModelObject * Primitive,
        BOOL * Result );
```

Возвращаемое значение:

TRUE - примитив является опорным.

Входные параметры:

RelType - тип родственных отношений,
Primitive - проверяемый примитив.

Метод позволяет проверить, является ли примитив опорным для объекта 3D (например, ребро кинематической операции опорным для операции скругления).

Reset – Сбросить заданные параметры

Интерфейс...

Синтаксис Automation:

```
BOOL Reset();
```

Синтаксис COM:

```
HRESULT Reset( BOOL * Result );
```

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IUserParameters

Интерфейс объекта, создаваемого библиотекой.

Иерархия:

IDispatch

IUserParameters

Является дополнительным интерфейсом для интерфейса IUserObject3D**.**

IUserParameters – свойства

Command – Номер команды

Интерфейс...

Тип данных: long

Синтаксис Automation:

Command = Object.Command	Получить свойство (*)
Object.Command = Command	Установить свойство (*)
Command = Object.GetCommand()	Получить свойство (**)
Object.SetCommand(Command)	Установить свойство (**)

Синтаксис COM:

Object.get_Command(&Command)	Получить свойство
Object.put_Command(Command)	Установить свойство

Примечание:

Свойство позволяет устанавливать команду редактирования библиотеки.

LibraryFileName – Имя файла библиотеки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Примечание:

Свойство позволяет устанавливать имя файла библиотеки.

LibraryName – Имя библиотеки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

Примечание:

Свойство позволяет устанавливать имя библиотеки.

ObjectID – Идентификатор объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ObjectID = Object.ObjectID	Получить свойство (*)
Object.ObjectID = ObjectID	Установить свойство (*)
ObjectID = Object.GetObjectID()	Получить свойство (**)
Object.SetObjectID(ObjectID)	Установить свойство (**)

Синтаксис COM:

Object.get_ObjectID(&ObjectID)	Получить свойство
Object.put_ObjectID(ObjectID)	Установить свойство

UserParams – Пользовательские параметры

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

UserParams = Object.UserParams	Получить свойство (*)

Object.UserParams = UserParams	Установить свойство (*)
UserParams = Object.GetUserParams()	Получить свойство (**)
Object.SetUserParams(UserParams)	Установить свойство (**)

Синтаксис COM:

Object.get_UserParams(&UserParams)	Получить свойство
Object.put_UserParams(UserParams)	Установить свойство

Значение свойства:

Принимает\возвращает безопасный массив SAFEARRAY типа VT_ARRAY | VT_UI1.

Примечание:

Свойство используется для сохранения/получения пользовательских параметров в виде безопасного массива SAFEARRAY байтов.

Компоненты

Интерфейс IBodyRepositions

Интерфейс коллекции операций перепозиционирования тела, поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IBodyRepositions

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::BodyRepositions.

КОМПАС версия v18

IBodyRepositions – свойства

BodyReposition – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IBodyReposition.

Синтаксис Automation:

BodyReposition Object.BodyReposition(Index);	=	Получить свойство (*)
BodyReposition Object.GetBodyReposition(Index);	=	Получить свойство (**)

Синтаксис COM:

Object.get_BodyReposition(&BodyReposition)	Index,	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IBodyRepositions – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IBodyReposition * * Result);

Возвращаемое значение:

	- указатель на интерфейс IBodyReposition.

Примечания:

1. Метод позволяет создать новый интерфейс операции перепозиционирования тела, поверхности.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IBodyReposition

Интерфейс перепозиционирования тела, поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IBodyReposition

Данный интерфейс можно получить с помощью метода коллекции операций перепозиционирования тела, поверхности IBodyRepositions::Add или свойства IBodyRepositions::BodyReposition.

IBodyReposition – свойства

CopyBoby – Копировать тело

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CopyBoby = Object.CopyBoby	Получить свойство (*)
Object.CopyBoby = CopyBoby	Установить свойство (*)
CopyBoby = Object.GetCopyBoby()	Получить свойство (**)
Object.SetCopyBoby(CopyBoby)	Установить свойство (**)

Синтаксис COM:

Object.get_CopyBoby(&CopyBoby)	Получить свойство
Object.put_CopyBoby(CopyBoby)	Установить свойство

Position – Параметры смещения

Интерфейс...

Тип данных: Указатель на интерфейс ILocalCoordinateSystem.

Синтаксис Automation:

Position = Object.Position;	Получить свойство (*)
Position = Object.GetPosition();	Получить свойство (**)

Синтаксис COM:

Object.get_Position(&Position)	Получить свойство

Примечание:

Свойство доступно только для чтения.

RepositionBody – Перемещаемое тело или поверхность

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject.

Синтаксис Automation:

RepositionBody = Object.RepositionBody	=	Получить свойство (*)
Object.RepositionBody = RepositionBody	=	Установить свойство (*)
RepositionBody = Object.GetRepositionBody()	=	Получить свойство (**)
Object.SetRepositionBody(RepositionBody)		Установить свойство (**)

Синтаксис COM:

Object.get_RepositionBody(&RepositionBody)	Получить свойство
Object.put_RepositionBody(RepositionBody)	Установить свойство

RepositionCentre – Точка центра смещения

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

RepositionCentre = Object.RepositionCentre	=	Получить свойство (*)
Object.RepositionCentre = RepositionCentre	=	Установить свойство (*)
RepositionCentre = Object.GetRepositionCentre()	=	Получить свойство (**)
Object.SetRepositionCentre(RepositionCentre)		Установить свойство (**)

Синтаксис COM:

Object.get_RepositionCentre(&RepositionCentre)	Получить свойство
Object.put_RepositionCentre(RepositionCentre)	Установить свойство

Интерфейс IChooseObjects

Интерфейс области применения для тел и компонентов документа в операции.

Иерархия:

IDispatch

IChooseObjects

Данный интерфейс позволяет выбирать тела и компоненты, участвующие в операции.

КОМПАС версия v18

IChooseObjects – свойства

ChooseBodies – Заменить массив тел (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ChooseBodies = Object.ChooseBodies	Получить свойство (*)
Object.ChooseBodies = ChooseBodies	Установить свойство (*)
ChooseBodies = Object.GetChooseBodies()	Получить свойство (**)
Object.SetChooseBodies(ChooseBodies)	Установить свойство (**)

Синтаксис COM:

Object.get_ChooseBodies(&ChooseBodies)	Получить свойство
Object.put_ChooseBodies(ChooseBodies)	Установить свойство

Примечание.

Свойство позволяет получать и устанавливать тело или массив тел, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY тел LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ChooseParts – Массив вставок (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ChooseParts = Object.ChooseParts	Получить свойство (*)
Object.ChooseParts = ChooseParts	Установить свойство (*)

ChooseParts Object.GetChooseParts()	=	Получить свойство (**)
Object.SetChooseParts(ChooseParts)		Установить свойство (**)

Синтаксис COM:

Object.get_ChooseParts(&ChooseParts)	Получить свойство
Object.put_ChooseParts(ChooseParts)	Установить свойство

Примечание.

Свойство позволяет получать и устанавливать компонент или массив компонентов, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY компонентов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ChoosePartsType – Область применения. Объекты

Интерфейс...

Тип данных: Значение из перечисления ksChoosePartsType.

Синтаксис Automation:

ChoosePartsType Object.ChoosePartsType	=	Получить свойство (*)
Object.ChoosePartsType ChoosePartsType	=	Установить свойство (*)
ChoosePartsType Object.GetChoosePartsType()	=	Получить свойство (**)
Object.SetChoosePartsType(ChoosePartsType)		Установить свойство (**)

Синтаксис COM:

Object.get_ChoosePartsType(&ChoosePartsType)	Получить свойство
Object.put_ChoosePartsType(ChoosePartsType)	Установить свойство

ChooseType – Область применения. Группы объектов

Интерфейс...

Тип данных: Значение из перечисления ksChooseType.

Синтаксис Automation:

ChooseType = Object.ChooseType	Получить свойство (*)
Object.ChooseType = ChooseType	Установить свойство (*)
ChooseType = Object.GetChooseType()	Получить свойство (**)
Object.SetChooseType(ChooseType)	Установить свойство (**)

Синтаксис COM:

Object.get_ChooseType(&ChooseType)	Получить свойство
Object.put_ChooseType(ChooseType)	Установить свойство

Интерфейс IParts7

Интерфейс коллекции компонентов 3D документа.

Иерархия:

IKompasAPIObject

IModelObjects

IParts7

Примечание:

1. Данный интерфейс можно получить от интерфейса IPart7, используя свойство IPart7::Parts.
2. Данный интерфейс можно получить от интерфейса IModelObjects, используя метод IUnknown QueryInterface.
3. От интерфейса 3D-документа IKompasDocument3D может быть получен верхний компонент IKompasDocument3D::TopPart, а от полученного верхнего компонента IPart7 может быть получена коллекция компонентов IParts7 с помощью свойства IPart7::Parts. Элементы коллекции-компоненты могут быть получены у коллекции с помощью свойства IParts7::Part. Либо с помощью свойства IModelObjects::Item как объекты типа IModelObject с последующим приведением к типу IPart7 через QueryInterface. От каждого элемента коллекции-компонента может быть получена вложенная коллекция компонентов с помощью свойства IPart7::Parts. И так далее, пока имеются вложенные коллекции компонентов.

IParts7 – свойства

Part – Компонент, заданный по индексу или по имени

Интерфейс...

Тип данных: указатель на интерфейс IPart7

Синтаксис Automation:

Part = iObject.Part (Index)	Получить свойство (*)
Part = iObject.GetPart (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Part(index, &Part)	Получить свойство

Входные параметры:

Index (Variant)	VT_BSTR - имя компонента, VT_I4 - индекс компонента.

Примечание:

Свойство доступно только для чтения.

IParts7 – методы

Add – Создать новый компонент и добавить его в документ и коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(LPCTSTR FileName, LPDISPATCH plane);

Синтаксис COM:

HRESULT Add([in] BSTR FileName, [in] IModelObject * Plane, [out, retval] IPart7 ** Result);

Входные параметры:

FileName	- имя файла детали, создаваемой в сборке,
Plane	- указатель на интерфейс плоского объекта, к которому приклеивается деталь.

Возвращаемое значение:

указатель на интерфейс IPart7	- в случае успешного завершения,
NULL	- в случае неудачи.

AddFromFile – Добавить существующий компонент из файла или из библиотеки моделей в документ и в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddFromFile (LPCTSTR FileName, BOOL ExternalFile, BOOL Redraw);

Синтаксис COM:

HRESULT AddFromFile ([in] BSTR FileName, [in, defaultvalue(FALSE)] VARIANT_BOOL ExternalFile, [in, defaultvalue(TRUE)] VARIANT_BOOL Redraw, [out, retval] IPart7 ** Result);

Входные параметры:

FileName	- имя файла, из которого будет вставлен компонент,
ExternalFile	- признак сохранения связи с файлом-источником: TRUE - вставка со ссылкой на внешний файл, FALSE - вставка "телом", (без сохранения ссылки на источник),
Redraw	- признак перестроения документа после вставки компонента: TRUE - перестроить документ, FALSE - не перестраивать.

Возвращаемое значение:

указатель на интер-фейс IPart7	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечания:

При вставке из файла указывается полный путь к файлу. При вставке из библиотеки моделей указывается полный путь к файлу библиотеки и путь внутри библиотеки моделей, например, "C:\\standard.l3dlКрепежные элементыlБолтыlБолт_1".

CreateDocument – Создать локальную деталь и добавить в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateDocument(DocumentTypeEnum Type, BOOL Local, BOOL Redraw);

Синтаксис COM:

HRESULT CreateDocument(DocumentTypeEnum Type, BOOL Local, BOOL Redraw, IPart7 * * Result);

Возвращаемое значение:

- указатель на интерфейс вставки IPart7.

Входные параметры:

Type	- тип документа вставки,
Local	- признак локальной детали,
Redraw	- TRUE для обновления окна документа.

CreateDocumentEx – Создать локальную деталь и добавить в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateDocumentEx(BSTR StartFileName, DocumentTypeEnum Type, BOOL Local, BOOL Redraw);

Синтаксис COM:

HRESULT CreateDocumentEx(BSTR StartFileName, DocumentTypeEnum Type, BOOL Local, BOOL Redraw, IPart7 * * Result);

Возвращаемое значение:

- указатель на интерфейс вставки IPart7.

Входные параметры:

StartFileName	- имя файла по умолчанию,
Type	- тип документа вставки,
Local	- признак локальной детали,
Redraw	- TRUE для обновления окна документа.

Интерфейс IPart7

Интерфейс компонента 3D документа.

Иерархия:

IKompasAPIObject

IModelObject

IPart7

IFeature7

ISourcePart7Params

ISheetMetalContainer

IModelContainer

IMassInertiaParam7

ISymbols3DContainer

IAuxiliaryGeomContainer

IModelObjectNotifyResult

IPropertyKeeper

ISurfaceContainer

Описание:

Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительные интерфейсы компонента:

- интерфейс объекта Дерева построения IFeature7,
- интерфейс параметров компонента в источнике ISourcePart7Params.
- интерфейс контейнера объектов гибки ISheetMetalContainer,
- интерфейс контейнера трехмерных объектов IModelContainer,
- интерфейс массово-центровочных характеристик IMassInertiaParam7.
- интерфейс контейнера условных обозначений в 3D ISymbols3DContainer,
- интерфейс контейнера объектов вспомогательной геометрии IAuxiliaryGeomContainer,
- интерфейс контейнера поверхностей ISurfaceContainer,
- интерфейс результатов редактирования 3D объекта IModelObjectNotifyResult,
- интерфейс получения/редактирования значения свойств IPropertyKeeper,
- интерфейс контейнера объектов измерений и диагностики IMeasurementContainer.

Примечание.

1. Компонент может быть получен от интерфейса коллекции компонентов IParts7.
2. Компонент может быть получен от интерфейса IModelObject или IFeature7,используя метод IUnknown QueryInterface.
3. Верхний компонент может быть получен от интерфейса 3D-документа IKompasDocument3D с помощью свойства IKompasDocument3D::TopPart.
4. От интерфейса 3D-документа IKompasDocument3D может быть получен верхний компонент IKompasDocument3D::TopPart, а от полученного верхнего компонента IPart7 может быть получена коллекция компонентов IParts7 с помощью свойства IPart7::Parts. Элементы коллекции-компоненты могут быть получены у коллекции с помощью свойства IParts7::Part. Либо с помощью свойства IModelObjects::Item как объекты типа IModelObject с последующим приведением к типу IPart7 через QueryInterface. От каждого элемента коллекции-компонента может быть получена вложенная коллекция компонентов с помощью свойства IPart7::Parts. И так далее, пока имеются вложенные коллекции компонентов.

IPart7 – свойства

CreateSpcObjects – Создавать объекты спецификации

Справка системы КОМПАС... Дополнительные параметры вставки компонента

kompas.chm::/ADD_COMPONENT_FROM_FILE.htm
Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateSpcObjects = Object.CreateSpcObjects	Получить свойство (*)
Object.CreateSpcObjects = CreateSpcObjects	Установить свойство (*)
CreateSpcObjects = Object.GetCreateSpcObjects()	Получить свойство (**)
Object.SetCreateSpcObjects(CreateSpcObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_CreateSpcObjects(&CreateSpcObjects)	Получить свойство
Object.put_CreateSpcObjects(CreateSpcObjects)	Установить свойство

Примечание:

Объекты удаляются при перестроении спецификации, если у компонента выключен данный флаг.

DefaultObject – Получить предопределенные элементы (плоскости, ось и СК)

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DefaultObject = Object.DefaultObject(Type)	Получить свойство (*)
DefaultObject = Object.GetDefaultObject(Type)	Получить свойство (**)

Синтаксис COM:

Object.get_DefaultObject(&Type)	Получить свойство

Входной параметр:

Type	- тип объекта - константа из перечисления ksObj3dTypeEnum.

Типы объектов (Type):

o3d_planeXOY	1	плоскость XOY,
o3d_planeXOZ	2	плоскость XOZ,
o3d_planeYOZ	3	плоскость YOZ,
o3d_pointCS	4	точка начала системы координат,
o3d_axisOX	71	ось OX,
o3d_axisOY	72	ось OY,
o3d_axisOZ	73	ось OZ.

Density – Плотность компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

Density = iObject.Density	Получить свойство (*)
Density = iObject.GetDensity()	Получить свойство (**)

Синтаксис COM:

iObject->get_Density(&Density)	Получить свойство

Значения свойства:

Плотность компонента (г/куб.мм).

Примечание:

1. Свойство доступно только для чтения.
2. Для установки плотности необходимо использовать метод IPart7::SetMaterial.

Detail – Является ли компонент деталью

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Detail = iObject.Detail	Получить свойство (*)
Detail = iObject.GetDetail()	Получить свойство (**)

Синтаксис COM:

iObject->get_Detail (&Detail)	Получить свойство

Значения свойства:

TRUE	- компонент является деталью,
FALSE	- компонент является сборкой.

Примечание:

Свойство доступно только для чтения.

DummyEmbodimentIndex – Индекс текущего исполнения для макета

Интерфейс...

Тип данных: long

Синтаксис Automation:

DummyEmbodimentIndex = Object.DummyEmbodimentIndex	=	Получить свойство (*)
DummyEmbodimentIndex = Object.GetDummyEmbodimentIndex()	=	Получить свойство (**)

Синтаксис COM:

Object.get_DummyEmbodimentIndex(&DummyEmbodimentIndex)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DummyFileName – Имя файла для макета

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DummyFileName = Object.DummyFileName	Получить свойство(*)
Object.DummyFileName = DummyFileName	Установить свойство (*)
DummyFileName = Object.GetDummyFileName()	Получить свойство (**)
Object.SetDummyFileName(DummyFileName)	Установить свойство (**)

Синтаксис COM:

Object.get_DummyFileName(&DummyFileName)	Получить свойство
Object.put_DummyFileName(DummyFileName)	Установить свойство

FileName – Имя файла компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

FileName = iObject.FileName	Получить свойство (*)
iObject.FileName = FileName	Установить свойство (*)
FileName = iObject.GetFileName()	Получить свойство (**)
iObject.SetFileName (FileName)	Установить свойство (**)

Синтаксис COM:

iObject->get_FileName (&FileName)	Получить свойство
iObject->put_FileName (FileName)	Установить свойство

Примечание:

1. Имя файла можно установить только у компонента, вставленного в сборку.
2. Свойство вступает в силу после вызова метода IModelObject::Update.

Fixed – Состояние фиксации компонента

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Fixed = iObject.Fixed	Получить свойство (*)
iObject.Fixed = Fixed	Установить свойство (*)
Fixed = iObject.GetFixed()	Получить свойство (**)
iObject.SetFixed(Fixed)	Установить свойство (**)

Синтаксис COM:

iObject->get_Fixed(&Fixed)	Получить свойство
iObject->put_Fixed(Fixed)	Установить свойство

Значения свойства:

TRUE	- компонент зафиксирован,
FALSE	- компонент не зафиксирован.

Примечание:

Свойство вступает в силу после вызова метода IModelObject::Update.

HatchParam – Параметры штриховки

Интерфейс...

Тип данных: Указатель на интерфейс IHatchParam

Синтаксис Automation:

HatchParam = Object.HatchParam	Получить свойство (*)
HatchParam = Object.GetHatchParam()	Получить свойство (**)

Синтаксис COM:

Object.get_HatchParam(&HatchParam)	Получить свойство

Свойство позволяет получать интерфейс параметров штриховки компонента.

Примечание:

Свойство доступно только для чтения.

InheritExclude – Признак исключения из расчета задан в файле-источнике. TRUE-Задан в источнике, FALSE-Переопределен у вставки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InheritExclude = Object.InheritExclude	Получить свойство (*)
Object.InheritExclude = InheritExclude	Установить свойство (*)
InheritExclude = Object.GetInheritExclude()	Получить свойство (**)
Object.SetInheritExclude(InheritExclude)	Установить свойство (**)

Синтаксис COM:

Object.get_InheritExclude(&InheritExclude)	Получить свойство
Object.put_InheritExclude(InheritExclude)	Установить свойство

InstanceCount – Количество вставок компонента

Интерфейс...

Пример...

Тип данных: long

Синтаксис Automation:

Count = iObject.InstanceCount(iPart7)	Получить свойство (*)
Count = iObject.GetInstanceCount(iPart7)	Получить свойство (**)

Синтаксис COM:

iObject->get_InstanceCount(iPart7, &Count)	Получить свойство

Входные параметры:

iPart7	- указатель на компонент или NULL.

Примечание:

1. Свойство доступно только для чтения.
2. Метод работает для верхнего компонента сборки и для вставок подсборок. Если iPart7 == NULL, то возвращается общее количество компонентов, вставленных в данную подсборку.

IsBillet – Признак вставки заготовки детали

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsBillet = iObject.IsBillet	Получить свойство (*)
iObject.IsBillet = IsBillet	Установить свойство (*)
IsBillet = iObject.GetIsBillet()	Получить свойство (**)
iObject.SetIsBillet(IsBillet)	Установить свойство (**)

Синтаксис COM:

iObject->get_IsBillet(&IsBillet)	Получить свойство
iObject->put_IsBillet(IsBillet)	Установить свойство

IsLayoutGeometry – Признак компоновочной геометрии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Fixed = iObject.Fixed	Получить свойство (*)
iObject.Fixed = Fixed	Установить свойство (*)
Fixed = iObject.GetFixed()	Получить свойство (**)
iObject.SetFixed(Fixed)	Установить свойство (**)

Синтаксис COM:

iObject->get_Fixed(&Fixed)	Получить свойство
iObject->put_Fixed(Fixed)	Установить свойство

IsLocal – Локальная деталь

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsLocal = Object.IsLocal	Получить свойство (*)
Object.IsLocal = IsLocal	Установить свойство (*)
IsLocal = Object.GetIsLocal()	Получить свойство (**)
Object.SetIsLocal(IsLocal)	Установить свойство (**)

Синтаксис COM:

Object.get_IsLocal(&IsLocal)	Получить свойство
Object.put_IsLocal(IsLocal)	Установить свойство

LeftHandedCS – Признак левосторонней системы координат

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LeftHandedCS = Object.LeftHandedCS	Получить свойство (*)
Object.LeftHandedCS = LeftHandedCS	Установить свойство (*)

```
LeftHandedCS =                  Получить свойство (**)
Object.GetLeftHandedCS()
Object.SetLeftHandedCS(         Установить свойство (**)
LeftHandedCS )
```

Синтаксис COM:

```
Object.get_LeftHandedCS(        Получить свойство
&LeftHandedCS )
Object.put_LeftHandedCS(        Установить свойство
LeftHandedCS )
```

LoadState – Тип загрузки компонента

Интерфейс...

Тип данных: из перечисления ksLoadStateEnum

Синтаксис Automation:

```
LoadState =                     Получить свойство (* )
iObject.LoadState
LoadState =                     Получить свойство (**)
iObject.GetLoadState()
```

Синтаксис COM:

```
iObject-                        Получить свойство
>get_LoadState(&LoadState)
```

Примечание:

Свойство позволяет получить тип загрузки компонента ksLoadStateEnum.

Marking – Обозначение компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Marking = iObject.Marking       Получить свойство (* )
iObject.Marking = Marking       Установить свойство (* )
Marking = iObject.GetMarking()  Получить свойство (**)
iObject.SetMarking ( Marking)   Установить свойство (**)
```

Синтаксис COM:

iObject->get_Marking (&Marking)	Получить свойство
iObject->put_Marking (Marking)	Установить свойство

Примечание:

Свойство вступает в силу после вызова метода IModelObject::Update.

Mass – Масса компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

Mass = iObject.Mass	Получить свойство (*)
Mass = iObject.GetMass()	Получить свойство (**)

Синтаксис COM:

iObject->get_Mass (&Mass)	Получить свойство

Значения свойства:

	Масса компонента.

Примечание:

Свойство доступно только для чтения.

MateConstraints – Коллекция сопряжений

Интерфейс...

Тип данных: Указатель на интерфейс IMateConstraints3D

Синтаксис Automation:

MateConstraints = Object.MateConstraints	Получить свойство (*)
MateConstraints = Object.GetMateConstraints ()	Получить свойство (**)

Синтаксис COM:

Object.get_MateConstraints(&MateConstraints)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Material – Материал компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

Material = iObject.Material	Получить свойство (*)
Material = iObject.GetMaterial()	Получить свойство (**)

Синтаксис COM:

iObject->get_Material (&Material)	Получить свойство

Значения свойства:

	Материал компонента.

Примечание:

1. Свойство доступно только для чтения.
2. Для установки плотности необходимо использовать метод IPart7::SetMaterial.

Parts – Коллекция компонентов

Интерфейс...

Тип данных: указатель на интерфейс IParts7

Синтаксис Automation:

Parts = iObject.Parts	Получить свойство (*)
Parts = iObject.GetParts()	Получить свойство (**)

Синтаксис COM:

iObject->get_Parts (&Parts)	Получить свойство

Значения свойства:

	- Указатель на интерфейс коллекции компонентов.

Примечание:

Свойство доступно только для чтения.

PartsEx – Массив SAFEARRAY компонентов

Интерфейс...

Пример...

Тип данных: VARIANT

Синтаксис Automation:

Objects = iObject.PartsEx(ObjType)	Получить свойство (*)
Objects = iObject.GetPartsEx(ObjType)	Получить свойство (**)

Синтаксис COM:

iObject->get_PartsEx(&Objects)	Получить свойство

Входные параметры:

ObjType (VARIANT)	- Тип коллекции. В настоящее время может принимать значение константы из перечисления ksPart7CollectionTypeEnum.

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH). Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Placement – Получить локальную систему координат компонента

Интерфейс...

Тип данных: Указатель на интерфейс IPlacement3D

Синтаксис Automation:

Placement = iObject.Placement	Получить свойство (*)
Placement = iObject.GetPlacement()	Получить свойство (**)

Синтаксис COM:

iObject->get_Placement (&Placement)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ReadOnly – Тип доступа к компоненту

Интерфейс...

Тип данных: из перечисления ksPartAccessTypeEnum

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = Object.GetReadOnly()	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

Свойство позволяет устанавливать и получать тип доступа к компоненту.

Примечание:

Тип ksATDisable установить нельзя. Он возвращается в том случае, если смена доступа защищена паролем.

SpecRough – Неуказанная шероховатость 3D

Интерфейс...

Тип данных: Указатель на интерфейс ISpecRough3D

Синтаксис Automation:

SpecRough = Object.SpecRough	Получить свойство (*)
SpecRough = Object.GetSpecRough()	Получить свойство (**)

Синтаксис COM:

Object.get_SpecRough(&SpecRough)	Получить свойство

Примечание:

Свойство доступно только для чтения.

StaffVisible – Управление видимостью состава

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StaffVisible = Object.StaffVisible	Получить свойство (*)
Object.StaffVisible = StaffVisible	Установить свойство (*)
StaffVisible = Object.GetStaffVisible()	Получить свойство (**)
Object.SetStaffVisible(StaffVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_StaffVisible(&StaffVisible)	Получить свойство
Object.put_StaffVisible(StaffVisible)	Установить свойство

Свойство позволяет устанавливать и получать видимость состава у компонента, взятого в документ.

Standard – Признак стандартного компонента

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Standard = iObject.Standard	Получить свойство (*)
iObject.Standard = Standard	Установить свойство (*)
Standard = iObject.GetStandard()	Получить свойство (**)
iObject.SetStandard (Standard)	Установить свойство (**)

Синтаксис COM:

iObject->get_Standard (&Standard)	Получить свойство
iObject->put_Standard (Standard)	Установить свойство

Примечание:

Свойство вступает в силу после вызова метода IModelObject::Update.

ToleranceRecalcType – Способ пересчета

Интерфейс...

Тип данных: Значение из перечисления ksToleranceRecalcsEnum.

Синтаксис Automation:

ToleranceRecalcType = Object.ToleranceRecalcType	Получить свойство (*)
Object.ToleranceRecalcType = ToleranceRecalcType	Установить свойство (*)
ToleranceRecalcType = Object.GetToleranceRecalcType()	Получить свойство (**)
Object.SetToleranceRecalcType(ToleranceRecalcType)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceRecalcType(&ToleranceRecalcType)	Получить свойство
Object.put_ToleranceRecalcType(ToleranceRecalcType)	Установить свойство

VariableTable – Интерфейс таблицы переменных

Интерфейс...

Тип данных: указатель на интерфейс IVariableTable

Синтаксис Automation:

IVariableTable * VariableTable = Part7.VariableTable	Получить свойство (*)
IVariableTable * VariableTable = Part7.GetVariableTable()	Получить свойство (**)

Синтаксис COM:

HRESULT get_VariableTable (IVariableTable ** pRes)	Получить свойство

Примечание:

Свойство доступно только для чтения.

UniqueNum – Уникальный номер объекта типа

Интерфейс...

Тип данных: long

Синтаксис Automation:

UniqueNum = Object.UniqueNum(OType)	Получить свойство (*)
Object.UniqueNum(OType) = UniqueNum	Установить свойство (*)
UniqueNum = Object.GetUniqueNum(OType)	Получить свойство (**)
Object.SetUniqueNum(OType, UniqueNum)	Установить свойство (**)

Синтаксис COM:

Object.get_UniqueNum(OType, &UniqueNum)	Получить свойство
Object.put_UniqueNum(OType, UniqueNum)	Установить свойство

Входные параметры:

ksObj3dTypeEnum - OType - тип объекта.

UseDummy – Использовать макет

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseDummy = Object. UseDummy	Получить свойство (*)
Object. UseDummy = UseDummy	Установить свойство (*)
UseDummy = Object.Get UseDummy()	Получить свойство (**)
Object.Set UseDummy(UseDummy)	Установить свойство (**)

Синтаксис COM:

Object.get_ UseDummy(& UseDummy)	Получить свойство
Object.put_ UseDummy(UseDummy)	Установить свойство

UserFolders – Коллекция пользовательских директорий

Интерфейс...

Тип данных: Указатель на интерфейс IUserFolders.

Синтаксис Automation:

UserFolders = Object.UserFolders	Получить свойство (*)
UserFolders = Object.GetUserFolders()	Получить свойство (**)

Синтаксис COM:

Object.get_UserFolders(&UserFolders)	Получить свойство

Примечание:

Свойство доступно только для чтения.

UserToleranceRecalcId – Идентификатор пользовательского пересчета

Интерфейс...

Тип данных: long

Синтаксис Automation:

UserToleranceRecalcId = Object. UserToleranceRecalcId	Получить свойство (*)
Object. UserToleranceRecalcId = UserToleranceRecalcId	Установить свойство (*)
UserToleranceRecalcId = Object.Get UserToleranceRecalcId()	Получить свойство (**)
Object.Set UserToleranceRecalcId(UserToleranceRecalcId)	Установить свойство (**)

Синтаксис COM:

Object.get_ UserToleranceRecalcId(& UserToleranceRecalcId)	Получить свойство
Object.put_ UserToleranceRecalcId(UserToleranceRecalcId)	Установить свойство

UserToleranceRecalcName – Имя для пользовательского способа пересчета

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

UserToleranceRecalcName = Object. UserToleranceRecalcName	Получить свойство (*)
Object. UserToleranceRecalcName = UserToleranceRecalcName	Установить свойство (*)
UserToleranceRecalcName = Object.Get UserToleranceRecalcName()	Получить свойство (**)
Object.Set UserToleranceRecalcName(UserToleranceRecalcName)	Установить свойство (**)

Синтаксис COM:

Object.get_ UserToleranceRecalcName(& UserToleranceRecalcName)	Получить свойство
Object.put_ UserToleranceRecalcName(UserToleranceRecalcName)	Установить свойство

ZonesManager – Менеджер зон

Интерфейс...

Тип данных: Указатель на интерфейс IZonesManager.

Синтаксис Automation:

ZonesManager = Object.ZonesManager	Получить свойство (*)
ZonesManager = Object.GetZonesManager()	Получить свойство (**)

Синтаксис COM:

Object.get_ZonesManager(&ZonesManager)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

IPart7 – методы

AddVariable – Создать переменную

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddVariable(BSTR Name, double Value, BSTR Note);

Синтаксис COM:

HRESULT AddVariable(BSTR Name, double Value, BSTR Note, IVariable7 ** Result);

Входные параметры:

Name	- имя переменной,
Value	- значение переменной,
Note	- примечание.

Возвращаемое значение:

- указатель на интерфейс переменной IVariable7.

Метод позволяет добавить переменную в массив переменных и документ.

BeginEdit – Войти в режим редактирования компонента сборки

Интерфейс...

Синтаксис Automation:

LPDISPATCH BeginEdit(IOpenDocumentParam * Param);

Синтаксис COM:

HRESULT BeginEdit(IOpenDocumentParam * Param, IKompasDocument3D ** Result);

Возвращаемое значение:

Указатель на документ 3D IKompasDocument3D

Входные параметры:

Param - указатель на интерфейс параметров открытия документа IOpenDocumentParam.

ChangeObjectLinks – Заменить ссылки на объекты по всем операциям

Интерфейс...

Синтаксис Automation:

```
BOOL ChangeObjectLinks( VARIANT SourceObjs,
                        VARIANT DestObjs,
                        BOOL RebuildAll );
```

Синтаксис COM:

```
HRESULT ChangeObjectLinks( VARIANT SourceObjs,
                           VARIANT DestObjs,
                           BOOL RebuildAll,
                           BOOL * Result );
```

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

SourceObjs - массив исходных примитивов,

DestObjs	- массив новых примитивов,
RebuildAll	- перестроить объекты.

Метод позволяет заменить ссылки в объектах 3D модели с одних опорных примитивов на другие.

DestroySubassembly – Разрушить подсборку

Интерфейс...

Синтаксис Automation:

BOOL DestroySubassembly();

Синтаксис COM:

HRESULT DestroySubassembly(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

EndEdit – Завершить процесс редактирования компонента сборки

Интерфейс...

Синтаксис Automation:

BOOL EndEdit();

Синтаксис COM:

HRESULT EndEdit(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetBodyById – Возвращает тело, заданное по идентификатору

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetBodyById(long BodyId);

Синтаксис COM:

HRESULT GetBodyById(long BodyId, IBody7 ** Result);

Возвращаемое значение:

- указатель на интерфейс IBody7.

Входные параметры:

BodyId	- идентификатор тела.

GetDummyEmbodimentMarking – Обозначение исполнения для макета

Интерфейс...

Синтаксис Automation:

BSTR GetDummyEmbodimentMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer);

Синтаксис COM:

HRESULT GetDummyEmbodimentMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer, BSTR * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

MarkingType	- индекс части обозначения из перечисления ksVariantMarkingTypeEnum,
AddSystemDelimer	- добавлять системные разделители.

GetMaxSag – Максимально допустимый прогиб кривой или поверхности в соседних точках на расстоянии шага

Интерфейс...

Синтаксис Automation:

double GetMaxSag();

Синтаксис COM:

HRESULT GetMaxSag(double * Result);

Возвращаемое значение:

- значение максимально допустимого прогиба кривой.

FindObject – Найти объект в текущем документе по объекту из другого документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH FindObject(LPDISPATCH Obj, LPDISPATCH SourcePart);

Синтаксис COM:

HRESULT FindObject(IModelObject * Obj, IPart7 * SourcePart, IModelObject ** Result);

Входные параметры:

Obj	- указатель на объект,
SourcePart	- владелец объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

FindObjectsByPoint – Найти объекты по точке

Интерфейс...

Синтаксис Automation:

VARIANT FindObjectsByPoint(double X, double Y, double Z, BOOL FirstLevel);

Синтаксис COM:

HRESULT FindObjectsByPoint(double X, double Y, double Z, BOOL FirstLevel, VARIANT * Result);

Возвращаемое значение:

- массив объектов, содержащих заданную точку, в виде SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH.

Входные параметры:

X ,Y, Z	- координаты точки,
FirstLevel	- искать только в текущей модели.

GetOpenDocumentParam – Получить интерфейс параметров открытия документа для редактирования компонента сборки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetOpenDocumentParam();

Синтаксис COM:

HRESULT GetOpenDocumentParam(IOpenDocumentParam ** Result);

Возвращаемое значение:

Указатель на интерфейс параметров открытия документа IOpenDocumentParam.

IsVariableNameValid – Проверить допустимость создания новой переменной с данным именем

Интерфейс...

Синтаксис Automation:

BOOL IsVariableNameValid(BSTR Name);

Синтаксис COM:

BOOL IsVariableNameValid(BSTR Name);

Входные параметры:

Name	- имя переменной.

Возвращаемое значение:

TRUE	- имя допустимо,
FALSE	- имя недопустимо.

Load – Загрузить компонент

Интерфейс...

Синтаксис Automation:

BOOL Load(BOOL full);

Синтаксис COM:

HRESULT Load(BOOL full, BOOL * result);

Входные параметры:

full	- резерв.

Возвращаемое значение:

TRUE	- компонент успешно загружен,
FALSE	- компонент не загружен.

MirroringPlacement – Изменить систему координат детали на зеркальную

Интерфейс...

Синтаксис Automation:

BOOL MirroringPlacement(ksObj3dTypeEnum Axis);

Синтаксис COM:

HRESULT MirroringPlacement(ksObj3dTypeEnum Axis, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Axis	- ось, относительно которой делается симметрия.

OpenSourceDocument – Открытие документа-источника на редактирование

Интерфейс...

Синтаксис Automation:

LPDISPATCH OpenSourceDocument(IOpenDocumentParam * Param);

Синтаксис COM:

HRESULT OpenSourceDocument(IOpenDocumentParam * Param, IKompasDocument3D * * Result);

Возвращаемое значение:

Указатель на интерфейс документа IKompasDocument3D.

Входные параметры:

Param	- Указатель на интерфейс параметров открытия документа IOpenDocumentParam.

Версия Компас v18.1

RebuildModel – Перестроить модель

Интерфейс...

Синтаксис Automation:

BOOL RebuildModel(BOOL Redraw);

Синтаксис COM:

HRESULT RebuildModel(BOOL Redraw, BOOL * Result);

Входные параметры:

Redraw	- признак перестроения документа: TRUE - перестроить документ, FALSE - не перестраивать.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SaveAs – Сохранить файл компонента под другим именем

Интерфейс...

Синтаксис Automation:

BOOL SaveAs(BSTR PathName);

Синтаксис COM:

HRESULT SaveAs([in] BSTR PathName, [out, retval] VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SelectByPoint – Выделить объекты, содержащие точку

Интерфейс...

Синтаксис Automation:

VARIANT SelectByPoint(VARIANT Objects,
double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SelectByPoint(VARIANT Objects,
double X,

double Y,
double Z,
VARIANT * Result);

Входные параметры:

Objects	- массив SafeArray объектов типа VT_ARRAY I VT_DISPATCH,
x, y, z	- координаты точки.

Возвращаемое значение:

Массив SafeArray объектов типа VT_ARRAY I VT_DISPATCH.

Примечание

Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY I VT_DISPATCH.

SetDummyEmbodiment – Установить текущее исполнение для макета по имени или индексу

Интерфейс...

Синтаксис Automation:

BOOL SetDummyEmbodiment(VARIANT Index);

Синтаксис COM:

HRESULT SetDummyEmbodiment(VARIANT Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Index	- имя или индекс.

SetMaterial – Установить материал и плотность компонента

Интерфейс...

Синтаксис Automation:

BOOL SetMaterial(LPCTSTR Name, double density);

Синтаксис COM:

HRESULT SetMaterial([in] BSTR Name, [in] double Density, [out, retval] VARIANT_BOOL* PVal);

Входные параметры:

Name	- наименование материала,
Density	- плотность материала (г/куб.мм).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Компонент, у которого меняется материал, должен быть деталью.
2. Компонент не должен быть деталью из библиотеки моделей или стандартным элементом.
3. Метод используется в детали для верхнего компонента, в сборках - для вставленных деталей, открытых на редактирование.
4. Изменение материала и плотности вступает в силу после вызова метода IModelObject::Update.

TransferObjects – Перенести в СК

Интерфейс...

Синтаксис Automation:

BOOL TransferObjects(VARIANT objects,
LPDISPATCH Lcs,
BOOL holdPosition);

Синтаксис COM:

HRESULT TransferObjects(VARIANT objects,
ILocalCoordinateSystem * Lcs,
VARIANT_BOOL holdPosition,
BOOL * Result);

Входные параметры:

objects	- объект или список SafeArray объектов которые требуется перенести в другую систему координат,
Lcs	- указатель на интерфейс локальной системы координат ILocalCoordinateSystem,
holdPosition	- сохранять положение.

Возвращаемое значение:

TRUE - в случае успеха.

Примечание:

1. Метод позволяет перенести следующие объекты модели IModelObject:

- 3D точка,
- ломаная,
- сплайн,
- эскиз,
- импортированная поверхность,
- операция без истории,
- деталь заготовка,
- локальная система координат.

2. Если объекты нужно перенести в систему координат детали, нужно в качестве параметра Lcs передать NULL.
3. Если нужно перенести один объект, его можно передать в VARIANT-е как указатель на DISPATCH, тип VARIANT-а VT_DISPATCH.
4. Если нужно перенести несколько объектов, нужно передать в VARIANT-е как указатель на массив SafeArray тип VARIANT-а VT_ARRAY | VT_DISPATCH.
5. Если параметр holdPosition == TRUE, то положение объекта в документе не изменяется.
6. Если параметр holdPosition == FALSE, то положение объекта будет задано относительно новой системы координат.
7. Значение флага holdPosition может не учитываться при наличии ассоциативных связей объектов или особенностей создания объекта.

Unload – Выгрузить компонент

Интерфейс...

Синтаксис Automation:

BOOL Unload(BOOL full)

Синтаксис COM:

HRESULT Unload(BOOL full, BOOL * result);

Входные параметры:

full	TRUE - выгрузить полностью, FALSE - выгрузить частично.

Возвращаемое значение:

TRUE	- компонент успешно загружен,
FALSE	- компонент не загружен.

Примечание:

Метод позволяет осуществить выгрузку компонента. Для верхнего компонента не используется.

UnloadEx – Выгрузить

Интерфейс...

Синтаксис Automation:

BOOL UnloadEx(ksLoadStateEnum Type);

Синтаксис COM:

HRESULT UnloadEx(ksLoadStateEnum Type, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Type	- тип загрузки документа из перечисления ksLoadStateEnum.

UpdatePlacement – Установить локальную систему координат детали

Интерфейс...

Синтаксис Automation:

BOOL UpdatePlacement(BOOL Redraw);

Синтаксис COM:

HRESULT UpdatePlacement(BOOL Redraw, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Redraw - требуется перерисовка.

Интерфейс IEmbodiment

Интерфейс исполнений.

Иерархия:

IDispatch

IKompasAPIObject

IEmbodiment

Описание:

Интерфейс можно получить помощью свойств IEmbodimentsManager::Embodiment, IEmbodimentsManager::TopEmbodiment, IEmbodimentsManager::CurrentEmbodiment, IEmbodiment::Embodiment, IEmbodiment::Owner.

Имеет дополнительные интерфейсы:

- IPropertyKeeper,
- IColorParam7.
- IMassInertiaParam7,

которые можно получить с помощью метода IUnknown::QueryInterfaceI.

Embodiment – свойства

Density – Плотность

Интерфейс...

Тип данных: double

Синтаксис Automation:

Density = Object.Density	Получить свойство (*)
Density = Object.GetDensity()	Получить свойство (**)

Синтаксис COM:

Object.get_Density(&Density)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Задать плотность можно с помощью метода IEmbodiment::SetMaterial.

Embodiment – Зависимое исполнение

Интерфейс...

Тип данных: Указатель на интерфейс IEmbodiment

Синтаксис Automation:

```
Embodiment = Object.Embodiment( Index )          Получить свойство (* )
Embodiment = Object.GetEmbodiment( Index )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Embodiment( Index, &Embodiment )      Получить свойство
```

Примечание:

Свойство доступно только для чтения.

EmbodimentsCount – Количество исполнений

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
EmbodimentsCount = Object.EmbodimentsCount       Получить свойство (* )
EmbodimentsCount = Object.GetEmbodimentsCount()  Получить свойство (**)
```

Синтаксис COM:

```
Object.get_EmbodimentsCount( &EmbodimentsCount )  Получить свойство
```

Примечание:

Свойство доступно только для чтения.

IsCurrent – Текущее исполнение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsCurrent = Object.IsCurrent                     Получить свойство (* )
Object.IsCurrent = IsCurrent                     Установить свойство (* )
IsCurrent = Object.GetIsCurrent()                Получить свойство (**)
Object.SetIsCurrent( IsCurrent )                 Установить свойство (**)
```

Синтаксис COM:

Object.get_IsCurrent(&IsCurrent)	Получить свойство
Object.put_IsCurrent(IsCurrent)	Установить свойство

Примечание:

Свойство позволяет узнать, является ли исполнение текущим.

LeftHandedCS – Признак левосторонней системы координат

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LeftHandedCS = Object.LeftHandedCS		Получить свойство (*)
Object.LeftHandedCS = LeftHandedCS		Установить свойство (*)
LeftHandedCS = Object.GetLeftHandedCS()	=	Получить свойство (**)
Object.SetLeftHandedCS(LeftHandedCS)		Установить свойство (**)

Синтаксис COM:

Object.get_LeftHandedCS(&LeftHandedCS	Получить свойство
Object.put_LeftHandedCS(LeftHandedCS)	Установить свойство

Mass – Масса компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

Mass = Object.Mass	Получить свойство (*)
Mass = Object.GetMass()	Установить свойство (**)

Синтаксис COM:

Object.get_Mass(&Mass)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Material – Материал

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Material = Object.Material	Получить свойство (*)
Material = Object.GetMaterial()	Получить свойство (**)

Синтаксис COM:

Object.get_Material(&Material)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Задать материал можно с помощью метода IEmbodiment::SetMaterial.

Name – Имя

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object. put_Name(Name)	Установить свойство

Owner – Объект, породивший этот объект

Интерфейс...

Тип данных: Указатель на интерфейс IEmbodiment

Синтаксис Automation:

Owner = Object.Owner	Получить свойство (*)
Owner = Object.GetOwner()	Получить свойство (**)

Синтаксис COM:

Object.get_Owner(&Owner)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Part – Компонент исполнения

Тип данных: Указатель на интерфейс IPart7

Синтаксис Automation:

Part = Object.Part	Получить свойство (*)
Part = Object.GetPart()	Получить свойство (**)

Синтаксис COM:

Object.get_Part(&Part)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SpecRough – Неуказанная шероховатость 3D

Интерфейс...

Тип данных: Указатель на интерфейс ISpecRough3D

Синтаксис Automation:

SpecRough = Object.SpecRough	Получить свойство (*)
SpecRough = Object.GetSpecRough()	Получить свойство (**)

Синтаксис COM:

Object.get_SpecRough(&SpecRough)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEmbodiment – методы

Delete – Удаление

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetMarking – Обозначение исполнения

Интерфейс...

Синтаксис Automation:

BSTR GetMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer);

Синтаксис COM:

HRESULT GetMarking(ksVariantMarkingTypeEnum MarkingType, BOOL AddSystemDelimer, BSTR * Result);

Возвращаемое значение:

- обозначение исполнения.

Входные параметры:

MarkingType	- какую часть обозначения надо вернуть,
AddSystemDelimer	- с разделителями.

Метод возвращает обозначение исполнения.

SetMarking – Обозначение исполнения

Интерфейс...

Синтаксис Automation:

BOOL SetMarking(ksVariantMarkingTypeEnum MarkingType, BSTR Marking);

Синтаксис COM:

HRESULT SetMarking(ksVariantMarkingTypeEnum MarkingType, BSTR Marking, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

MarkingType - какую часть обозначения надо установить,
Marking - обозначение.

Метод устанавливает обозначение исполнения.

SetMaterial – Установить материал

Интерфейс...

Синтаксис Automation:

BOOL SetMaterial(BSTR Name, double Density);

Синтаксис COM:

HRESULT SetMaterial(BSTR Name, double Density, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Name - наименование материала,
Density - плотность материала.

Update – Изменить свойства объекта

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IFeature7

Справка системы КОМПАС...

`kompas.chm::/659_81_3_Derevo_modeli.htm`

Интерфейс объекта Дерева построения.

Иерархия:

IDispatch

IFeature7

Примечание:

1. Данный интерфейс можно получить от интерфейса IModelObject как свойство IModelObject::Owner.
2. Данный интерфейс можно получить от интерфейса IModelObject или IPart7 с помощью метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

IFeature7 – свойства

Excluded – Исключен из расчета

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Excluded = iObject.Excluded	Получить свойство (*)
iObject.Excluded = Excluded	Установить свойство (*)
Excluded = iObject.GetExcluded()	Получить свойство (**)
iObject.SetExcluded (Excluded)	Установить свойство (**)

Синтаксис COM:

iObject->get_Excluded(&Excluded)	Получить свойство
iObject->put_Excluded(Excluded)	Установить свойство

FeatureType – Тип объекта

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

FeatureType = iObject.FeatureType		Получить свойство (*)
FeatureType	=	Получить свойство (**)
iObject.GetFeatureType()		

Синтаксис COM:

iObject->get_FeatureType(&FeatureType)	Получить свойство

Возвращаемое значение:

- тип объектов из перечисления ksObj3dTypeEnum.

Примечание:

1. Свойство доступно только для чтения.
2. Тип возвращаемого объекта может принимать значения:

o3d_part	- деталь,
o3d_entity	- объект,
o3d_mateConstraint	- сопряжение.

ModelObjects – Объекты входящие в состав данного объекта (грани, ребра, вершины)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ModelObjects = iObject.ModelObjects (ObjType)	Получить свойство (*)
ModelObjects = iObject.GetModelObjects (ObjType)	Получить свойство (**)

Синтаксис COM:

iObject->get_ModelObjects (ObjType, &ModelObjects)	Получить свойство

Входные параметры:

VARIANT ObjType	- тип объектов из перечисления KompasAPIObjectTypeEnum.

Примечание:

1. Свойство доступно только для чтения.
2. По типу объекта o3d_unknown возвращается коллекция всех объектов.
3. Коллекция объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH). Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Name – Имя объекта

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

get_Name(&Name)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ObjectError – Получить номер ошибки

Интерфейс...

Тип данных: long

Синтаксис Automation:

ObjectError = Object.ObjectError	Получить свойство (*)
ObjectError = Object.GetObjectError()	Получить свойство (**)

Синтаксис COM:

Object.get_ObjectError(&ObjectError)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Если на объекте взведено несколько ошибок, возвращается номер первой.
3. Номера ошибок соответствуют перечислению ErrorType3d.

OwnerFeature – Объект-владелец

Интерфейс...

Тип данных: Указатель на интерфейс IFeature7

Синтаксис Automation:

OwnerFeature = iObject.OwnerFeature	Получить свойство (*)
OwnerFeature = iObject.GetOwnerFeature()	Получить свойство (**)

Синтаксис COM:

iObject->get_OwnerFeature(&OwnerFeature)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ResultBodies – Массив тел (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Значения свойства:

Массив SafeArray типа VT_ARRAY | VT_DISPATCH.

Синтаксис Automation:

ResultBodies = iObject.ResultBodies;	Получить свойство (*)
ResultBodies = iObject.GetResultBodies();	Получить свойство (**)

Синтаксис COM:

iObject->get_ResultBodies(&ResultBodies)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить массив SAFEARRAY всех тел, входящих в дерево. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

State – Возвращает комбинацию флагов состояния объекта

Интерфейс...

Тип данных: из перечисления ksFeatureStateEnum

Синтаксис Automation:

State = Object.State;	Получить свойство (*)
ResultBodies = State = Object.GetState();	Получить свойство (**)

Синтаксис COM:

Object.get_State(&State)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SubFeatures – Коллекция элементов дерева заданного типа в виде массива SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

SubFeatures = Object.SubFeatures(treeType, through, libObject)	Получить свойство (*)
SubFeatures = Object. GetSubFeatures (treeType, through, libObject)	Получить свойство (**)

Синтаксис COM:

Object.get_SubFeatures(treeType, through, libObject, &SubFeatures)	Получить свойство

Параметры:

treeType	- тип дерева из перечисления ksTreeTypeEnum,
through	- TRUE - выдавать все объекты, даже скрытые,
libObject	- TRUE - выдавать содержимое для библиотечного элемента.

Примечание:

Свойство доступно только для чтения.

UpdateStamp – Значение, определяющее время изменения геометрии

Интерфейс...

Тип данных: long

Синтаксис Automation:

UpdateStamp = iObject.UpdateStamp		Получить свойство (*)
UpdateStamp iObject.GetUpdateStamp()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_UpdateStamp(&UpdateStamp)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Valid – Признак невырожденности объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Valid = iObject.Valid	Получить свойство (*)
Valid = iObject.GetValid()	Получить свойство (**)

Синтаксис COM:

iObject->get_Valid(&Valid)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Variable – Получить параметрическую переменную по имени, индексу

Интерфейс...

Тип данных: указатель на интерфейс IVariable7

Синтаксис Automation:

Variable = Object.Variables(External, InSource, Index)	Получить свойство (*)
Variable = Object.GetVariables(External, InSource, Index)	Получить свойство (**)

Синтаксис COM:

Variable = Object.GetVariables(External, InSource, Index)	Получить свойство

Входные параметры:

BOOL ExternalOnly	- только внешние,
BOOL InSource	- из источника,
VARIANT Index	- имя или индекс переменной.

Примечание:

Свойство доступно только для чтения.

Variables – Получить массив параметрических переменных в виде SAFEARRAY VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Variables = Object.Variables( ExternalOnly, InSource )          Получить свойство (* )
Variables = Object.GetVariables( ExternalOnly, InSource )       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Variables( ExternalOnly, InSource, &Variables )          Получить свойство
```

Входные параметры:

```
BOOL ExternalOnly                    - только внешние,
BOOL InSource                        - из источника.
```

Примечание:

Свойство доступно только для чтения.

VariablesCount – Получить количество параметрических переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
VariablesCount = Object.VariablesCount( External, InSource )        Получить свойство (* )
VariablesCount = Object.GetVariablesCount( External, InSource )     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_VariablesCount( External, InSource, &VariablesCount )        Получить свойство
```

Входные параметры:

```
BOOL ExternalOnly                    - только внешние,
BOOL InSource                        - из источника.
```

Примечание:

Свойство доступно только для чтения.

IFeature7 – методы

Delete – Удалить компонент

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete (VARIANT_BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IMassInertiaParam7

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_MIX3D.htm

Интерфейс массово-центровочных характеристик.

Иерархия:

IDispatch

IMassInertiaParam7

Примечание:

1. Интерфейс контейнера является дополнительным к интерфейсу IPart7 и IEmbodiment. Данный интерфейс можно получить от IPart7 и от IEmbodiment посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);
2. Интерфейс позволяет получить доступ к массово-центровочным характеристикам детали и сборки. С помощью интерфейса можно получить и отредактировать массово-центровочные характеристики объекта.
3. Новые характеристики вступают в силу после вызова метода ImodelObject::Update.

IMassInertiaParam7 – свойства

Actual – Актуальность расчета

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/Z_INFO_WIN.htm

Тип данных: BOOL

Синтаксис Automation:

Actual = iObject.Actual	Получить свойство (*)
Actual = iObject.GetActual( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Actual(&Actual)	Получить свойство

Значения свойства:

TRUE	- информация актуальна,
FALSE	- не актуальна.

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить сведения об актуальности информации.

Area – Значение площади

Интерфейс...

Тип данных: double

Синтаксис Automation:

Area = iObject.Area	Получить свойство (*)
Area = iObject.GetArea( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Area(&Area)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить значение площади.

CalculateMass – Расчетное значение массы

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/PR_IDS_COMPONENT_PROPERTIES_CAPTION.htm

Тип данных: double

Синтаксис Automation:

CalculateMass = iObject.CalculateMass( )	Получить свойство (*)
CalculateMass = iObject.GetCalculateMass( )	Получить свойство (**)

Синтаксис COM:

iObject->get_CalculateMass(&CalculateMass)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить единицы измерения массы.

Density – Плотность

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Density = iObject.Density              Получить свойство (* )
Density = iObject.GetDensity()         Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Density(  &Density )      Получить свойство
```

Примечание:

1. Свойство позволяет получить плотность тела (г/куб.мм).
2. Свойство доступно только для чтения.
3. Для установки плотности необходимо использовать метод IMassInertiaParam7::SetMaterial.

DensityMode – Режим задания плотности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
DensityMode = iObject.DensityMode          Получить свойство (* )
iObject.DensityMode = DensityMode          Установить свойство (* )
DensityMode = iObject.GetDensityMode( )    Получить свойство (**)
iObject.SetDensityMode( DensityMode )      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DensityMode( &DensityMode )   Получить свойство
iObject->put_DensityMode( DensityMode )    Установить свойство
```

Значения свойства:

```
TRUE     - ручное задание плотности,
FALSE    - значение плотности из Справочника.
```

Примечание:

Свойство позволяет установить и получить режим задания плотности.

HandBookDensity – Значение плотности из Справочника

Интерфейс...

Тип данных: double

Синтаксис Automation:

HandBookDensity = Получить свойство (*)
iObject.HandBookDensity
HandBookDensity = Получить свойство (**)
iObject.GetHandBookDensity( )

Синтаксис COM:

iObject->get_HandBookDensity(Получить свойство
&HandBookDensity)

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить значение плотности из Справочника.

Jx – Осевой момент инерции в центральной системе координат относительно центральной оси координат X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jx = iObject.Jx Получить свойство (*)
Jx = iObject.GetJx( ) Получить свойство (**)

Синтаксис COM:

iObject->get_Jx(&Jx) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно относительно центральной оси координат X.

Jy – Осевой момент инерции в центральной системе координат относительно центральной оси координат Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jy = iObject.Jy	Получить свойство (*)
Jy = iObject.GetJy( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Jy(&Jy)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно относительно центральной оси координат Y.

Jz – Осевой момент инерции в центральной системе координат относительно центральной оси координат Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jz = iObject.Jz	Получить свойство (*)
Jz = iObject.GetJz( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Jz(&Jz)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно относительно центральной оси координат Z.

Jxy – Центробежный момент инерции в центральной системе координат относительно центральных осей координат X и Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jxy = iObject.Jxy	Получить свойство (*)
Jxy = iObject.GetJxy( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Jxy(&Jxy) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение центробежного момента инерции относительно центральных осей координат X и Y.

Jxz – Центробежный момент инерции центральной системе координат относительно центральных осей координат X и Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jxz = iObject.Jxz Получить свойство (*)
Jxz = iObject.GetJxz( ) Получить свойство (**)

Синтаксис COM:

iObject->get_Jxz(&Jxz) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение центробежного момента инерции относительно центральных осей координат X и Z.

Jyz – Центробежный момент инерции в центральной системе координат относительно центральных осей координат Y и Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jyz = iObject.Jyz Получить свойство (*)
Jyz = iObject.GetJyz( ) Получить свойство (**)

Синтаксис COM:

iObject->get_Jyz(&Jyz) Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение центробежного момента инерции относительно центральных осей координат Y и Z.

Jx0 – Осевой момент инерции в главной центральной системе координат относительно оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jx0 = iObject.Jx0	Получить свойство (*)
Jx0 = iObject.GetJx0( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Jx0(&Jx0)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Jy0 – Осевой момент инерции в главной центральной системе координат относительно оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Jy0 = iObject.Jy0	Получить свойство (*)
Jy0 = iObject.GetJy0( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Jy0(&Jy0)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Jz0 – Осевой момент инерции в главной центральной системе координат относительно оси Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Jz0 = iObject.Jz0                    Получить свойство (* )
Jz0 = iObject.GetJz0( )              Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Jz0( &Jz0 )             Получить свойство
```

Примечание:

Свойство доступно только для чтения.

LengthUnits – Единицы измерения длины

Интерфейс...

Тип данных: из перечисления ksLengthUnitEnum

Синтаксис Automation:

```
LengthUnits = iObject.LengthUnits          Получить свойство (* )
iObject.LengthUnits = LengthUnits          Установить свойство (* )
LengthUnits = iObject.GetLengthUnits( )    Получить свойство (**)
iObject.SetLengthUnits( LengthUnits )      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_LengthUnits( &LengthUnits )   Получить свойство
iObject->put_LengthUnits( LengthUnits )    Установить свойство
```

Примечание:

Свойство позволяет установить и получить единицы измерения длины.

Lx – Осевой момент инерции относительно глобальной (исходной) оси координат X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lx = iObject.Lx                  Получить свойство (* )
Lx = iObject.GetLx( )            Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Lx( &Lx )           Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходной оси координат X.

Ly – Осевой момент инерции относительно глобальной (исходной) оси координат Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Ly = iObject.Ly              Получить свойство (* )
Ly = iObject.GetLy( )        Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Ly( &Ly )       Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходной оси координат Y.

Lz – Осевой момент инерции относительно глобальной (исходной) оси координат Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lz = iObject.Lz              Получить свойство (* )
Lz = iObject.GetLz( )        Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Lz( &Lz )       Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходной оси координат Z.

Lxy – Центробежный момент инерции относительно исходных осей координат X и Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lxy = iObject.Lxy                Получить свойство (* )
Lxy = iObject.GetLxy( )          Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Lxy( &Lxy )         Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходных осей координат X и Y.

Lxz – Центробежный момент инерции относительно исходных осей координат X и Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lxz = iObject.Lxz                Получить свойство (* )
Lxz = iObject.GetLxz( )          Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Lxz( &Lxz )         Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходных осей координат X и Z.

Lyz – Центробежный момент инерции относительно исходных осей координат Y и Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

Lyz = iObject.Lyz	Получить свойство (*)
Lyz = iObject.GetLyz( )	Получить свойство (**)

Синтаксис COM:

iObject->get_Lyz(&Lyz)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить значение осевого момента инерции относительно исходных осей координат Y и Z.

ManualDensity – Ручное задание плотности

Интерфейс...

Тип данных: double

Синтаксис Automation:

iObject.ManualDensity = ManualDensity	Установить свойство (*)
iObject.SetManualDensity(ManualDensity)	Установить свойство (**)

Синтаксис COM:

iObject->put_ManualDensity(ManualDensity)	Установоить свойство

Примечание:

1. Свойство доступно только для записи. Позволяет явно установить значение плотности. Введенное значение появится в поле Плотность.
2. Единицы измерения плотности определяются значениями свойств IMassInertiaParam7::MassUnits и IMassInertiaParam7::LengthUnits.
3. Для задания плотности необходимо выполнить следующие действия:

- ▼ установить режим Расчет параметров плотности (см. свойство IMassInertiaParam7::MassSettingMode).
- ▼ установить режим Ручной ввод плотности (см. свойство IMassInertiaParam7::DensityMode).

ManualMass – Ручное задание массы

Интерфейс...

Тип данных: double

Синтаксис Automation:

iObject.ManualMass = ManualMass Установить свойство (*)
iObject.SetManualMass(ManualMass) Установить свойство (**)

Синтаксис COM:

iObject->put_ManualMass(ManualMass) Установить свойство

Примечание:

Свойство доступно только для записи. Позволяет вручную установить значение массы. Для задания массы необходимо установить режим ручного задания параметров (см. свойство IMassInertiaParam7::MassSettingMode).

ManualMassCentre – Признак ручного задания центра масс

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ManualMassCentre = Получить свойство (*)
iObject.ManualMassCentre
iObject.ManualMassCentre = Установить свойство (*)
ManualMassCentre
ManualMassCentre = Получить свойство (**)
iObject.GetManualMassCentre( )
iObject.SetManualMassCentre(ManualMassCentre) Установить свойство (**)

Синтаксис COM:

iObject->get_ManualMassCentre(&ManualMassCentre) Получить свойство
iObject->put_ManualMassCentre(ManualMassCentre) Установить свойство

Значения свойства:

TRUE - ручное задание координат центра масс,
FALSE - рассчитанное значение.

Примечание:

Свойство позволяет установить и получить признак ручного задания центра масс.

Mass – Масса компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Mass = iObject.Mass                Получить свойство (* )
Mass = iObject.GetMass()           Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Mass ( &Mass)         Получить свойство
```

Примечание:

1. Свойство позволяет получить массу тела.
2. Свойство доступно только для чтения.

MassSettingMode – Режим задания параметров

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/PR_IDS_COMPONENT_PROPERTIES_CAPTION.htm`

Тип данных: из перечисления ksMassSettingModeEnum

Синтаксис Automation:

```
MassSettingMode                    =  Получить свойство (* )
iObject.MassSettingMode
iObject.MassSettingMode            =  Установить свойство (* )
MassSettingMode
MassSettingMode                    =  Получить свойство (**)
iObject.GetMassSettingMode( )
iObject.SetMassSettingMode(           Установить свойство (**)
MassSettingMode  )
```

Синтаксис COM:

```
iObject->get_MassSettingMode(      Получить свойство
&MassSettingMode  )
iObject->put_MassSettingMode(      Установить свойство
MassSettingMode  )
```

Примечание:

Свойство позволяет установить и получить режим задания параметров.

MassUnits – Единицы измерения массы

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/PR_IDS_COMPONENT_PROPERTIES_CAPTION.htm

Тип данных: из перечисления ksMassUnitsEnum

Синтаксис Automation:

MassUnits = iObject.MassUnits	Получить свойство (*)
iObject.MassUnits = MassUnits	Установить свойство (*)
MassUnits = iObject.GetMassUnits( )	Получить свойство (**)
iObject.SetMassUnits(MassUnits)	Установить свойство (**)

Синтаксис COM:

iObject->get_MassUnits(&MassUnits)	Получить свойство
iObject->put_MassUnits(MassUnits)	Установить свойство

Примечание:

Свойство позволяет установить и получить единицы измерения массы.

Material – Материал

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Material = iObject.Material	Получить свойство (*)
Material = iObject.GetMaterial()	Получить свойство (**)

Синтаксис COM:

iObject->get_Material (&Material)	Получить свойство

Примечание:

1. Свойство позволяет получить материал тела.
2. Свойство доступно только для чтения.
3. Для установки материала необходимо использовать метод IMassInertiaParam7::SetMaterial.

MaterialLocation – Идентификатор материала

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

MaterialLocation = Object.MaterialLocation	=	Получить свойство (*)
Object.MaterialLocation = MaterialLocation	=	Установить свойство (*)
MaterialLocation = Object.GetMaterialLocation()	=	Получить свойство (**)
Object.SetMaterialLocation(MaterialLocation)		Установить свойство (**)

Синтаксис COM:

Object.get_MaterialLocation(&MaterialLocation)	Получить свойство
Object.put_MaterialLocation(MaterialLocation)	Установить свойство

SourceData – Данные из источника

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SourceData = iObject.SourceData	Получить свойство (*)
iObject.SourceData = SourceData	Установить свойство (*)
SourceData = iObject.GetSourceData( )	Получить свойство (**)
iObject.SetDensityMode(DensityMode)	Установить свойство (**)

Синтаксис COM:

iObject->get_SourceData(&SourceData)	Получить свойство
iObject->put_SourceData(SourceData)	Установить свойство

Значения свойства:

TRUE	- данные из источника,
FALSE	- отличаются от источника.

Примечание:

1. Свойство доступно только для компонента, вставленного в сборку.
2. Свойство позволяет установить и получить признак того, что данные взяты из источника.

Volume – Значение объема

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Volume = iObject.Volume              Получить свойство (* )
Volume = iObject.GetVolume( )        Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Volume( &Volume )       Получить свойство
```

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получить значение объема.

Xc – Координата X центра масс

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Xc = iObject.Xc                      Получить свойство (* )
iObject.Xc = Xc                      Установить свойство (* )
Xc = iObject.GetXc( )                Получить свойство (**)
iObject.SetXc( Xc )                  Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Xc( &Xc )               Получить свойство
iObject->put_Xc( Xc )                Установить свойство
```

Примечание:

Свойство позволяет установить и получить координату X центра масс. Для установки координат центра масс должен быть включен режим ручной установки центра масс (см. свойство IMassInertiaParam7::ManualMassCentre).

Yc – Координата Y центра масс

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Zc = iObject.Zc                 Получить свойство (* )
iObject.Zc = Zc                 Установить свойство (* )
Zc = iObject.GetZc( )           Получить свойство (**)
iObject.SetZc( Zc )             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Zc( &Zc )          Получить свойство
iObject->put_Zc( Zc )           Установить свойство
```

Примечание:

Свойство позволяет установить и получить координату Z центра масс. Для установки координат центра масс должен быть включен режим ручной установки центра масс (см. свойство IMassInertiaParam7::ManualMassCentre).

Zc – Координата Z центра масс

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Yc = iObject.Yc                 Получить свойство (* )
iObject.Yc = Yc                 Установить свойство (* )
Yc = iObject.GetYc( )           Получить свойство (**)
iObject.SetYc( Yc )             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Yc( &Yc )          Получить свойство
iObject->put_Yc( Yc )           Установить свойство
```

Примечание:

Свойство позволяет установить и получить координату Y центра масс. Для установки координат центра масс должен быть включен режим ручной установки центра масс (см. свойство IMassInertiaParam7::ManualMassCentre).

IMassInertiaParam7 – методы

Calculate – Рассчитать параметры

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_MIX3D.htm

Синтаксис Automation:

BOOL Calculate( );

Синтаксис COM:

HRESULT Calculate (VARIANT_BOOL* Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет рассчитать массово-центровочные характеристики объекта. Метод следует вызывать после изменения одного из параметров объекта, чтобы привести остальные параметры в соответствие с измененным.

GetAxisX – Получить вектор направлений главных центральных осей инерции по оси X

Интерфейс...

Синтаксис Automation:

BOOL GetAxisX(double * x, double * y, double * z);

Синтаксис COM:

HRESULT GetAxisX(double * x, double * y, double * z, VARIANT_BOOL * Result);

Выходные параметры:

	Координаты x, y, z точки вектора направления по оси X.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получать координаты x, y, z точки вектора направления по оси X.

GetAxisY – Получить вектор направлений главных центральных осей инерции по оси Y

Интерфейс...

Синтаксис Automation:

BOOL GetAxisY(double * x, double * y, double * z);

Синтаксис COM:

HRESULT GetAxisY(double * x, double * y, double * z, VARIANT_BOOL * Result);

Выходные параметры:

Координаты x, y, z точки вектора направления по оси Y.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет получать координаты x, y, z точки вектора направления по оси Y.

GetAxisZ – Получить вектор направлений главных центральных осей инерции по оси Z

Интерфейс...

Синтаксис Automation:

BOOL GetAxisZ(double * x, double * y, double * z);

Синтаксис COM:

HRESULT GetAxisZ(double * x, double * y, double * z, VARIANT_BOOL * Result);

Выходные параметры:

- Координаты x, y, z точки вектора направления по оси Z.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет получать координаты x, y, z точки вектора направления по оси Z.

SetMaterial – Установить материал и плотность

Интерфейс...

Синтаксис Automation:

BOOL SetMaterial(BSTR Name,
double density);

Синтаксис COM:

HRESULT SetMaterial(BSTR Name,
double Density,
BOOL * PVal);

Входные параметры:

Name	- наименование материала,
Density	- плотность материала (г/куб.мм).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет рассчитать массово-центровочные характеристики объекта. Метод следует вызывать после изменения одного из параметров объекта, чтобы привести остальные параметры в соответствие с измененным.

Интерфейс ISourcePart7Params

Интерфейс параметров компонента в источнике.

Иерархия:

IDispatch

ISourcePart7Params

Описание:

Интерфейс является дополнительным для интерфейса компонента IPart7. Он позволяет получить параметры компонента и параметры документа из источника. Данный интерфейс можно получить от IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);

ISourcePart7Params – свойства

DocumentAuthor – Автор документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DocumentAuthor = iObject.DocumentAuthor;	Получить свойство (*)
iObject.DocumentAuthor = DocumentAuthor;	Установить свойство (*)
DocumentAuthor = iObject.GetDocumentAuthor();	Получить свойство (**)

```
iObject.DocumentAuthor(                 Установить свойство (**)
DocumentAuthor );
```

Синтаксис COM:

```
iObject->get_DocumentAuthor(            Получить свойство
&DocumentAuthor );
iObject->put_DocumentAuthor(            Установить свойство
DocumentAuthor );
```

DocumentComment – Комментарий

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
DocumentComment =                       Получить свойство (* )
iObject.DocumentComment;
iObject.DocumentComment=                Установить свойство (* )
DocumentComment;
DocumentComment =                       Получить свойство (**)
iObject.DocumentComment();
iObject.DocumentComment(                Установить свойство (**)
DocumentComment );
```

Синтаксис COM:

```
iObject-                                Получить свойство
>get_DocumentComment(
&DocumentComment );
iObject-                                Установить свойство
>put_DocumentComment(
DocumentComment );
```

SourceMarking – Обозначение в источнике

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
SourceMarking =                         Получить свойство (* )
iObject.SourceMarking;
iObject.SourceMarking =                 Установить свойство (* )
SourceMarking;
```

```
SourceMarking =                          Получить свойство (**)
iObject.GetSourceMarking();
iObject.SourceMarking(                   Установить свойство (**)
SourceMarking );
```

Синтаксис COM:

```
iObject->get_SourceMarking(              Получить свойство
&SourceMarking );
iObject->put_SourceMarking(              Установить свойство
SourceMarking );
```

SourceName – Имя компонента в источнике

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
SourceName = iObject.SourceName;         Получить свойство (* )
iObject.SourceName = SourceName;         Установить свойство (* )
SourceName = iObject.GetSourceName();    Получить свойство (**)
iObject.SetSourceName( SourceName );     Установить свойство (**)
```

Синтаксис COM:

```
iObject-                                 Получить свойство
>get_SourceName(
&SourceName );
iObject-                                 Установить свойство
>put_SourceName(
SourceName );
```

Интерфейс IEmbodimentsManager

Интерфейс менеджера исполнений

Иерархия:

IDispatch

    IEmbodimentsManager

Описание:

Данный интерфейс можно получить у интерфейсов IPart7, IKompasDocument3D, IAssociationView с помощью метода IUnknown::QueryInterface.

IEmbodimentsManager – свойства

CurrentEmbodiment – Текущее исполнение

Интерфейс...

Тип данных: Указатель на интерфейс IEmbodiment

Синтаксис Automation:

CurrentEmbodiment= Object.CurrentEmbodiment	Получить свойство (*)
CurrentEmbodiment= Object.GetCurrentEmbodiment	Получить свойство (**)

Синтаксис COM:

Object.get_CurrentEmbodiment(&CurrentEmbodiment)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CurrentEmbodimentIndex – Индекс текущего исполнения

Интерфейс...

Тип данных: long

Синтаксис Automation:

CurrentEmbodimentMarking Object.CurrentEmbodiment Index	=	Получить свойство (*)
CurrentEmbodimentMarking Object.GetCurrentEmbodiment Index ()	=	Получить свойство (**)

Синтаксис COM:

Object.get_CurrentEmbodimentIndex(&CurrentEmbodimentIndex)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Embodiment – Получить исполнение по индексу или обозначению

Интерфейс...

Тип данных: Указатель на интерфейс IEmbodiment

Синтаксис Automation:

Embodiment = Object.Embodiment(Index)	Получить свойство (*)

Embodiment = Object.GetEmbodiment(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Embodiment(Index,&Embodiment)	Получить свойство

Примечание:

Свойство доступно только для чтения.

EmbodimentAdditionalNumber – Дополнительный номер исполнения модели, использованного для вставки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

EmbodimentAdditionalNumber = Object.EmbodimentAdditionalNumber;		Получить свойство (*)
Object.EmbodimentAdditionalNumber = EmbodimentAdditionalNumber;		Установить свойство (*)
EmbodimentAdditionalNumber = Object.GetEmbodimentAdditionalNumber();		Получить свойство (**)
Object.SetEmbodimentAdditionalNumber(EmbodimentAdditionalNumber);		Установить свойство (**)

Синтаксис COM:

Object.get_EmbodimentAdditionalNumber(&EmbodimentAdditionalNumber ;	Получить свойство
Object.put_EmbodimentAdditionalNumber(EmbodimentAdditionalNumber);	Установить свойство

EmbodimentCount – Количество исполнений

Интерфейс...

Тип данных: long

Синтаксис Automation:

EmbodimentCount = Object.EmbodimentCount	Получить свойство (*)
EmbodimentCount = Object.GetEmbodimentCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EmbodimentCount(&EmbodimentCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TopEmbodiment – Основное исполнение

Интерфейс...

Тип данных: Указатель на интерфейс IEmbodiment.

Синтаксис Automation:

TopEmbodiment = Object.TopEmbodiment	Получить свойство (*)
TopEmbodiment = Object.GetTopEmbodiment	Получить свойство (**)

Синтаксис COM:

Object.get_TopEmbodiment(&TopEmbodiment)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEmbodimentsManager– методы

AddEmbodiment – Добавляет новое исполнение и делает его текущим

Интерфейс...

Синтаксис Automation:

BOOL AddEmbodiment(VARIANT ParentIndex, BSTR Marking);

Синтаксис COM:

HRESULT AddEmbodiment(VARIANT ParentIndex, BSTR Marking, BOOL * Result);

Входные параметры:

ParentIndex	- индекс родительского исполнения,
Marking	- обозначение исполнения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

AddMirrorEmbodiment – Добавляет новое зеркальное исполнение и делает его текущим

Интерфейс...

Синтаксис Automation:

BOOL AddMirrorEmbodiment(VARIANT ParentIndex, BSTR BaseMarking, BSTR EmbodimentNumber, BSTR AdditionalNumber);

Синтаксис COM:

HRESULT AddMirrorEmbodiment(VARIANT ParentIndex, BSTR BaseMarking, BSTR EmbodimentNumber, BSTR AdditionalNumber, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

ParentIndex - индекс родительского исполнения,
BaseMarking - обозначение исполнения,
EmbodimentNumber - номер исполнения,
AdditionalNumber - дополнительный номер.

DeleteEmbodiment – Удалить исполнение по имени или индексу

Интерфейс...

Синтаксис Automation:

BOOL DeleteEmbodiment(VARIANT Index);

Синтаксис COM:

HRESULT DeleteEmbodiment(VARIANT Index, BOOL * Result);

Входные параметры:

Index - индекс удаляемого обозначения.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetCurrentEmbodimentMarking – Обозначение текущего исполнения

Интерфейс...

Синтаксис Automation:

BSTR GetCurrentEmbodimentMarking(ksVariantMarkingTypeEnum MarkingType,
BOOL AddSystemDelimer);

Синтаксис COM:

HRESULT GetCurrentEmbodimentMarking(ksVariantMarkingTypeEnum MarkingType,
BOOL AddSystemDelimer,
BSTR * Result);

Возвращаемое значение:

- обозначение текущего исполнения.

Входные параметры:

MarkingType - часть обозначения из перечисления ksVariantMarkingTypeEnum,
AddSystemDelimer - добавлять системные разделители.

Метод позволяет получить обозначение текущего исполнения.

GetEmbodimentMarking – Получить обозначение исполнения

Интерфейс...

Синтаксис Automation:

BSTR GetEmbodimentMarking(long Index,
ksVariantMarkingTypeEnum MarkingType,
BOOL AddSystemDelimer);

Синтаксис COM:

HRESULT GetEmbodimentMarking(long Index,
ksVariantMarkingTypeEnum MarkingType,
BOOL AddSystemDelimer,
BSTR * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Index - индекс обозначения,
MarkingType - часть обозначения из перечисления ksVariantMarkingTypeEnum,
AddSystemDelimer - добавлять системные разделители.

GetEmbodimentsTree – Дерево исполнений. Массив узлов дерева исполнений SAFEARRAY BSTR – (VT_ARRAY | VT_BSTR)

Интерфейс...

Синтаксис Automation:

VARIANT GetEmbodimentsTree(BOOL AddSpaces);

Синтаксис COM:

HRESULT GetEmbodimentsTree(BOOL AddSpaces, VARIANT * Result);

Возвращаемое значение:

Массив SAFEARRAY BSTR - (VT_ARRAY | VT_BSTR);

Входные параметры:

AddSpaces - добавлять пробелы.

Метод возвращает массив обозначений исполнений.

Примечание:

При AddSpaces = TRUE перед обозначением исполнения добавляется соответствующее уровню вложенности количество пробелов.

SetCurrentEmbodiment – Установить текущее исполнение по имени или индексу

Интерфейс...

Синтаксис Automation:

BOOL SetCurrentEmbodiment(VARIANT Index);

Синтаксис COM:

HRESULT SetCurrentEmbodiment(VARIANT Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Index - индекс или имя исполнения.

SetEmbodimentMarking – Установить обозначение исполнения

Интерфейс...

Синтаксис Automation:

BOOL SetEmbodimentMarking(long Index,
ksVariantMarkingTypeEnum MarkingType,
BSTR Marking);

Синтаксис COM:

HRESULT SetEmbodimentMarking(long Index,
ksVariantMarkingTypeEnum MarkingType,
BSTR Marking,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	индекс обозначения,
MarkingType	- часть обозначения из перечисления ksVariantMarkingTypeEnum,
Marking	- обозначение.

Интерфейс IBilletObsolete

Интерфейс Детали заготовки и Зеркальной детали

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
ILocalCSObject
IEmbodimentsManager
IBilletObsolete

IBilletObsolete – свойства

FileName – Имя файла

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)

Object.SetFileName(FileName) — Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName) — Получить свойство
Object.put_FileName(FileName) — Установить свойство

Интерфейс IMateConstraints3D

Интерфейс коллекции сопряжений 3D.

Иерархия:

Интерфейс можно получить методами IPart7::MateConstraints,
IKompasDocument3D1::MateConstraints,
IProcess3D::MateConstraints

IMateConstraints3D – свойства

MateConstraint3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IMateConstraints3D.

Синтаксис Automation:

MateConstraint3D = Object.MateConstraint3D(Index) — Получить свойство (*)
MateConstraint3D = Object.GetMateConstraint3D(Index) — Получить свойство (**)

Синтаксис COM:

Object.get_MateConstraint3D(Index, &MateConstraint3D) — Получить свойство

Примечание.

Свойство доступно только для чтения.

ObjectConstraints – Возвращает массив сопряжений, связанных с объектом

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ObjectConstraints = Получить свойство (*)
Object.ObjectConstraints(Object)
ObjectConstraints = Получить свойство (**)
Object.GetObjectConstraints(Object)

Синтаксис COM:

Object.get_ObjectConstraints(Получить свойство
Object, &ObjectConstraints)

Примечание.

Свойство доступно только для чтения.

IMateConstraints3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(MateConstraintType Type);

Синтаксис COM:

HRESULT Add(MateConstraintType Type, IMateConstraint3D * * Result);

Возвращаемое значение:

- новое сопряжение.

Входные параметры:

Type - тип сопряжения.

AddUserMate – Создает новый пользовательский элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddUserMate(MateConstraintType Type);

Синтаксис COM:

HRESULT AddUserMate(MateConstraintType Type, IMateConstraint3D * * Result);

Возвращаемое значение:

- новое сопряжение.

Входные параметры:

Type - тип сопряжения.

Интерфейс IMateConstraint3D

Интерфейс сопряжений 3D.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

Интерфейс можно получить методами:

IMateConstraints3D::MateConstraint3D,

IMateConstraints3D::ObjectConstraints,

IMateConstraints3D::Add,

IMateConstraints3D::AddUserMate.

IMateConstraint3D – свойства

Alignment – Варианты выравнивания направлений для сопряжений

Интерфейс...

Тип данных: из перечисления ksMateConstraintAlignmentEnum

Синтаксис Automation:

Alignment = Object.Alignment	Получить свойство (*)
Object.Alignment = Alignment	Установить свойство (*)
Alignment = Object.GetAlignment()	Получить свойство (**)
Object.SetAlignment(Alignment)	Установить свойство (**)

Синтаксис COM:

Object.get_Alignment(&Alignment)	Получить свойство
Object.put_Alignment(Alignment)	Установить свойство

BaseObject1 – Базовый объект 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject1 = Object.BaseObject1	Получить свойство (*)
Object.BaseObject1 = BaseObject1	Установить свойство (*)
BaseObject1 = Object.GetBaseObject1()	Получить свойство (**)
Object.SetBaseObject1(BaseObject1)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject1(&BaseObject1)	Получить свойство
Object.put_BaseObject1(BaseObject1)	Установить свойство

BaseObject2 – Базовый объект 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject2 = Object.BaseObject2	Получить свойство (*)
Object.BaseObject2 = BaseObject2	Установить свойство (*)
BaseObject2 = Object.GetBaseObject2()	Получить свойство (**)
Object.SetBaseObject2(BaseObject2)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject2(&BaseObject2)	Получить свойство
Object.put_BaseObject2(BaseObject2)	Установить свойство

ConstraintType – Тип сопряжения

Интерфейс...

Тип данных: из перечисления MateConstraintType

Синтаксис Automation:

ConstraintType = Object.ConstraintType	Получить свойство (*)
ConstraintType = Object.GetConstraintType()	Получить свойство (**)

Синтаксис COM:

Object.get_ConstraintType(&ConstraintType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Fixed – Установить признак фиксации

Интерфейс...

Тип данных: из перечисления: ksMateFixedTypeEnum

Синтаксис Automation:

Fixed = Object.Fixed	Получить свойство (*)
Object.Fixed = Fixed	Установить свойство (*)
Fixed = Object.GetFixed()	Получить свойство (**)
Object.SetFixed(Fixed)	Установить свойство (**)

Синтаксис COM:

Object.get_Fixed(&Fixed)	Получить свойство
Object.put_Fixed(Fixed)	Установить свойство

ParamValue – Параметр для ограничений (расстояние или угол между объектами)

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamValue = Object.ParamValue	Получить свойство (*)
Object.ParamValue = ParamValue	Установить свойство (*)
ParamValue = Object.GetParamValue()	Получить свойство (**)
Object.SetParamValue(ParamValue)	Установить свойство (**)

Синтаксис COM:

Object.get_ParamValue(&ParamValue)	Получить свойство
Object.put_ParamValue(ParamValue)	Установить свойство

IMateConstraint3D – методы

GetMateParams – Получить параметры математических объектов, участвующих в сопряжении

Интерфейс...

Синтаксис Automation:

ksMateType GetMateParams(long Index, VARIANT * Params);

Синтаксис COM:

HRESULT GetMateParams(long Index, VARIANT * Params, ksMateType * Result);

Возвращаемое значение:

тип математического объекта, участвующего в сопряжении из ksMateType - если параметры получены,

ksMateUnknown - в случае ошибки.

Входные параметры:

Index - индекс объекта.

Выходные параметры:

Params - параметры объекта массив SafeArray вещественных чисел.

Примечание:

1. Параметры возвращаются в системе координат сборки. При необходимости их можно перевести в систему координат нужного компонента с помощью функции IPart::TransformPoint.
2. Геометрический объект для сопряжения описывается набором из 8-ми вещественных чисел, сгруппированных следующим образом:

- три координаты точки Pc, Pc = { pointX, pointY, pointZ },
- три координаты вектора V, V = { vectorI, vectorJ, vectorK },
- две координаты, соответствующие радиусам, V = { vectorI, vectorJ, vectorK }.

Интерфейс IMate3DByAngle

Интерфейс сопряжения под углом.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DByAngle

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IMate3DByAngle– свойства

Axis – Ось плоского угла

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Axis = Object.Axis              Получить свойство (* )
 Object.Axis = Axis             Установить свойство (* )
Axis = Object.GetAxis()         Получить свойство (**)
Object.SetAxis( Axis )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Axis( &Axis )    Получить свойство
Object.put_Axis(  Axis )    Установить свойство
```

Angle3D – Тип угла. TRUE – пространственный угол, FALSE – плоский угол

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Angle3D = Object.Angle3D              Получить свойство (* )
Object.Angle3D = Angle3D              Установить свойство (* )
Angle3D = Object.GetAngle3D()         Получить свойство (**)
Object.SetAngle3D( Angle3D )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle3D( &Angle3D )    Получить свойство
Object.put_Angle3D(  Angle3D )    Установить свойство
```

Интерфейс IMate3DDependentPosition

Интерфейс сопряжения Зависимое положение.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DDependentPosition

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IMate3DDependentPosition – свойства

BySample – По образцу

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
BySample = Object.BySample              Получить свойство (* )
Object.BySample = BySample              Установить свойство (* )
BySample = Object.GetBySample()         Получить свойство (**)
Object.SetBySample( BySample )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BySample( &BySample )        Получить свойство
Object.put_BySample(  BySample )        Установить свойство
```

SampleObject1 – Образец – зависимый компонент

Интерфейс...

Тип данных: Указатель на интерфейс IPart7

Синтаксис Automation:

```
SampleObject1 = Object.SampleObject1    Получить свойство (* )
Object.SampleObject1 = SampleObject1    Установить свойство (* )
SampleObject1 =                         Получить свойство (**)
Object.GetSampleObject1()
Object.SetSampleObject1(                Установить свойство (**)
SampleObject1) )
```

Синтаксис COM:

```
Object.get_SampleObject1(               Получить свойство
&SampleObject1 )
Object.put_SampleObject1(               Установить свойство
SampleObject1 ))
```

SampleObject2 – Образец – базовый компонент

Интерфейс...

Тип данных: Указатель на интерфейс IPart7

Синтаксис Automation:

SampleObject2 = Object.SampleObject2	Получить свойство (*)
Object.SampleObject2 = SampleObject2	Установить свойство (*)
SampleObject2 = Object.GetSampleObject2()	Получить свойство (**)
Object.SetSampleObject2(SampleObject2))	Установить свойство (**)

Синтаксис COM:

Object.get_SampleObject2(&SampleObject2))	Получить свойство
Object.put_SampleObject2(SampleObject2)	Установить свойство

Интерфейс IMate3DSymmetry

Интерфейс сопряжения Симметрия.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DSymmetry

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface.

IMate3DSymmetry– свойства

Plane – Плоскость симметрии

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)

Object.SetPlane(Plane) Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane) Получить свойство
Object.put_Plane(Plane) Установить свойство

Интерфейс IMate3DByTangent

Интерфейс сопряжения по касательной.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DByTangent

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface

IMate3DByTangent – свойства

TangentType – Вид касания

Интерфейс...

Тип данных: из перечисления ksMateTangentTypeEnum

Синтаксис Automation:

TangentType = Object.TangentType Получить свойство (*)
Object.TangentType = TangentType Установить свойство (*)
TangentType = Object.GetTangentType() Получить свойство (**)
Object.SetTangentType(TangentType) Установить свойство (**)

Синтаксис COM:

Object.get_TangentType(&TangentType) Получить свойство
Object.put_TangentType(TangentType) Установить свойство

Интерфейс IMate3DCamGear

Интерфейс сопряжения двух компонентов кулачкового механизма Кулачек – Толкатель.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DCamGear

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface

IMate3DCamGear – свойства

CamFaces – Массив SAFEARRAY граней кулачка

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

CamFaces = Object.CamFaces	Получить свойство (*)
Object.CamFaces = CamFaces	Установить свойство (*)
CamFaces = Object.GetCamFaces()	Получить свойство (**)
Object.SetCamFaces(CamFaces)	Установить свойство (**)

Синтаксис COM:

Object.get_CamFaces(&CamFaces)	Получить свойство
Object.put_CamFaces(CamFaces)	Установить свойство

FollowerFace – Рабочая грань толкателя

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

FollowerFace = Object.FollowerFace	Получить свойство (*)
Object.FollowerFace = FollowerFace	Установить свойство (*)
FollowerFace = Object.GetFollowerFace()	Получить свойство (**)
Object.SetFollowerFace(FollowerFace)	Установить свойство (**)

Синтаксис COM:

Object.get_FollowerFace(&FollowerFace)	Получить свойство
Object.put_FollowerFace(FollowerFace)	Установить свойство

RotationAxis – Ось вращения кулачка

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

RotationAxis = Object.RotationAxis	Получить свойство (*)
Object.RotationAxis = RotationAxis	Установить свойство (*)
RotationAxis = Object.GetRotationAxis()	Получить свойство (**)
Object.SetRotationAxis(RotationAxis)	Установить свойство (**)

Синтаксис COM:

Object.get_RotationAxis(&RotationAxis)	Получить свойство
Object.put_RotationAxis(RotationAxis)	Установить свойство

Trajectory – Траектория перемещения толкателя

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Trajectory = Object.Trajectory	Получить свойство (*)
Object.Trajectory = Trajectory	Установить свойство (*)
Trajectory = Object.GetTrajectory()	Получить свойство (**)
Object.SetTrajectory(Trajectory)	Установить свойство (**)

Синтаксис COM:

Object.get_Trajectory(&Trajectory)	Получить свойство
Object.put_Trajectory(Trajectory)	Установить свойство

Интерфейс IMate3DTransmission

Интерфейс сопряжения Механическая передача.

Иерархия:

IDispatch

IKompasAPIObject

IMateConstraint3D

IMate3DTransmission

Данный интерфейс может быть получен от интерфейса коллекции сопряжений IMateConstraints3D с помощью свойства IMateConstraints3D::MateConstraint3D или метода IMateConstraints3D::Add с последующим приведением интерфейса IMateConstraint3D к данному интерфейсу с помощью метода IUnknown::QueryInterface

IMate3DTransmission – свойства

Direction1 – Направление движения компонента 1. TRUE – направление 1, FALSE – направление 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction1 = Object.Direction1	Получить свойство (*)
Object.Direction1 = Direction1	Установить свойство (*)

Direction1 = Object.GetDirection1()	Получить свойство (**)
Object.SetDirection1(Direction1)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction1(&Direction1)	Получить свойство
Object.put_Direction1(Direction1)	Установить свойство

Direction2 – Направление движения компонента 2. TRUE – направление 1, FALSE – направление 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction2 = Object.Direction2	Получить свойство (*)
Object.Direction2 = Direction2	Установить свойство (*)
Direction2 = Object.GetDirection2()	Получить свойство (**)
Object.SetDirection2(Direction2)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction2(&Direction2)	Получить свойство
Object.put_Direction2(Direction2)	Установить свойство

MotionType1 – Тип движения компонента 1

Интерфейс...

Тип данных: из перечисления ksMateMotionTypeEnum

Синтаксис Automation:

MotionType1 = Object.MotionType1	Получить свойство (*)
Object.MotionType1 = MotionType1	Установить свойство (*)

```
MotionType1 =                    Получить свойство (**)
Object.GetMotionType1()
Object.SetMotionType1(           Установить свойство (**)
MotionType1 )
```

Синтаксис COM:

```
Object.get_MotionType1(          Получить свойство
&MotionType1 )
Object.put_MotionType1(          Установить свойство
MotionType1 )
```

MotionType2 – Тип движения компонента 2

Интерфейс...

Тип данных: из перечисления ksMateMotionTypeEnum

Синтаксис Automation:

```
MotionType2 =                    Получить свойство (* )
Object.MotionType2
Object.MotionType2 =             Установить свойство (* )
MotionType2
MotionType2 =                    Получить свойство (**)
Object.GetMotionType2()
Object.SetMotionType2(           Установить свойство (**)
MotionType2 )
```

Синтаксис COM:

```
Object.get_MotionType2(          Получить свойство
&MotionType2 )
Object.put_MotionType2(          Установить свойство
MotionType2 )
```

RotationAxis1 – Ось вращения компонента 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
RotationAxis1 =                  Получить свойство (* )
Object.RotationAxis1
Object.RotationAxis1 =           Установить свойство (* )
RotationAxis1
```

RotationAxis1 = Object.GetRotationAxis1()	Получить свойство (**)
Object.SetRotationAxis1(RotationAxis1)	Установить свойство (**)

Синтаксис COM:

Object.get_RotationAxis1(&RotationAxis1)	Получить свойство
Object.put_RotationAxis1(RotationAxis1)	Установить свойство

RotationAxis2 – Ось вращения компонента 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

RotationAxis2 = Object.RotationAxis2	Получить свойство (*)
Object.RotationAxis2 = RotationAxis2	Установить свойство (*)
RotationAxis2 = Object.GetRotationAxis2()	Получить свойство (**)
Object.SetRotationAxis2(RotationAxis2)	Установить свойство (**)

Синтаксис COM:

Object.get_RotationAxis2(&RotationAxis2)	Получить свойство
Object.put_RotationAxis2(RotationAxis2)	Установить свойство

Scale1 – Коэффициент соотношения для 1–го компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale1 = Object.Scale1	Получить свойство (*)
Scale1 = Object.GetScale1()	Получить свойство (**)

Синтаксис COM:

Object.get_Scale1(&Scale1) Получить свойство

Примечание:

Свойство доступно только для чтения.

Scale2 – Коэффициент соотношения для 2-го компонента

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Scale2 = Object.Scale2              Получить свойство (* )
Scale2 = Object.GetScale2()         Получить свойство (**)
```

Синтаксис COM:

Object.get_Scale1(&Scale2) Получить свойство

Примечание:

Свойство доступно только для чтения.

Trajectory1 – Траектория перемещения компонента 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Trajectory1 = Object.Trajectory1          Получить свойство (* )
Object.Trajectory1 = Trajectory1          Установить свойство (* )
Trajectory1 = Object.GetTrajectory1()     Получить свойство (**)
Object.SetTrajectory1( Trajectory1 ))     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Trajectory1( &Trajectory1    Получить свойство
)
Object.put_Trajectory1(  Trajectory1 )  Установить свойство
```

Trajectory2 – Траектория перемещения компонента 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Trajectory2 =                   Получить свойство (* )
Object.Trajectory2
```

Object.Trajectory2 = Trajectory2	Установить свойство (*)
Trajectory2 = Object.GetTrajectory2()	Получить свойство (**)
Object.SetTrajectory2(Trajectory2)	Установить свойство (**)

Синтаксис COM:

Object.get_Trajectory2(&Trajectory2)	Получить свойство
Object.put_Trajectory2(Trajectory2)	Установить свойство

IMate3DTransmission – методы

SetScale – Установить коэффициенты соотношений компонентов

Интерфейс...

Синтаксис Automation:

BOOL SetScale(double Scale1, double Scale2);

Синтаксис COM:

HRESULT SetScale(double Scale1, double Scale2, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Scale1	- коэффициент 1,
Scale2	- коэффициент 2.

Интерфейс IMoldCavity

Интерфейс операции вычитания компонентов.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IMoldCavity

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям IMoldCavities::Add или свойства IMoldCavities::MoldCavity.

КОМПАС версия v18

IMoldCavity – свойства

Parts – Объединяемые компоненты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Parts = Object.Parts	Получить свойство (*)
Object.Parts = Parts	Установить свойство (*)
Parts = Object.GetParts()	Получить свойство (**)
Object.SetParts(Parts)	Установить свойство (**)

Синтаксис COM:

Object.get_Parts(&Parts)	Получить свойство
Object.put_Parts(Parts)	Установить свойство

Примечание:

Список объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство (*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

ScaleCentre – Точка центра масштабирования

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

ScaleCentre = Object.ScaleCentre	Получить свойство (*)
Object.ScaleCentre = ScaleCentre	Установить свойство (*)
ScaleCentre = Object.GetScaleCentre()	Получить свойство (**)

Object.SetScaleCentre(ScaleCentre) Установить свойство (**)

Синтаксис COM:

Object.get_ScaleCentre(&ScaleCentre) Получить свойство
Object.put_ScaleCentre(ScaleCentre) Установить свойство

Интерфейс IMoldCavities

Интерфейс коллекции операций вычитания компонентов.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IMoldCavities

КОМПАС версия v18

IMoldCavities – свойства

MoldCavity – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IMoldCavity.

Синтаксис Automation:

MoldCavity = Object.MoldCavity(Index); Получить свойство (*)
MoldCavity = Object.GetMoldCavity(Index); Получить свойство (**)

Синтаксис COM:

Object.get_MoldCavity(Index, &MoldCavity); Получить свойство

Входные параметры:

VARIANT Index - индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IMoldCavities – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IMoldCavity ** Result);

Возвращаемое значение:

- указатель на интерфейс IMoldCavity.

Примечания:

Метод позволяет создать новый интерфейс операции вычитания компонентов.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IUnionComponents

Интерфейс операции объединения компонентов.

Иерархия:

IDispatch

IKompasAPIObject

IModelObjects

IUnionComponents

Данный интерфейс можно получить с помощью метода коллекции операций объединение компонентов IUnionsComponents::Add или свойства IUnionsComponents::UnionComponents.

КОМПАС версия v18

IUnionComponents – свойства

Parts – Объединяемые компоненты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Parts = Object.Parts	Получить свойство (*)
Object.Parts = Parts	Установить свойство (*)
Parts = Object.GetParts()	Получить свойство (**)
Object.SetParts(Parts)	Установить свойство (**)

Синтаксис COM:

Object.get_Parts(&Parts)	Получить свойство
Object.put_Parts(Parts)	Установить свойство

Примечание:

Список объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Интерфейс IUnionsComponents

Интерфейс коллекции операций объединение компонентов.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IUnionsComponents

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::UnionsComponents.

КОМПАС версия v18

IUnionsComponents – свойства

UnionComponents – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IUnionComponents.

Синтаксис Automation:

UnionComponents = Object.UnionComponents(Index);	Получить свойство (*)
UnionComponents = Object.GetUnionComponents(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_UnionCompone nts(Index, &UnionComponents);	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IUnionsComponents – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IUnionComponents ** Result);

Возвращаемое значение:

	- указатель на интерфейс IUnionComponents.

Примечания:

Метод позволяет создать новый интерфейс операции объединения компонентов.

После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Макрообъекты 3D

Интерфейс IMacroObjects3D

Интерфейс коллекции макроэлементов 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IMacroObjects3D

Данный интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::MacroObjects3D.

КОМПАС версия v18

IMacroObjects3D – свойства

MacroObject3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IMacroObject3D.

Синтаксис Automation:

MacroObject3D = Object.MacroObject3D(Index);	Получить свойство (*)
MacroObject3D = Object.GetMacroObject3D(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_MacroObject3D(Index, &MacroObject3D);	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IMacroObjects3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IMacroObject3D ** Result);

Возвращаемое значение:

- указатель на интерфейс IMacroObject3D.

Примечания:

Метод позволяет создать новый интерфейс макрообъекта.

После получения нового интерфейса нужно задать параметры объекта и вызвать метод IModelObject::Update.

Интерфейс IMacroObject3D

Интерфейс макроэлемента 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IMacroObject3D

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям IMacroObjects3D::Add или свойства IMacroObjects3D::MacroObject3D.

КОМПАС версия v18

IMacroObject3D – свойства

AssociationObject – Установить опорный объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

AssociationObject = Object.AssociationObject(Index)	Получить свойство (*)
Object.AssociationObject(Index) = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject(Index)	Получить свойство (**)
Object.SetAssociationObject(Index, AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(Index, &AssociationObject)	Получить свойство
Object.put_AssociationObject(Index, AssociationObject)	Установить свойство

Входные параметры:

long Index	- индекс объекта.

AssociationObjectCount – Количество опорных объектов

Интерфейс...

Тип данных: long

Синтаксис Automation:

AssociationObjectCount = Object.AssociationObjectCount ;	Получить свойство (*)
AssociationObjectCount = Object.GetAssociationObjectCount();	Получить свойство (**)

Синтаксис COM:

Object.get_AssociationObjectCount(&AssociationObjectCount);	Получить свойство

Примечание:

Свойство доступно только для чтения.

DoubleClickEditable – Редактирование по двойному клику поддерживается

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DoubleClickEditable = Object.DoubleClickEditable	Получить свойство (*)
Object.DoubleClickEditable = DoubleClickEditable	Установить свойство (*)
DoubleClickEditable = Object.GetDoubleClickEditable()	Получить свойство (**)
Object.SetDoubleClickEditable(DoubleClickEditable)	Установить свойство (**)

Синтаксис COM:

Object.get_DoubleClickEditable(&DoubleClickEditable)	Получить свойство
Object.put_DoubleClickEditable(DoubleClickEditable)	Установить свойство

Objects – Внутренние объекты макро

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Objects = Object.Objects	Получить свойство (*)
Object.Objects = Objects	Установить свойство (*)
Objects = Object.GetObjects()	Получить свойство (**)
Object.SetObjects(Objects)	Установить свойство (**)

Синтаксис COM:

Object.get_Objects(&Objects)	Получить свойство
Object.put_Objects(Objects)	Установить свойство

Примечание:

Список объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Если объект один, возвращается VARIANT с типом VT_DISPATCH.

PropertyObjectEditable – Поддерживается интерфейс внешних свойств объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PropertyObjectEditable = Object.PropertyObjectEditable	=	Получить свойство (*)
Object.PropertyObjectEditable = PropertyObjectEditable	=	Установить свойство (*)
PropertyObjectEditable = Object.GetPropertyObjectEditable()	=	Получить свойство (**)
Object.SetPropertyObjectEditable(PropertyObjectEditable)		Установить свойство (**)

Синтаксис COM:

Object.get_PropertyObjectEditable(&PropertyObjectEditable)	Получить свойство
Object.put_PropertyObjectEditable(PropertyObjectEditable)	Установить свойство

StaffVisible – Управление видимостью состава

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StaffVisible = Object.StaffVisible	=	Получить свойство (*)
Object.StaffVisible = StaffVisible	=	Установить свойство (*)
StaffVisible = Object.GetStaffVisible()	=	Получить свойство (**)
Object.SetStaffVisible(StaffVisible)		Установить свойство (**)

Синтаксис COM:

Object.get_StaffVisible(&StaffVisible)	Получить свойство
Object.put_StaffVisible(StaffVisible)	Установить свойство

IMacroObject3D – методы

ClearAssociationObject – Удалить все опорные объекты

Интерфейс...

Синтаксис Automation:

BOOL ClearAssociationObject();

Синтаксис COM:

HRESULT ClearAssociationObject(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Destroy – Разрушить макроэлемент

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Листовое тело

Интерфейс ISheetMetalContainer

Интерфейс контейнера объектов гибки.

Иерархия:

IDispatch

ISheetMetalContainer

Описание:

ISheetMetalContainer является дополнительным к интерфейсу IPart7 и позволяет работать с коллекциями объектов гибки (листовые тела, сгибы и т.д). Данный интерфейс можно получить от IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ISheetMetalContainer – свойства

SheetMetalBends – Указатель на коллекцию сгибов

Интерфейс...

Тип данных: указатель на интерфейс коллекции сгибов ISheetMetalBends

Синтаксис Automation:

ISheetMetalBends iObject.SheetMetalBends; = Получить свойство (*)

ISheetMetalBends iObject.GetSheetMetalBends(); = Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalBends();	Получить свойство

Примечание.

Свойство доступно только для чтения.

SheetMetalBendedStraightens – Коллекция операций Согнуть/Разогнуть

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalBendedStraightens.

Синтаксис Automation:

SheetMetalBendedStraightens Object.SheetMetalBendedStraightens;	=	Получить свойство (*)
SheetMetalBendedStraightens Object.GetSheetMetalBendedStraightens();	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalBendedStraightens(&SheetMetalBendedStraightens);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalBendUnfoldParameters – Параметры развертки

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalBendUnfoldParameters.

Синтаксис Automation:

SheetMetalBendUnfoldParameters Object.SheetMetalBendUnfoldParameters;	=	Получить свойство (*)
SheetMetalBendUnfoldParameters Object.GetSheetMetalBendUnfoldParameters();	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalBendUnfoldParameters(&SheetMetalBendUnfoldParameters);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalBodies – Указатель на коллекцию листовых тел

Интерфейс...

Тип данных: указатель на интерфейс коллекции листовых тел ISheetMetalBodies

Синтаксис Automation:

ISheetMetalBodies iObject.SheetMetalBodies;	=	Получить свойство (*)
ISheetMetalBodies iObject.GetSheetMetalBodies();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalBodies();	Получить свойство

Примечание.

Свойство доступно только для чтения.

SheetMetalClosedCorners – Коллекция замыканий углов

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalClosedCorners.

Синтаксис Automation:

SheetMetalClosedCorners Object.SheetMetalClosedCorners;	=	Получить свойство (*)
SheetMetalClosedCorners Object.GetSheetMetalClosedCorners() ;	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalClosedCorners(&SheetMetalClosedCorners);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalCuts – Указатель на коллекцию элементов листового тела "вырез"

Интерфейс...

Тип данных: указатель на интерфейс коллекции элементов "вырез" ISheetMetalCuts

Синтаксис Automation:

```
ISheetMetalCuts iObject.SheetMetalCuts;     = Получить свойство (* )
ISheetMetalCuts iObject.GetSheetMetalCuts(); = Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_SheetMetalCuts();   Получить свойство
```

Примечание.

Свойство доступно только для чтения.

SheetMetalHoles – Указатель на коллекцию элементов "отверстие"

Интерфейс...

Тип данных: указатель на интерфейс коллекции элементов "отверстие" ISheetMetalHoles.

Синтаксис Automation:

```
ISheetMetalHoles iObject.SheetMetalHoles;     = Получить свойство (* )
ISheetMetalHoles iObject.GetSheetMetalHoles(); = Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_SheetMetalHoles();   Получить свойство
```

Примечание.

Свойство доступно только для чтения.

Свойство доступно только для чтения.

SheetMetalJalousies – Коллекция операций Жалюзи

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalJalousies.

Синтаксис Automation:

```
SheetMetalJalousies = Object.SheetMetalJalousies;          Получить свойство (* )
SheetMetalJalousies = Object.GetSheetMetalJalousies();     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_SheetMetalJalousies( &SheetMetalJalousies );    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

SheetMetalLineBends – Указатель на коллекцию объектов "сгиб по линии"

Интерфейс...

Тип данных: указатель на интерфейс коллекции сгибов по линии ISheetMetalLineBends

Синтаксис Automation:

```
ISheetMetalLineBends = iObject.SheetMetalLineBends;      Получить свойство (* )
ISheetMetalLineBends = iObject.GetSheetMetalLineBends();  Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_SheetMetalLineBends();   Получить свойство
```

Примечание.

Свойство доступно только для чтения.

SheetMetalRuledShells – Коллекция обечаек

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalBodies.

Синтаксис Automation:

```
SheetMetalRuledShells = Object.SheetMetalRuledShells      Получить свойство (* )
SheetMetalRuledShells = Object.GetSheetMetalRuledShells()  Получить свойство (**)
```

Синтаксис COM:

Object.get_SheetMetalRuledShells(&SheetMetalRuledShells)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

SheetMetalLinearRuledShells – Вторая коллекция обечаек

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalBodies.

Синтаксис Automation:

SheetMetalLinearRuledShells = Object.SheetMetalLinearRuledShells	=	Получить свойство (*)
SheetMetalLinearRuledShells = Object.GetSheetMetalLinearRuledShells()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalLinearRuledShells(&SheetMetalLinearRuledShells)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

SheetMetalPlates – Коллекция пластин

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalPlates.

Синтаксис Automation:

SheetMetalPlates = Object.SheetMetalPlates;	Получить свойство (*)
SheetMetalPlates = Object.GetSheetMetalPlates();	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalPlates(&SheetMetalPlates);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalPressFormings – Коллекция операций Открытая/ Закрытая штамповка

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalPressFormings.

Синтаксис Automation:

SheetMetalPressFormings Object.SheetMetalPressFormings;	=	Получить свойство (*)
SheetMetalPressFormings Object.GetSheetMetalPressFormings();	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalPressFormings(&SheetMetalPressFormings);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalRibs – Коллекция операций Ребро усиления

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalRibs.

Синтаксис Automation:

SheetMetalRibs = Object.SheetMetalRibs;	Получить свойство (*)
SheetMetalRibs = Object.GetSheetMetalRibs();	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalRibs(&SheetMetalRibs);	Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalShoulders – Коллекция операций Буртик

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalShoulders.

Синтаксис Automation:

SheetMetalShoulders = Object.SheetMetalShoulders;	Получить свойство (*)

SheetMetalShoulders = Object.GetSheetMetalShoulders(); Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalShoulders(&SheetMetalShoulders); Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalSketchBends – Коллекция сгибов по эскизу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalSketchBends.

Синтаксис Automation:

SheetMetalSketchBends = Object.SheetMetalSketchBends; Получить свойство (*)
SheetMetalSketchBends = Object.GetSheetMetalSketchBends(); Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalSketchBends(&SheetMetalSketchBends); Получить свойство

Примечание:

Свойство доступно только для чтения.

SheetMetalUndercuts – Коллекция подсечек

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalLineBends.

Синтаксис Automation:

SheetMetalUndercuts = Object.SheetMetalUndercuts; Получить свойство (*)
SheetMetalUndercuts = Object.GetSheetMetalUndercuts(); Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalUndercuts(&SheetMetalUndercuts); Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ISheetMetalBends

Справка системы КОМПАС...

kompas.chm::/Sgiby.htm

Интерфейс коллекции сгибов.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalBends

Примечание.

Получить интерфейс можно, используя свойство контейнера тел гибки ISheetMetalContainer::SheetMetalBends

ISheetMetalBends – свойства

SheetMetalBend – Сгиб, заданный по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/Sgiby.htm

Тип данных – указатель на интерфейс ISheetMetalBend.

Синтаксис Automation:

SheetMetalBend iObject.SheetMetalBend(index)	=	Получить свойство (*)
SheetMetalBend iObject.GetSheetMetalBend(index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalBend(index, &SheetMetalBend)	Получить свойство

Входные параметры:

VARIANT index	- индекс операции.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISheetMetalBends – методы

Add – Создать сгиб (добавить сгиб в коллекцию)

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/Sgiby.htm`

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalBend** Result);

Возвращаемое значение:

Указатель на интерфейс сгиба ISheetMetalBend	- в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый интерфейс сгиба.
2. После получения нового интерфейса нужно задать параметры листового тела и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalBodies

Справка системы КОМПАС...

`kompas.chm::/786_Glava93_Listovoe_telo.htm`

Интерфейс коллекции листовых тел.

Иерархия:

IDispatch
  IKompasAPIObject
    IKompasCollection
      IModelObjects
        ISheetMetalBodies

Примечание.

Получить интерфейс можно, используя свойство контейнера тел гибки ISheetMetalContainer::SheetMetalBodies.

ISheetMetalBodies – свойства

SheetMetalBody – Листовое тело, заданное по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/786_Glava93_Listovoe_telo.htm

Тип данных: указатель на интерфейс ISheetMetalBody

Синтаксис Automation:

SheetMetalBody iObject.SheetMetalBody(index)	=	Получить свойство (*)
SheetMetalBody iObject.GetSheetMetalBody(index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalBody(index, &SheetMetalBody)	Получить свойство

Входные параметры:

VARIANT index	- индекс операции выдавливания.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISheetMetalBodies – методы

Add – Создать листовое тело

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/786_Glava93_Listovoe_telo.htm

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalBody** Result);

Возвращаемое значение:

Указатель на интерфейс листового тела ISheetMetalBody - в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый интерфейс базового листового тела.
2. После получения нового интерфейса нужно задать параметры листового тела и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalCuts

Справка системы КОМПАС...

kompas.chm::/CM_SHMT_CUT.htm

Интерфейс коллекции элементов листового тела "вырез".

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
ISheetMetalCuts

Примечание.

Получить интерфейс можно, используя свойство контейнера тел гибки ISheetMetalContainer::SheetMetalCuts.

ISheetMetalCuts – свойства

SheetMetalCut – Объект "вырез", заданный по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/CM_SHMT_HOLE.htm

Тип данных: указатель на интерфейс ISheetMetalCut

Синтаксис Automation:

ISheetMetalCut iObject.SheetMetalCut(index) = Получить свойство (*)

ISheetMetalCut iObject.GetSheetMetalCut(index) = Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalCut(index, &SheetMetalCut) — Получить свойство

Входные параметры:

VARIANT index — индекс объекта.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISheetMetalCuts – методы

Add – Создать объект "вырез"

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/CM_SHMT_HOLE.htm`

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalCut** Result);

Возвращаемое значение:

Указатель на интерфейс отверстия ISheetMetalCut — в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый объект "вырез".
2. После получения нового объекта нужно задать его параметры и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalHoles

Справка системы КОМПАС...

`kompas.chm::/CM_SHMT_HOLE.htm`

Интерфейс коллекции элементов листового тела "отверстие".

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalHoles

Примечание.

Получить интерфейс можно, используя свойство контейнера тел гибки ISheetMetalContainer::SheetMetalHoles.

ISheetMetalHoles – свойства

SheetMetalHole – Объект "отверстие", заданный по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/CM_SHMT_HOLE.htm`

Тип данных: указатель на интерфейс ISheetMetalHole

Синтаксис Automation:

ISheetMetalHole iObject.SheetMetalHole(index)	=	Получить свойство (*)
ISheetMetalHole iObject.GetSheetMetalHole(index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalHole(index, &SheetMetalHole)	Получить свойство

Входные параметры:

VARIANT index	- индекс объекта.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISheetMetalHoles – методы

Add – Создать объект "отверстие"

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/CM_SHMT_HOLE.htm

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalHole** Result);

Возвращаемое значение:

Указатель на интерфейс отверстия ISheetMetalHole - в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый объект "отверстие".
2. После получения нового объекта нужно задать его параметры и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalLineBends

Справка системы КОМПАС...

kompas.chm::/808_94_5_Sgib_po_linii.htm

Интерфейс коллекции объектов "сгиб по линии".

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
ISheetMetalLineBends

Примечание.

Получить интерфейс можно, используя свойство контейнера тел гибки ISheetMetalContainer::SheetMetalLineBends.

ISheetMetalLineBends – свойства

SheetMetalLineBend – Операция "сгиб по линии", заданная по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/808_94_5_Sgib_po_linii.htm

Тип данных: указатель на интерфейс ISheetMetalLineBend

Синтаксис Automation:

ISheetMetalLineBend iObject.SheetMetalLineBend(index)	=	Получить свойство (*)
ISheetMetalLineBend iObject.GetSheetMetalLineBend(index)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SheetMetalLineBend(index, &SheetMetalLineBend)	Получить свойство

Входные параметры:

VARIANT index	- индекс операции.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISheetMetalLineBends – методы

Add – Создать объект "сгиб по линии" (добавить сгиб в коллекцию)

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/808_94_5_Sgib_po_linii.htm`

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalBend** Result);

Возвращаемое значение:

Указатель на линии ISheetMetalLineBend	- в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый интерфейс сгиба по линии.
2. После получения нового интерфейса нужно задать параметры листового тела и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalBody

Справка системы КОМПАС...

kompas.chm::/Param_list_tela.htm

Интерфейс параметров листового тела.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalBody

Примечание:

Интерфейс можно получить с помощью метода коллекции операций выдавливания ISheetMetalBodies::Add или свойства ISheetMetalBodies::SheetMetalBody.

ISheetMetalBody – свойства

BendCoefficient – Коэффициент нейтрального слоя

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendCoefficient = iObject.BendCoefficient;	=	Получить свойство (*)
iObject.BendCoefficient = BendCoefficient;	=	Установить свойство (*)
BendCoefficient = iObject.GetBendCoefficient;	=	Получить свойство (**)
iObject.SetBendCoefficient(BendCoefficient);		Установить свойство (**)

Синтаксис COM:

iObject->get_BendCoefficient(&BendCoefficient);	Получить свойство
iObject->put_BendCoefficient(BendCoefficient);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - коэффициент нейтрального слоя. Позволяет считывать и устанавливать коэффициент нейтрального слоя.

BendReduction – Уменьшение сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendReduction = iObject.BendReduction;	Получить свойство (*)
iObject.BendReduction = BendReduction;	Установить свойство (*)
BendReduction = iObject.GetBendReduction;	Получить свойство (**)
iObject.SetBendReduction(BendReduction);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendReduction(&BendReduction);	Получить свойство
iObject->put_BendReduction(BendReduction);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - уменьшение сгиба. Позволяет считывать и устанавливать уменьшение сгиба.

BendTablePath – Путь к таблице сгибов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

BendTablePath = iObject.BendTablePath;	Получить свойство (*)
iObject.BendTablePath = BendTablePath;	Установить свойство (*)
BendTablePath = iObject.GetBendTablePath;	Получить свойство (**)
iObject.SetBendTablePath(BendTablePath);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendTablePath(&BendTablePath);	Получить свойство

iObject->put_BendTablePath(BendTablePath);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - из таблицы. Свойство позволяет считывать и устанавливать путь к таблице сгибов.

BendValue – Величина сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendValue = iObject.BendValue;	Получить свойство (*)
iObject.BendValue = BendValue;	Установить свойство (*)
BendValue = iObject.GetBendValue;	Получить свойство (**)
iObject.SetBendValue(BendValue);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendValue(&BendValue);	Получить свойство
iObject->put_BendValue(BendValue);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - величина сгиба. Позволяет считывать и устанавливать величину сгиба.

Depth – Глубина выдавливания

Интерфейс...

Тип данных: double

Синтаксис Automation:

Depth = iObject.Depth(normal);	Получить свойство (*)
iObject.Depth(normal) = Depth;	Установить свойство (*)
Depth = iObject.GetDepth(normal);	Получить свойство (**)

iObject.SetDepth(normal, Depth);	Установить свойство (**)

Синтаксис COM:

iObject->get_Depth(normal, &Depth);	Получить свойство
iObject->put_Depth(normal, Depth);	Установить свойство

Входные параметры:

BOOL normal	- TRUE - глубина выдавливания в прямом направлении, - FALSE - глубина выдавливания в обратном направлении.

Примечание.

Позволяет считывать и устанавливать глубину выдавливания.

DepthObject – Объект, задающий глубину выдавливания

Интерфейс...

Тип данных: указатель на объект IModelObject

Синтаксис Automation:

DepthObject = iObject.DepthObject(normal);	Получить свойство (*)
iObject.DepthObject(normal) = DepthObject;	Установить свойство (*)
DepthObject = iObject.GetDepthObject(normal);	Получить свойство (**)
iObject.SetDepthObject(normal, DepthObject);	Установить свойство (**)

Синтаксис COM:

iObject->get_DepthObject(normal, &DepthObject);	Получить свойство
iObject->put_DepthObject(normal, DepthObject);	Установить свойство

Входные параметры:

BOOL normal	- TRUE - объект выдавливания в прямом направлении, - FALSE - объект выдавливания в обратном направлении.

Примечание.

Позволяет считывать и устанавливать указатель на объект, задающий глубину выдавливания.

Direction – Направление выдавливания

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

Direction = iObject.Direction;	Получить свойство (*)
iObject.Direction = Direction;	Установить свойство (*)
Direction = iObject.GetDirection;	Получить свойство (**)
iObject.SetDirection(Direction);	Установить свойство (**)

Синтаксис COM:

iObject->get_Direction(&Direction);	Получить свойство
iObject->put_Direction(Direction);	Установить свойство

Примечание.

Позволяет считывать и устанавливать направление выдавливания.

ExtrusionType – Тип выдавливания

Интерфейс...

Тип данных: из перечисления ksEndTypeEnum

Синтаксис Automation:

ExtrusionType = iObject.ExtrusionType(normal);	Получить свойство (*)
iObject.ExtrusionType(normal) = ExtrusionType;	Установить свойство (*)

ExtrusionType = iObject.GetExtrusionType(normal);	Получить свойство (**)
iObject.SetExtrusionType(normal, ExtrusionType);	Установить свойство (**)

Синтаксис COM:

iObject->get_ExtrusionType(&ExtrusionType);	Получить свойство
iObject->put_ExtrusionType(ExtrusionType);	Установить свойство

Входные параметры:

BOOL normal	- TRUE - тип выдавливания в прямом направлении, - FALSE - тип выдавливания в обратном направлении.

Примечание.

Позволяет считывать и устанавливать тип выдавливания.

Radius – Радиус сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

Radius = iObject.Radius;	Получить свойство (*)
iObject.Radius = Radius;	Установить свойство (*)
Radius = iObject.GetRadius;	Получить свойство (**)
iObject.SetRadius(Radius);	Установить свойство (**)

Синтаксис COM:

iObject->get_Radius(&Radius);	Получить свойство
iObject->put_Radius(Radius);	Установить свойство

Примечание.

Позволяет считывать и устанавливать радиус сгиба.

Sketch – Эскиз

Интерфейс...

Тип данных: указатель на интерфейс ISketch

Синтаксис Automation:

```
Sketch = iObject.Sketch;                Получить свойство (* )
iObject.Sketch = Sketch;                Установить свойство (* )
Sketch = iObject.GetSketch;             Получить свойство (**)
iObject.SetSketch( Sketch );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Sketch(                    Получить свойство
&Sketch );
iObject->put_Sketch(                    Установить свойство
Sketch );
```

Примечание.

Позволяет установить новый эскиз для операции и получить указатель на текущий эскиз.

Straighten – Разогнуть тело

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Straighten = iObject.Straighten;        Получить свойство (* )
iObject.Straighten = Straighten;        Установить свойство (* )
Straighten = iObject.GetStraighten;     Получить свойство (**)
iObject.SetStraighten( Straighten );    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Straighten(                Получить свойство
&Straighten );
iObject->put_Straighten(                Установить свойство
Straighten );
```

Примечание.

Позволяет разгибать и сгибать тело.

Thickness – Толщина листового тела

Интерфейс...

Тип данных: double

Синтаксис Automation:

Thickness = iObject.Thickness( );	Получить свойство (*)
iObject.Thickness( ) = Thickness;	Установить свойство (*)
Thickness = iObject.GetThickness;	Получить свойство (**)
iObject.SetThickness(Thickness);	Установить свойство (**)

Синтаксис COM:

iObject->get_Thickness(&Thickness);	Получить свойство
iObject->put_Thickness(Thickness);	Установить свойство

Примечание.

Позволяет установить и получить толщину листового тела.

ThicknessDirection – Направление для толщины

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ThicknessDirection = iObject.ThicknessDirection;	Получить свойство (*)
iObject.ThicknessDirection = ThicknessDirection;	Установить свойство (*)
ThicknessDirection = iObject.GetThicknessDirection;	Получить свойство (**)
iObject.SetThicknessDirection(ThicknessDirection);	Установить свойство (**)

Синтаксис COM:

iObject->get_ThicknessDirection(&ThicknessDirection);	Получить свойство
iObject->put_ThicknessDirection(ThicknessDirection);	Установить свойство

Примечание.

Позволяет установить и получить направление для толщины.

UnfoldType – Способ определения длины развертки

Интерфейс...

Тип данных: из перечисления ksUnfoldTypeEnum

Синтаксис Automation:

```
UnfoldType = iObject.UnfoldType;     Получить свойство (* )
iObject.UnfoldType = UnfoldType;     Установить свойство (* )
UnfoldType =                         Получить свойство (**)
iObject.GetUnfoldType;
iObject.SetUnfoldType(               Установить свойство (**)
UnfoldType );
```

Синтаксис COM:

```
iObject->get_UnfoldType(     Получить свойство
&UnfoldType );
iObject->put_UnfoldType(     Установить свойство
UnfoldType );
```

Примечание.

Позволяет считывать и устанавливать способ определения длины развертки. Если задан способ определения длины развертки - "из таблицы", то будут недоступны свойства установки (Set) для коэффициента нейтрального слоя, величины сгиба и уменьшения сгиба, т.к они будут браться из базы.

ISheetMetalBody – методы

GetSideParameters – Получить параметры листового тела в одном направлении

Интерфейс...

Синтаксис Automation:

```
BOOL GetSideParameters( BOOL normal,
End_Type * ExtrusionType,
double * Depth,
IModelObject ** DepthObject );
```

Синтаксис COM:

```
HRESULT GetSideParameters( BOOL normal,
End_Type * ExtrusionType,
double * Depth,
IModelObject ** DepthObject );
```

Входные параметры:

normal	- направление: TRUE - в прямом направлении, FALSE - в обратном.

Выходные параметры:

ExtrusionType	- способ построения листового тела,
Depth	- глубина выдавливания,
DepthObject	- указатель на объект, задающий глубину выдавливания.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание.

Параметр Depth в позволяет задать на сколько выдавить за объект, задающий глубину выдавливания, или какое расстояние оставить до объекта, задающего глубину выдавливания - в зависимости от типа построения (для типов построения "до поверхности" и "до вершины" см. перечисление ksEndTypeEnum).

SetSideParameters – Установить параметры листового тела в одном направлении

Интерфейс...

Синтаксис Automation:

```
BOOL SetSideParameters( BOOL normal,
End_Type ExtrusionType,
double Depth,
IModelObject * DepthObject );
```

Синтаксис COM:

```
HRESULT SetSideParameters( BOOL normal,
End_Type ExtrusionType,
double Depth,
IModelObject * DepthObject );
```

Входные параметры:

normal	- направление: TRUE - в прямом направлении, FALSE - в обратном,
ExtrusionType	- способ построения листового тела,
Depth	- глубина выдавливания,

DepthObject	- указатель на объект, задающий глубину выдавливания.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание.

Параметр Depth в позволяет задать на сколько выдавить за объект, задающий глубину выдавливания, или какое расстояние оставить до объекта, задающего глубину выдавливания - в зависимости от типа построения (для типов построения "до поверхности" и "до вершины" см. перечисление ksEndTypeEnum).

Интерфейс ISheetMetalBend

Справка системы КОМПАС...

`kompas.chm::/Sgiby.htm`

Интерфейс параметров сгиба.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalBend

Примечание:

Интерфейс можно получить с помощью метода коллекции сгибов ISheetMetalBends::Add или свойства ISheetMetalBends::SheetMetalBend.

ISheetMetalBend – свойства

Angle – Угол сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle();	Получить свойство (*)
iObject.Angle() = Angle;	Установить свойство (*)
Angle = iObject.GetAngle();	Получить свойство (**)
iObject.SetAngle(Angle);	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle(&Angle);	Получить свойство

```
iObject->put_Angle( Angle );          Установить свойство
```

Примечание.

Позволяет считывать и устанавливать угол сгиба.

AngleType – Способ задания угла – угол сгиба / дополняющий угол

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AngleType = iObject.AngleType();      Получить свойство (* )
iObject.AngleType() = AngleType;      Установить свойство (* )
AngleType =                           Получить свойство (**)
iObject.GetAngleType();
iObject.SetAngleType( AngleType );    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_AngleType(               Получить свойство
&AngleType );
iObject->put_AngleType(               Установить свойство
AngleType );
```

Значения свойства:

```
TRUE      - угол интерпретируется как угол сгиба,
FALSE     - угол интерпретируется как дополняющий
            угол.
```

Примечание.

Позволяет считывать и устанавливать способ задания угла.

BendCoefficient – Коэффициент нейтрального слоя

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BendCoefficient =                     Получить свойство (* )
iObject.BendCoefficient();
iObject.BendCoefficient() =           Установить свойство (* )
BendCoefficient;
BendCoefficient =                     Получить свойство (**)
iObject.GetBendCoefficient();
```

iObject.SetBendCoefficient(BendCoefficient);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendCoefficient(&BendCoefficient);	Получить свойство
iObject->put_BendCoefficient(BendCoefficient);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - коэффициент нейтрального слоя. Позволяет считывать и устанавливать коэффициент нейтрального слоя. Get - свойство будет возвращать коэффициент нейтрального слоя базового листового тела пока не будет установлен коэффициент нейтрального слоя для сгиба.

BendObject – Ребро

Интерфейс...

Тип данных: указатель на интерфейс IModelObject.

Синтаксис Automation:

BendObject = iObject.BendObject();	Получить свойство (*)
iObject.BendObject() = BendObject;	Установить свойство (*)
BendObject = iObject.GetBendObject();	Получить свойство (**)
iObject.SetBendObject(BendObject);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendObject(&BendObject);	Получить свойство
iObject->put_BendObject(BendObject);	Установить свойство

Примечание.

Позволяет получать и устанавливать интерфейс опорного ребра.

BendObjects – Ребра

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BendObjects = Object.BendObjects;	Получить свойство (*)

```
Object.BendObjects = BendObjects;          Установить свойство (* )
BendObjects = Object.GetBendObjects();     Получить свойство (**)
Object.SetBendObjects( BendObjects );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BendObjects( &BendObjects );    Получить свойство
Object.put_BendObjects(  BendObjects );    Установить свойство
```

BendReduction – Уменьшение сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BendReduction =                Получить свойство (* )
iObject.BendReduction();
iObject.BendReduction() =      Установить свойство (* )
BendReduction;
BendReduction =                Получить свойство (**)
iObject.GetBendReduction(
);
iObject.SetBendReduction(      Установить свойство (**)
BendReduction );
```

Синтаксис COM:

```
iObject-                       Получить свойство
>get_BendReduction(
&BendReduction );
iObject-                       Установить свойство
>put_BendReduction(
BendReduction );
```

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - уменьшение сгиба. Позволяет считывать и устанавливать уменьшение сгиба. Get - свойство будет возвращать уменьшение сгиба базового листового тела пока не будет установлено "уменьшение сгиба" для сгиба.

BendRelease – Тип освобождения сгиба

Интерфейс...

Тип данных: из перечисления ksBendReleaseTypeEnum

Синтаксис Automation:

BendRelease = iObject.BendRelease();	Получить свойство (*)
iObject.BendRelease() = BendRelease;	Установить свойство (*)
BendRelease = iObject.GetBendRelease();	Получить свойство (**)
iObject.SetBendRelease(BendRelease);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendRelease(&BendRelease);	Получить свойство
iObject->put_BendRelease(BendRelease);	Установить свойство

Примечание.

Позволяет считывать и устанавливать тип освобождения сгиба. Свойство доступно при включенной опции "использовать освобождение сгиба".

BendTablePath – Путь к таблице сгибов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

BendTablePath = iObject.BendTablePath();	Получить свойство (*)
BendTablePath = iObject.GetBendTablePath();	Получить свойство (**)

Синтаксис COM:

iObject->get_BendTablePath(&BendTablePath);	Получить свойство

Примечание.

Свойство доступно только для чтения. Таблица сгибов одна на документ. Путь к таблице сгибов выбирается у базового листового тела.

BendValue – Величина сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendValue = iObject.BendValue();	Получить свойство (*)
iObject.BendValue() = BendValue;	Установить свойство (*)
BendValue = iObject.GetBendValue();	Получить свойство (**)
iObject.SetBendValue(BendValue);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendValue(&BendValue);	Получить свойство
iObject->put_BendValue(BendValue);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - величина сгиба. Позволяет считывать и устанавливать величину сгиба. Get - свойство будет возвращать величину сгиба базового листового тела пока не будет установлена "величина сгиба" для сгиба.

DeviationLeftSide – Угол на сгибе слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

DeviationLeftSide = iObject.DeviationLeftSide();	Получить свойство (*)
iObject.DeviationLeftSide() = DeviationLeftSide;	Установить свойство (*)
DeviationLeftSide = iObject.GetDeviationLeftSide();	Получить свойство (**)
iObject.SetDeviationLeftSide(DeviationLeftSide);	Установить свойство (**)

Синтаксис COM:

iObject->get_DeviationLeftSide(&DeviationLeftSide);	Получить свойство
iObject->put_DeviationLeftSide(DeviationLeftSide);	Установить свойство

Примечание.

Позволяет считывать и устанавливать угол на сгибе слева при способе построения левой стороны - "по уклону и углу слева".

DeviationRightSide – Угол на сгибе справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

DeviationRightSide = iObject.DeviationRightSide();	Получить свойство (*)
iObject.DeviationRightSide() = DeviationRightSide;	Установить свойство (*)
DeviationRightSide = iObject.GetDeviationRightSide();	Получить свойство (**)
iObject.SetDeviationRightSide(DeviationRightSide);	Установить свойство (**)

Синтаксис COM:

iObject->get_DeviationRightSide(&DeviationRightSide);	Получить свойство
iObject->put_DeviationRightSide(DeviationRightSide);	Установить свойство

Примечание.

Позволяет считывать и устанавливать угол на сгибе справа при способе построения правой стороны - "по уклону и углу справа".

Direction – Направление построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = iObject.Direction();	Получить свойство (*)
iObject.Direction() = Direction;	Установить свойство (*)
Direction = iObject.GetDirection();	Получить свойство (**)
iObject.SetDirection(Direction);	Установить свойство (**)

Синтаксис COM:

```
iObject->get_Direction(          Получить свойство
&Direction );
iObject->put_Direction(          Установить свойство
Direction );
```

Примечание.

Позволяет установить и получить направление построения сгиба.

DismissalAngleType – Способ освобождения угла сгиба

Интерфейс...

Тип данных: из перечисления ksBendAngleReleaseTypeEnum

Синтаксис Automation:

```
DismissalAngleType =             Получить свойство (* )
iObject.DismissalAngleType();
iObject.DismissalAngleType() =   Установить свойство (* )
DismissalAngleType;
DismissalAngleType =             Получить свойство (**)
iObject.GetDismissalAngleType();
iObject.SetDismissalAngleType(   Установить свойство (**)
DismissalAngleType );
```

Синтаксис COM:

```
iObject->get_DismissalAngleType(     Получить свойство
&DismissalAngleType );
iObject->put_DismissalAngleType(     Установить свойство
DismissalAngleType );
```

Примечание.

Позволяет считывать и устанавливать способ освобождения угла сгиба. Свойство доступно при включенной опции "использовать освобождение угла".

DismissalDepth – Глубина разгрузки сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DismissalDepth =                 Получить свойство (* )
iObject.DismissalDepth();
iObject.DismissalDepth() =       Установить свойство (* )
DismissalDepth;
DismissalDepth =                 Получить свойство (**)
iObject.GetDismissalDepth();
```

iObject.SetDismissalDepth(DismissalDepth); Установить свойство (**)

Синтаксис COM:

iObject->get_DismissalDepth(&DismissalDepth); Получить свойство
iObject->put_DismissalDepth(DismissalDepth); Установить свойство

Примечание.

Позволяет считывать и устанавливать глубину разгрузки сгиба. Свойство доступно при включенной опции "использовать освобождение сгиба".

DismissalWidth – Ширина разгрузки сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

DismissalWidth = iObject.DismissalWidth(); Получить свойство (*)
iObject.DismissalWidth() = DismissalWidth; Установить свойство (*)
DismissalWidth = iObject.GetDismissalWidth(); Получить свойство (**)
iObject.SetDismissalWidth(DismissalWidth); Установить свойство (**)

Синтаксис COM:

iObject->get_DismissalWidth(&DismissalWidth); Получить свойство
iObject->put_DismissalWidth(DismissalWidth); Установить свойство

Примечание.

Позволяет считывать и устанавливать ширину разгрузки сгиба. Свойство доступно при включенной опции "использовать освобождение сгиба".

DismissalWithWidth – Учитывать ширину

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DismissalWithWidth = iObject.DismissalWithWidth();	Получить свойство (*)
iObject.DismissalWithWidth() = DismissalWithWidth;	Установить свойство (*)
DismissalWithWidth = iObject.GetDismissalWithWidth();	Получить свойство (**)
iObject.SetDismissalWithWidth(DismissalWithWidth);	Установить свойство (**)

Синтаксис COM:

iObject->get_DismissalWithWidth(&DismissalWithWidth);	Получить свойство
iObject->put_DismissalWithWidth(DismissalWithWidth);	Установить свойство

Значения свойства:

TRUE	- учитывать ширину,
FALSE	- не учитывать.

Примечание.

Позволяет считывать и устанавливать признак - "учитывать ширину". Свойство доступно при включенной опции "использовать освобождение сгиба".

Disposal – Тип размещения сгиба на ребре

Интерфейс...

Тип данных: из перечисления ksBendDisposalEnum

Синтаксис Automation:

Disposal = iObject.Disposal();	Получить свойство (*)
iObject.Disposal() = Disposal;	Установить свойство (*)
Disposal = iObject.GetDisposal();	Получить свойство (**)
iObject.SetDisposal(Disposal);	Установить свойство (**)

Синтаксис COM:

iObject->get_Disposal(&Disposal);	Получить свойство

```
iObject->put_Disposal(          Установить свойство
Disposal );
```

Примечание.

Позволяет установить и получить тип размещения сгиба на ребре.

DistanceLeftSide – Отступ слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DistanceLeftSide = iObject.DistanceLeftSide();      Получить свойство (* )
iObject.DistanceLeftSide() = DistanceLeftSide;      Установить свойство (* )
DistanceLeftSide =                                  Получить свойство (**)
iObject.GetDistanceLeftSide();
iObject.SetDistanceLeftSide( DistanceLeftSide );    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DistanceLeftSide(      Получить свойство
&DistanceLeftSide );
iObject->put_DistanceLeftSide(      Установить свойство
DistanceLeftSide );
```

Примечание.

Позволяет считывать и устанавливать отступ слева.

DistanceRightSide – Отступ справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DistanceRightSide =                 Получить свойство (* )
iObject.DistanceRightSide();
iObject.DistanceRightSide() =       Установить свойство (* )
DistanceRightSide;
DistanceRightSide =                 Получить свойство (**)
iObject.GetDistanceRightSide();
iObject.SetDistanceRightSide(       Установить свойство (**)
DistanceRightSide );
```

Синтаксис COM:

```
iObject->get_DistanceRightSide(     Получить свойство
&DistanceRightSide );
```

```
iObject->put_DistanceRightSide(          Установить свойство
DistanceRightSide );
```

Примечание.

Позволяет считывать и устанавливать отступ справа.

InternalLength – Определение длины по внутреннему контуру сгиба / по касанию к сгибу

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
InternalLength =                         Получить свойство (* )
iObject.InternalLength();
iObject.InternalLength() =               Установить свойство (* )
InternalLength;
InternalLength =                         Получить свойство (**)
iObject.GetInternalLength();
iObject.SetInternalLength(               Установить свойство (**)
InternalLength );
```

Синтаксис COM:

```
iObject->get_InternalLength(             Получить свойство
&InternalLength );
iObject->put_InternalLength(             Установить свойство
InternalLength );
```

Значения свойства:

```
TRUE                                     - по внутреннему контуру сгиба,
FALSE                                    - по касанию к сгибу.
```

Примечание.

Позволяет считывать и устанавливать тип определения длины.

InternalRadius – Внутренний радиус

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
InternalRadius =                         Получить свойство (* )
iObject.InternalRadius();
iObject.InternalRadius() =               Установить свойство (* )
InternalRadius;
```

```
InternalRadius =                    Получить свойство (**)
iObject.GetInternalRadius();
iObject.SetInternalRadius(          Установить свойство (**)
InternalRadius );
```

Синтаксис COM:

```
iObject->get_InternalRadius(        Получить свойство
&InternalRadius );
iObject->put_InternalRadius(        Установить свойство
InternalRadius );
```

Значения свойства:

```
TRUE        - внутренний радиус,
FALSE       - наружный радиус.
```

Примечание.

Позволяет получать и устанавливать признак "внутренний радиус".

LeftSideAngle – Угол слева (уклон)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
LeftSideAngle =                     Получить свойство (* )
iObject.LeftSideAngle();
iObject.LeftSideAngle() =           Установить свойство (* )
LeftSideAngle;
LeftSideAngle =                     Получить свойство (**)
iObject.GetLeftSideAngle();
iObject.SetLeftSideAngle(           Установить свойство (**)
LeftSideAngle );
```

Синтаксис COM:

```
iObject->get_LeftSideAngle(         Получить свойство
&LeftSideAngle );
iObject->put_LeftSideAngle(         Установить свойство
LeftSideAngle );
```

Примечание.

Позволяет считывать и устанавливать уклон слева при способе построения левой стороны - "по уклону и углу слева".

LeftSideType – Способ построения левой боковой стороны

Интерфейс...

Тип данных: из перечисления ksBendSideTypeEnum

Синтаксис Automation:

```
LeftSideType =                  Получить свойство (* )
iObject.LeftSideType();
iObject.LeftSideType() =        Установить свойство (* )
LeftSideType;
LeftSideType =                  Получить свойство (**)
iObject.GetLeftSideType();
iObject.SetLeftSideType(        Установить свойство (**)
LeftSideType );
```

Синтаксис COM:

```
iObject->get_LeftSideType(      Получить свойство
&LeftSideType );
iObject->put_LeftSideType(      Установить свойство
LeftSideType );
```

Примечание.

Позволяет считывать и устанавливать способ построения левой боковой стороны.

Length – Длина прямой части сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = iObject.Length();      Получить свойство (* )
iObject.Length() = Length;      Установить свойство (* )
Length = iObject.GetLength();   Получить свойство (**)
iObject.SetLength( Length );    Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Length(            Получить свойство
&Length );
iObject->put_Length(            Установить свойство
Length );
```

Примечание.

Позволяет считывать и устанавливать длину прямой части сгиба.

LengthBuildingType1 – Способ задания длины продолжения сгиба слева

Интерфейс...

Тип данных: из перечисления ksLengthBuildingTypeEnum

Синтаксис Automation:

```
LengthBuildingType1 = Object.LengthBuildingType1;          Получить свойство (* )
Object.LengthBuildingType1 = LengthBuildingType1;          Установить свойство (* )
LengthBuildingType1 =                                      Получить свойство (**)
Object.GetLengthBuildingType1();
Object.SetLengthBuildingType1( LengthBuildingType1 );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LengthBuildingType1(                            Получить
&LengthBuildingType1 );                                    свойство
Object.put_LengthBuildingType1(                            Установить
LengthBuildingType1 );                                     свойство
```

LengthBuildingType2 – Способ задания длины продолжения сгиба справа

Интерфейс...

Тип данных: из перечисления ksLengthBuildingTypeEnum

Синтаксис Automation:

```
LengthBuildingType2 = Object.LengthBuildingType2;          Получить свойство (* )
Object.LengthBuildingType2 = LengthBuildingType2;          Установить свойство (* )
LengthBuildingType2 =                                      Получить свойство (**)
Object.GetLengthBuildingType2();
Object.SetLengthBuildingType2( LengthBuildingType2 );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LengthBuildingType2(                            Получить
&LengthBuildingType2 );                                    свойство
Object.put_LengthBuildingType2(                            Установить
LengthBuildingType2 );                                     свойство
```

LengthBy2Sides – Задание длины по двум сторонам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LengthBy2Sides = Object.LengthBy2Sides;	Получить свойство (*)
Object.LengthBy2Sides = LengthBy2Sides;	Установить свойство (*)
LengthBy2Sides = Object.GetLengthBy2Sides();	Получить свойство (**)
Object.SetLengthBy2Sides(LengthBy2Sides);	Установить свойство (**)

Синтаксис COM:

Object.get_LengthBy2Sides(&LengthBy2Sides);	Получить свойство
Object.put_LengthBy2Sides(LengthBy2Sides);	Установить свойство

LengthObject1 – Вершина или плоскость для задания левой длины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

LengthObject1 = Object.LengthObject1;	Получить свойство (*)
Object.LengthObject1 = LengthObject1;	Установить свойство (*)
LengthObject1 = Object.GetLengthObject1();	Получить свойство (**)
Object.SetLengthObject1(LengthObject1 ;	Установить свойство (**)

Синтаксис COM:

Object.get_LengthObject1(&LengthObject1);	Получить свойство
Object.put_LengthObject1(LengthObject1);	Установить свойство

LengthObject2 – Вершина или плоскость для задания правой длины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
LengthObject2 =                    Получить свойство (* )
Object.LengthObject2;
Object.LengthObject2 =             Установить свойство (* )
LengthObject2;
LengthObject2 =                    Получить свойство (**)
Object.GetLengthObject2();
Object.SetLengthObject2(           Установить свойство (**)
LengthObject2) ;
```

Синтаксис COM:

```
Object.get_LengthObject2( &LengthObject2 );    Получить свойство
Object.put_LengthObject2(  LengthObject2 );    Установить свойство
```

LengthType – Способ задания длины (прямой части) сгиба

Интерфейс...

Тип данных: из перечисления ksBendLengthTypeEnum

Синтаксис Automation:

```
LengthType =                       Получить свойство (* )
iObject.LengthType();
iObject.LengthType() =             Установить свойство (* )
LengthType;
LengthType =                       Получить свойство (**)
iObject.GetLengthType();
iObject.SetLengthType(             Установить свойство (**)
LengthType );
```

Синтаксис COM:

```
iObject->get_LengthType(     Получить свойство
&LengthType );
iObject->put_LengthType(     Установить свойство
LengthType );
```

Примечание.

Позволяет считывать и устанавливать способ задания длины (прямой части) сгиба.

LengthType1 – Способ задания длины

Интерфейс...

Тип данных: из перечисления ksBendLengthTypeEnum

Синтаксис Automation:

```
LengthType1 = Object.LengthType1;                Получить свойство (* )
Object.LengthType1 = LengthType1;                Установить свойство (* )
LengthType1 = Object.GetLengthType1();           Получить свойство (**)
Object.SetLengthType1( LengthType1 );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LengthType1( &LengthType1 );      Получить свойство
Object.put_LengthType1(  LengthType1 );      Установить свойство
```

LengthType2 – Способ задания длины

Интерфейс...

Тип данных: из перечисления ksBendLengthTypeEnum

Синтаксис Automation:

```
LengthType2 = Object.LengthType2;                Получить свойство (* )
Object.LengthType2 = LengthType2;                Установить свойство (* )
LengthType2 = Object.GetLengthType2();           Получить свойство (**)
Object.SetLengthType2( LengthType2 );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_LengthType2( &LengthType2 );      Получить свойство
Object.put_LengthType2(  LengthType2 );      Установить свойство
```

Length1 – Длина сгиба слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length1 = Object.Length1;                    Получить свойство (* )
Object.Length1 = Length1;                    Установить свойство (* )
Length1 = Object.GetLength1();               Получить свойство (**)
Object.SetLength1( Length1 );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Length1( &Length1 );      Получить свойство
Object.put_Length1(  Length1 );      Установить свойство
```

Примечание:

Свойство используется, если установлен признак LengthBy2Sides.

Length2 – Длина сгиба справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length2 = Object.Length2;                    Получить свойство (* )
Object.Length2 = Length2;                    Установить свойство (* )
Length2 = Object.GetLength2();               Получить свойство (**)
Object.SetLength2( Length2 );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Length2( &Length2 );    Получить свойство
Object.put_Length2(  Length2 );    Установить свойство
```

Примечание:

Свойство используется, если установлен признак LengthBy2Sides.

Offset – Смещение сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Offset = iObject.Offset();       Получить свойство (* )
iObject.Offset() = Offset;       Установить свойство (* )
Offset =                         Получить свойство (**)
iObject.GetOffset();
iObject.SetOffset( Offset );     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Offset( &Offset );    Получить свойство
iObject->put_Offset( Offset );     Установить свойство
```

Примечание.

Позволяет считывать и устанавливать смещение сгиба относительно листового тела.

OffsetDirection1 – Направление смещения от объекта при задании левой длины по вершине или поверхности

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

```
OffsetDirection1 = Object.OffsetDirection1;                Получить свойство (* )
Object.OffsetDirection1 = OffsetDirection1;                Установить свойство (* )
OffsetDirection1 = Object.GetOffsetDirection1();           Получить свойство (**)
Object.SetOffsetDirection1( OffsetDirection1 )             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OffsetDirection1( &OffsetDirection1 );          Получить свойство
Object.put_OffsetDirection1(  OffsetDirection1 );          Установить свойство
```

OffsetDirection2 – Направление смещения от объекта при задании правой длины по вершине или поверхности

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

```
OffsetDirection2 = Object.OffsetDirection2;                Получить свойство (* )
Object.OffsetDirection2 = OffsetDirection2;                Установить свойство (* )
OffsetDirection2 = Object.GetOffsetDirection2();           Получить свойство (**)
Object.SetOffsetDirection2( OffsetDirection2 )             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OffsetDirection2( &OffsetDirection2 );          Получить свойство
Object.put_OffsetDirection2(  OffsetDirection2 );          Установить свойство
```

OffsetFromLengthObject1 – Смещение от объекта при задании левой длины по вершине или поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
OffsetFromLengthObject1 =                                  Получить свойство (* )
Object.OffsetFromLengthObject1;
Object.OffsetFromLengthObject1 =                           Установить свойство (* )
OffsetFromLengthObject1;
OffsetDirection1 = Object.GetOffsetDirection1();           Получить свойство (**)
Object.SetOffsetDirection1( OffsetDirection1 );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OffsetDirection1( &OffsetDirection1 );    Получить свойство
Object.put_OffsetDirection1(  OffsetDirection1 );    Установить свойство
```

OffsetFromLengthObject2 – Смещение от объекта при задании правой длины по вершине или поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
OffsetFromLengthObject2 =                            Получить свойство (* )
Object.OffsetFromLengthObject2;
Object.OffsetFromLengthObject2 =                     Установить свойство (* )
OffsetFromLengthObject2;
OffsetDirection2 = Object.GetOffsetDirection2();     Получить свойство (**)
Object.SetOffsetDirection1( OffsetDirection2 );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OffsetDirection1( &OffsetDirection2 );    Получить свойство
Object.put_OffsetDirection1(  OffsetDirection2 );    Установить свойство
```

OffsetType – Тип смещения

Интерфейс...

Тип данных: из перечисления ksBendOffsetTypeEnum

Синтаксис Automation:

```
OffsetType =                  Получить свойство (* )
iObject.OffsetType();
iObject.OffsetType() =        Установить свойство (* )
OffsetType;
OffsetType =                  Получить свойство (**)
iObject.GetOffsetType();
iObject.SetOffsetType(        Установить свойство (**)
OffsetType );
```

Синтаксис COM:

```
iObject->get_OffsetType(      Получить свойство
&OffsetType );
iObject->put_OffsetType(      Установить свойство
OffsetType );
```

Примечание.

Позволяет считывать и устанавливать тип смещения сгиба относительно листового тела.

Radius – Внутренний радиус

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = iObject.Radius();              Получить свойство (* )
iObject.Radius() = Radius;              Установить свойство (* )
Radius = iObject.GetRadius();           Получить свойство (**)
iObject.SetRadius( Radius );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Radius( &Radius );     Получить свойство
iObject->put_Radius( Radius );      Установить свойство
```

Примечание.

Позволяет получать и устанавливать радиус сгиба.

RightSideAngle – Угол справа (уклон)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
RightSideAngle =                    Получить свойство (* )
iObject.RightSideAngle();
iObject.RightSideAngle() =          Установить свойство (* )
RightSideAngle;
RightSideAngle =                    Получить свойство (**)
iObject.GetRightSideAngle();
iObject.SetRightSideAngle(          Установить свойство (**)
RightSideAngle );
```

Синтаксис COM:

```
iObject->get_RightSideAngle(        Получить свойство
&LeftSideAngle );
iObject->put_RightSideAngle(        Установить свойство
LeftSideAngle );
```

Примечание.

Позволяет считывать и устанавливать уклон справа при способе построения правой стороны - "по уклону и углу справа".

RightSideType – Способ построения правой боковой стороны

Интерфейс...

Тип данных: из перечисления ksBendSideTypeEnum

Синтаксис Automation:

RightSideType = iObject.RightSideType();	Получить свойство (*)
iObject.RightSideType() = RightSideType;	Установить свойство (*)
RightSideType = iObject.GetRightSideType();	Получить свойство (**)
iObject.SetRightSideType(RightSideType);	Установить свойство (**)

Синтаксис COM:

iObject->get_RightSideType(&RightSideType);	Получить свойство
iObject->put_RightSideType(RightSideType);	Установить свойство

Примечание.

Позволяет считывать и устанавливать способ построения правой боковой стороны.

Straighten – Разогнуть сгиб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Straighten = iObject.Straighten();	Получить свойство (*)
iObject.Straighten() = Straighten;	Установить свойство (*)
Straighten = iObject.GetStraighten();	Получить свойство (**)
iObject.SetStraighten(Straighten);	Установить свойство (**)

Синтаксис COM:

```
iObject->get_Straighten(        Получить свойство
&Straighten );
iObject->put_Straighten(        Установить свойство
Straighten );
```

Значения свойства:

```
TRUE                    - сгиб разогнут,
FALSE                   - сгиб согнут.
```

Примечание.

Позволяет сгибать и разгибать сгиб.

WideningLeftSide – Расширение слева

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
WideningLeftSide =              Получить свойство (* )
iObject.WideningLeftSide();
iObject.WideningLeftSide() =    Установить свойство (* )
WideningLeftSide;
WideningLeftSide =              Получить свойство (**)
iObject.GetWideningLeftSide();
iObject.SetWideningLeftSide(    Установить свойство (**)
WideningLeftSide );
```

Синтаксис COM:

```
iObject->get_WideningLeftSide(      Получить свойство
&WideningLeftSide );
iObject->put_WideningLeftSide(      Установить свойство
WideningLeftSide );
```

Примечание.

Позволяет считывать и устанавливать расширение слева при способе построения левой стороны - "расширение сгиба слева".

WideningRightSide – Расширение справа

Интерфейс...

Тип данных: double

Синтаксис Automation:

WideningRightSide = iObject.WideningRightSide();	Получить свойство (*)
iObject.WideningRightSide() = WideningRightSide;	Установить свойство (*)
WideningRightSide = iObject.GetWideningRightSide();	Получить свойство (**)
iObject.SetWideningRightSide(WideningRightSide);	Установить свойство (**)

Синтаксис COM:

iObject->get_WideningRightSide(&WideningRightSide);	Получить свойство
iObject->put_WideningRightSide(WideningRightSide);	Установить свойство

Примечание.

Позволяет считывать и устанавливать расширение справа при способе построения правой стороны - "расширение сгиба справа".

WithoutAngleRelease – Без освобождения углов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithoutAngleRelease = iObject.WithoutAngleRelease();	Получить свойство (*)
iObject.WithoutAngleRelease() = WithoutAngleRelease;	Установить свойство (*)
WithoutAngleRelease = iObject.GetWithoutAngleRelease();	Получить свойство (**)
iObject.SetWithoutAngleRelease(WithoutAngleRelease);	Установить свойство (**)

Синтаксис COM:

iObject->get_WithoutAngleRelease(&WithoutAngleRelease);	Получить свойство
iObject->put_WithoutAngleRelease(WithoutAngleRelease);	Установить свойство

Значения свойства:

TRUE	- без освобождения углов,
FALSE	- с освобождением.

Примечание.

Позволяет считывать и устанавливать признак - "без освобождения углов".

WithoutBendRelease – Без освобождения сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithoutBendRelease = iObject.WithoutBendRelease();	Получить свойство (*)
iObject.WithoutBendRelease() = WithoutBendRelease;	Установить свойство (*)
WithoutBendRelease = iObject.GetWithoutBendRelease();	Получить свойство (**)
iObject.SetWithoutBendRelease(WithoutBendRelease);	Установить свойство (**)

Синтаксис COM:

iObject->get_WithoutBendRelease(&WithoutAngleRelease);	Получить свойство
iObject->put_WithoutBendRelease(WithoutAngleRelease);	Установить свойство

Значения свойства:

TRUE	- без освобождения сгиба,
FALSE	- с освобождением.

Примечание.

Позволяет считывать и устанавливать признак - "без освобождения сгиба".

UnfoldType – Способ определения длины развертки

Интерфейс...

Тип данных: из перечисления ksUnfoldTypeEnum

Синтаксис Automation:

UnfoldType = iObject.UnfoldType();	Получить свойство (*)
iObject.UnfoldType() = UnfoldType;	Установить свойство (*)
UnfoldType = iObject.GetUnfoldType();	Получить свойство (**)
iObject.SetUnfoldType(UnfoldType);	Установить свойство (**)

Синтаксис COM:

```
iObject->get_UnfoldType(          Получить свойство
&UnfoldType );
iObject->put_UnfoldType(          Установить свойство
UnfoldType );
```

Примечание.

Позволяет считывать и устанавливать способ определения длины развертки. Если задан способ определения длины развертки - из таблицы, то будут недоступны свойства установки (Set) для коэффициента нейтрального слоя, величины сгиба и уменьшения сгиба, так как они будут браться из базы.

Width – Ширина сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = iObject.Width();          Получить свойство (* )
iObject.Width() = Width;          Установить свойство (* )
Width = iObject.GetWidth();       Получить свойство (**)
iObject.SetWidth( Width );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Width( &Width );     Получить свойство
iObject->put_Width( Width );      Установить свойство
```

Примечание.

Позволяет получить и установить ширину сгиба.

Интерфейс ISheetMetalBendedStraighten

Интерфейс операции Согнуть/Разогнуть.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalBendedStraighten

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям ISheetMetalBendedStraightens::Add или свойства ISheetMetalBendedStraightens::SheetMetalBendedStraighten.

КОМПАС версия v18

ISheetMetalBendedStraighten – свойства

BendObjects – Сгибы. Массив объектов в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BendObjects = Object.BendObjects;	Получить свойство (*)
Object.BendObjects = BendObjects;	Установить свойство (*)
BendObjects = Object.GetBendObjects();	Получить свойство (**)
Object.SetBendObjects(BendObjects);	Установить свойство (**)

Синтаксис COM:

Object.get_BendObjects(&BendObjects);	Получить свойство
Object.put_BendObjects(BendObjects);	Установить свойство

FixedFace – Неподвижная грань

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

FixedFace = Object.FixedFace;	Получить свойство (*)
Object.FixedFace = FixedFace;	Установить свойство (*)
FixedFace = Object.GetFixedFace();	Получить свойство (**)
Object.SetFixedFace(FixedFace);	Установить свойство (**)

Синтаксис COM:

Object.get_FixedFace(&FixedFace);	Получить свойство
Object.put_FixedFace(FixedFace);	Установить свойство

FoldLinesEnabled – Отображения линий сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
FoldLinesEnabled = Object.FoldLinesEnabled          Получить свойство (* )
Object.FoldLinesEnabled = FoldLinesEnabled          Установить свойство (* )
FoldLinesEnabled = Object.GetFoldLinesEnabled()     Получить свойство (**)
Object.SetFoldLinesEnabled( FoldLinesEnabled )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_FoldLinesEnabled( &FoldLinesEnabled )    Получить свойство
Object.put_FoldLinesEnabled(  FoldLinesEnabled )    Установить свойство
```

Версия Компас v18.1

FoldLinesStyle – Стиль линии

Интерфейс...

Тип данных: из перечисления ksCurveStyleEnum.

Синтаксис Automation:

```
FoldLinesStyle = Object.FoldLinesStyle              Получить свойство (* )
Object.FoldLinesStyle = FoldLinesStyle              Установить свойство (* )
FoldLinesStyle = Object.GetFoldLinesStyle()         Получить свойство (**)
Object.SetFoldLinesStyle( FoldLinesStyle )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_FoldLinesStyle( &FoldLinesStyle )    Получить свойство
Object.put_FoldLinesStyle(  FoldLinesStyle )    Установить свойство
```

Версия Компас v18.1

Интерфейс ISheetMetalBendedStraightens

Интерфейс коллекции элементов "Согнуть/Разогнуть"

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ISheetMetalBendedStraightens

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalBendedStraightens.

КОМПАС версия v18

ISheetMetalBendedStraightens – свойства

SheetMetalBendedStraighten – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalBendedStraighten.

Синтаксис Automation:

SheetMetalBendedStraighten Object.SheetMetalBendedStraighten(Index)	=	Получить свойство (*)
SheetMetalBendedStraighten Object.GetSheetMetalBendedStraighten(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalBendedStraighten(Index, &SheetMetalBendedStraighten)	Получить свойство

Входные параметры:

VARIANT index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalBendedStraightens – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum Type, ISheetMetalBendedStraighten * * Result);

Возвращаемое значение:

- указатель на интерфейс ISheetMetalBendedStraighten.

Входные параметры:

Type - тип объекта из перечисления ksObj3dTypeEnum.

Примечания:

1. Метод позволяет создать новый интерфейс операций **Согнуть / Разогнуть**.
2. Допустимыми значениями Type являются o3d_sheetMetalBendStraighten (Согнуть), o3d_sheetMetalBendBended (Разогнуть)
3. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalBendUnfoldParameters

Интерфейс параметров развертки листового тела.

Иерархия:

IDispatch

ISheetMetalBendUnfoldParameters

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalBendUnfoldParameters.

КОМПАС версия v18

ISheetMetalBendUnfoldParameters – свойства

ExcludedBendObjects – Исключенные сгибы. Массив объектов в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
ExcludedBendObjects = Object.ExcludedBendObjects;   Получить свойство (* )
Object.ExcludedBendObjects = ExcludedBendObjects;   Установить свойство (* )
ExcludedBendObjects = Object.GetExcludedBendObjects();   Получить свойство (**)
```

Object.SetExcludedBendObjects(ExcludedBendObjects);	Установить свойство (**)

Синтаксис COM:

Object.get_ExcludedBendObjects(&ExcludedBendObjects);	Получить свойство
Object.put_ExcludedBendObjects(ExcludedBendObjects);	Установить свойство

FixedFaces – Неподвижные грани

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

FixedFaces Object.FixedFaces;	=	Получить свойство (*)
Object.FixedFaces FixedFaces;	=	Установить свойство (*)
FixedFaces Object.GetFixedFaces();	=	Получить свойство (**)
Object.SetFixedFaces(FixedFaces);		Установить свойство (**)

Синтаксис COM:

Object.get_FixedFaces(&FixedFaces);	Получить свойство
Object.put_FixedFaces(FixedFaces);	Установить свойство

IsCreated – Заданы ли параметры развертки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCreated = Object.IsCreated	Получить свойство (*)
IsCreated = Object.GetIsCreated()	Получить свойство (**)

Синтаксис COM:

Object.get_IsCreated(&IsCreated)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Unfold – Развернуть

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Unfold = Object.Unfold;          Получить свойство (* )
Object.Unfold = Unfold;          Установить свойство (* )
Unfold = Object.GetUnfold();     Получить свойство (**)
Object.SetUnfold( Unfold );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Unfold( &Unfold );    Получить свойство
Object.put_Unfold(  Unfold );    Установить свойство
```

UnfoldPlane – Плоскость развертки

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
UnfoldPlane                 =  Получить свойство (* )
Object.UnfoldPlane;
Object.UnfoldPlane          =  Установить свойство (* )
UnfoldPlane;
UnfoldPlane                 =  Получить свойство (**)
Object.GetUnfoldPlane());
Object.SetUnfoldPlane(         Установить свойство (**)
UnfoldPlane );
```

Синтаксис COM:

```
Object.get_UnfoldPlane(                Получить
&UnfoldPlane );                        свойство
Object.put_UnfoldPlane(                Установить
UnfoldPlane );                         свойство
```

ISheetMetalBendUnfoldParameters – методы

DeleteParam – Удалить параметры развертки

Интерфейс...

Синтаксис Automation:

BOOL DeleteParam();

Синтаксис COM:

HRESULT DeleteParam(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

UpdateParam – Установить измененные параметры развертки

Интерфейс...

Синтаксис Automation:

BOOL UpdateParam();

Синтаксис COM:

HRESULT UpdateParam(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс ISheetMetalClosedCorners

Интерфейс коллекции элементов Замыкание углов.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ISheetMetalClosedCorners

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalClosedCorners.

КОМПАС версия v18

ISheetMetalClosedCorners – свойства

SheetMetalClosedCorner – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalClosedCorner.

Синтаксис Automation:

SheetMetalClosedCorner Object.SheetMetalClosedCorner(Index)	=	Получить свойство (*)
SheetMetalClosedCorner Object.GetSheetMetalClosedCorner(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalClosedCorner(Index, &SheetMetalClosedCorner)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalClosedCorners – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalClosedCorner ** Result);

Возвращаемое значение:

- указатель на интерфейс ISheetMetalClosedCorner.

Примечания:

1. Метод позволяет создать новый интерфейс операции **Замыкание углов**.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalClosedCorner

Интерфейс операции Замыкание углов.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalClosedCorner

КОМПАС версия v18

ISheetMetalClosedCorner – свойства

ClosingClosedType – Способ замыкания углов

Интерфейс...

Тип данных: Значение из перечисления ksClosingClosedTypeEnum.

Синтаксис Automation:

ClosingClosedType = Object.ClosingClosedType(Index);	=	Получить свойство (*)
Object.ClosingClosedType(Index) = ClosingClosedType;		Установить свойство (*)
ClosingClosedType = Object.GetClosingClosedType(Index);	=	Получить свойство (**)
Object.SetClosingClosedType(Index, ClosingClosedType);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingClosedType (Index, &ClosingClosedType);	Получить свойство
Object.put_ClosingClosedType (Index, ClosingClosedType);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingContinue – Продолжать замыкание прилегающих парных сгибов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosingContinue = Object.ClosingContinue(Index);	Получить свойство (*)
Object.ClosingContinue(Index) = ClosingContinue;	Установить свойство (*)
ClosingContinue = Object.GetClosingContinue(Index);	Получить свойство (**)
Object.SetClosingContinue(Index, ClosingContinue);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingContinue(Index, &ClosingContinue);	Получить свойство
Object.put_ClosingContinue(Index, ClosingContinue);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingCorneringType – Обработка угла при замыкании

Интерфейс...

Тип данных: Значение из перечисления ksClosingCorneringEnum.

Синтаксис Automation:

ClosingCorneringType = Object.ClosingCorneringType(Index);	Получить свойство (*)
Object.ClosingCorneringType(Index) = ClosingCorneringType;	Установить свойство (*)
ClosingCorneringType = Object.GetClosingCorneringType(Index);	Получить свойство (**)

Object.SetClosingCorneringType(Index, ClosingCorneringType);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingCorneringType(Index, &ClosingCorneringType);	Получить свойство
Object.put_ClosingCorneringType(Index, ClosingCorneringType);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingDirection – Направление. Переставить сторону

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosingDirection = Object.ClosingDirection(Index);	Получить свойство (*)
Object.ClosingDirection(Index) = ClosingDirection;	Установить свойство (*)
ClosingDirection = Object.GetClosingDirection(Index);	Получить свойство (**)
Object.SetClosingDirection(Index, ClosingDirection);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingDirection(Index, &ClosingDirection);	Получить свойство
Object.put_ClosingDirection(Index, ClosingDirection);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingGapValue – Значение зазора при замыкании углов

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingGapValue = Object.ClosingGapValue(Index);	Получить свойство (*)
Object.ClosingGapValue(Index) = ClosingGapValue;	Установить свойство (*)
ClosingGapValue = Object.GetClosingGapValue(Index);	Получить свойство (**)
Object.SetClosingGapValue(Index, ClosingGapValue);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingGapValue(Index, &ClosingGapValue);	Получить свойство
Object.put_ClosingGapValue(Index, ClosingGapValue);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingHoleDiameter – Диаметр отверстия при замыкании угла

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingHoleDiameter = Object.ClosingHoleDiameter(Index);	Получить свойство (*)
Object.ClosingHoleDiameter(Index) = ClosingHoleDiameter;	Установить свойство (*)
ClosingHoleDiameter = Object.GetClosingHoleDiameter(Index);	Получить свойство (**)
Object.SetClosingHoleDiameter(Index, ClosingHoleDiameter);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHoleDiameter(Index, &ClosingHoleDiameter);	Получить свойство
Object.put_ClosingHoleDiameter(Index, ClosingHoleDiameter);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingHoleOffset – Смещение отверстия при замыкании угла

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingHoleOffset = Object.ClosingHoleOffset(Index);	=	Получить свойство (*)
Object.ClosingHoleOffset(Index) = ClosingHoleOffset;		Установить свойство (*)
ClosingHoleOffset = Object.GetClosingHoleOffset(Index);	=	Получить свойство (**)
Object.SetClosingHoleOffset(Index, ClosingHoleOffset);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHoleOffset(Index, &ClosingHoleOffset);	Получить свойство
Object.put_ClosingHoleOffset(Index, ClosingHoleOffset);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

ClosingHolePlacement – Размещение отверстия при круговой обработке угла

Интерфейс...

Тип данных: Значение из перечисления ksClosingHolePlacementEnum.

Синтаксис Automation:

ClosingHolePlacement = Object.ClosingHolePlacement(Index);	Получить свойство (*)
Object.ClosingHolePlacement(Index) = ClosingHolePlacement;	Установить свойство (*)
ClosingHolePlacement = Object.GetClosingHolePlacement(Index);	Получить свойство (**)
Object.SetClosingHolePlacement(Index, ClosingHolePlacement);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHolePlacement(Index, &ClosingHolePlacement);	Получить свойство
Object.put_ClosingHolePlacement(Index, ClosingHolePlacement);	Установить свойство

Входные параметры:

long Index	- индекс сгиба.

CornersCount – Количество углов

Интерфейс...

Тип данных: long

Синтаксис Automation:

CornersCount = Object.CornersCount	Получить свойство (*)
CornersCount = Object.GetCornersCount()	Получить свойство (**)

Синтаксис COM:

Object.get_CornersCount(&CornersCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CornerObject – Угол. Объект (ребро или грань), задающий угол листового тела

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

CornerObject = Object.CornerObject(Index)		Получить свойство (*)
CornerObject Object.GetCornerObject(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_CornerObject(&CornerObject)	Index,	Получить свойство

Входные параметры:

long Index	- индекс сгиба.

Примечание:

Свойство доступно только для чтения.

CornersObjects – Углы. Массив объектов в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

CornersObjects Object.CornersObjects;	=	Получить свойство (*)
Object.CornersObjects CornersObjects;	=	Установить свойство (*)
CornersObjects Object.GetCornersObjects();	=	Получить свойство (**)
Object.SetCornersObjects(CornersObjects);		Установить свойство (**)

Синтаксис COM:

Object.get_CornersObjects(&CornersObjects);	Получить свойство
Object.put_CornersObjects(CornersObjects);	Установить свойство

DefailtParametersIndex – Единые параметры. Индекс угла, задающий общие параметры

Интерфейс...

Тип данных: long

Синтаксис Automation:

DefailtParametersIndex = Object.DefailtParametersIndex ;	Получить свойство (*)
Object.DefailtParametersIndex = DefailtParametersIndex ;	Установить свойство (*)
DefailtParametersIndex = Object.GetDefailtParametersIndex();	Получить свойство (**)
Object.SetDefailtParametersIndex(DefailtParametersIndex);	Установить свойство (**)

Синтаксис COM:

Object.get_DefailtParametersIndex(&DefailtParametersIndex);	Получить свойство
Object.put_DefailtParametersIndex(DefailtParametersIndex);	Установить свойство

ISheetMetalClosedCorner – методы

AddCornerObject – Добавить угол сгиба

Интерфейс...

Синтаксис Automation:

BOOL AddCornerObject(IModelObject * NewVal);

Синтаксис COM:

HRESULT AddCornerObject(IModelObject * NewVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

NewVal	- указатель на объект, задающий угол сгиба.

ClearCornerObjects – Удалить все углы

Интерфейс...

Синтаксис Automation:

BOOL ClearCornerObjects();

Синтаксис COM:

HRESULT ClearCornerObjects(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

DeleteCornerObject – Удалить угол сгиба

Интерфейс...

Синтаксис Automation:

BOOL DeleteCornerObject(long Index);

Синтаксис COM:

HRESULT DeleteCornerObject(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Index	- индекс сгиба.

Интерфейс ISheetMetalCut

Справка системы КОМПАС...

kompas.chm::/CM_SHMT_CUT.htm

Интерфейс параметров элемента листового тела "Вырез".

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalCut

Примечание:

Интерфейс можно получить с помощью метода коллекции элементов "Вырез" ISheetMetalCuts::Add или свойства ISheetMetalCuts::SheetMetalCut.

ISheetMetalCut – свойства

Body – Тело для операции

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Body = iObject.Body	Получить свойство (*)
iObject.Body = Body	Установить свойство (*)
Body = iObject.GetBody()	Получить свойство (**)
iObject.SetBody(Body)	Установить свойство (**)

Синтаксис COM:

iObject->get_Body(&Body)	Получить свойство
iObject->put_Body(Body)	Установить свойство

Примечание.

В качестве тела можно передавать листовое тело ISheetMetalBody, или примитив, принадлежащий листовому телу.

Cut – Результат операции

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Cut = iObject.Cut;	Получить свойство (*)
iObject.Cut = Cut;	Установить свойство (*)
Cut = iObject.GetCut();	Получить свойство (**)

```
iObject.SetCut( Cut );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Cut( &Cut );       Получить свойство
iObject->put_Cut( Cut );        Установить свойство
```

Значения свойства:

```
TRUE                            - вычитание элемента,
FALSE                           - пересечение элементов.
```

Примечание.

Позволяет считывать и устанавливать результат операции.

CutType – Тип построения выреза

Интерфейс...

Тип данных: из перечисления ksHoleCutTypeEnum

Синтаксис Automation:

```
CutType = iObject.CutType;      Получить свойство (* )
iObject.CutType = CutType;      Установить свойство (* )
CutType =                       Получить свойство (**)
iObject.GetCutType();
iObject.SetCutType(             Установить свойство (**)
CutType );
```

Синтаксис COM:

```
iObject->get_CutType( &CutType );       Получить свойство
iObject->put_CutType( CutType );        Установить свойство
```

Примечание.

Позволяет считывать и устанавливать тип построения отверстия.

Depth – Глубина выреза

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Depth = iObject.Depth;          Получить свойство (* )
iObject.Depth = Depth;          Установить свойство (* )
Depth =                         Получить свойство (**)
iObject.GetDepth();
```

```
iObject.SetDepth( Depth );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Depth( &Depth );          Получить свойство
iObject->put_Depth( Depth );           Установить свойство
```

Примечание.

1. Свойство доступно при типе построения отверстия "на глубину", см.ksHoleCutTypeEnum.
2. Позволяет считывать и устанавливать глубину отверстия.

DepthObject – Объект, задающий глубину вырезания

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
DepthObject =                  Получить свойство (* )
iObject.DepthObject;
iObject.DepthObject =          Установить свойство (* )
DepthObject;
DepthObject =                  Получить свойство (**)
iObject.GetDepthObject();
iObject.SetDepthObject(        Установить свойство (**)
DepthObject );
```

Синтаксис COM:

```
iObject->get_DepthObject(      Получить свойство
&DepthObject );
iObject->put_DepthObject(      Установить свойство
DepthObject );
```

Примечание.

1. Свойство доступно при типе построения отверстия "до грани", см.ksHoleCutTypeEnum.
2. Позволяет получать указатель на интерфейс объекта и устанавливать грань, задающую глубину выдавливания.

Sketch – Эскиз выреза

Интерфейс...

Тип данных: указатель на интерфейс ISketch

Синтаксис Automation:

```
Sketch = iObject.Sketch;          Получить свойство (* )
iObject.Sketch = Sketch;          Установить свойство (* )
```

Sketch = iObject.GetSketch();	Получить свойство (**)
iObject.SetSketch(Sketch);	Установить свойство (**)

Синтаксис COM:

iObject->get_Sketch(&Sketch);	Получить свойство
iObject->put_Sketch(Sketch);	Установить свойство

Примечание.

Позволяет получать указатель на интерфейс эскиза выреза и устанавливать новый эскиз.

Интерфейс ISheetMetalHole

Справка системы КОМПАС...

`kompas.chm::/CM_SHMT_HOLE.htm`

Интерфейс параметров элемента листового тела "Отверстие".

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalHole

Примечание:

Интерфейс можно получить с помощью метода коллекции элементов "отверстие" ISheetMetalHoles::Add или свойства ISheetMetalHoles::SheetMetalHole.

ISheetMetalHole – свойства

AssociationVertex – Точечный объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationVertex = Object.AssociationVertex;	Получить свойство (*)
Object.AssociationVertex = AssociationVertex;	Установить свойство (*)
AssociationVertex = Object.GetAssociationVertex();	Получить свойство (**)
Object.SetAssociationVertex(AssociationVertex);	Установить свойство (**)

Синтаксис COM:

```
Object.get_AssociationVer    Получить свойство
tex( &AssociationVertex );
Object.put_AssociationVer    Установить свойство
tex(  AssociationVertex );
```

Axis – Создавать ось

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
 Axis = Object. Axis;          Получить свойство (* )
Object. Axis =  Axis;          Установить свойство (* )
 Axis = Object.Get Axis();     Получить свойство (**)
Object.Set Axis(  Axis );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ Axis( & Axis );    Получить свойство
Object.put_ Axis(   Axis );    Установить свойство
```

BasePlane – Опорная грань

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
BasePlane =                    Получить свойство (* )
iObject.BasePlane;
iObject.BasePlane =            Установить свойство (* )
BasePlane;
BasePlane =                    Получить свойство (**)
iObject.GetBasePlane();
iObject.SetBasePlane(          Установить свойство (**)
BasePlane );
```

Синтаксис COM:

```
iObject->get_BasePlane(      Получить свойство
&BasePlane );
iObject->put_BasePlane(      Установить свойство
BasePlane );
```

Примечание.

Позволяет устанавливать и получать указатель на интерфейс грани, на которой будет располагаться эскиз отверстия.

Body – Тело для операции

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
Body = Object.Body;          Получить свойство (* )
Object.Body = Body;          Установить свойство (* )
Body = Object.GetBody();     Получить свойство (**)
Object.SetBody( Body );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Body( &Body );    Получить свойство
Object.put_Body(  Body );    Установить свойство
```

CutType – Тип построения отверстия

Интерфейс...

Тип данных: из перечисления ksHoleCutTypeEnum

Синтаксис Automation:

```
CutType = iObject.CutType;   Получить свойство (* )
iObject.CutType = CutType;   Установить свойство (* )
CutType =                    Получить свойство (**)
iObject.GetCutType();
iObject.SetCutType(          Установить свойство (**)
CutType );
```

Синтаксис COM:

```
iObject->get_CutType(        Получить свойство
&CutType );
iObject->put_CutType(        Установить свойство
CutType );
```

Примечание.

Позволяет считывать и устанавливать тип построения отверстия.

Depth – Глубина отверстия

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Depth = iObject.Depth;          Получить свойство (* )
iObject.Depth = Depth;          Установить свойство (* )
Depth =                         Получить свойство (**)
iObject.GetDepth();
iObject.SetDepth( Depth );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Depth(             Получить свойство
&Depth );
iObject->put_Depth( Depth       Установить свойство
);
```

Примечание.

1. Свойство доступно при типе построения отверстия "на глубину", см.ksHoleCutTypeEnum.
2. Позволяет считывать и устанавливать глубину отверстия.

DepthObject – Объект, задающий глубину вырезания

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
DepthObject =                   Получить свойство (* )
iObject.DepthObject;
iObject.DepthObject =           Установить свойство (* )
DepthObject;
DepthObject =                   Получить свойство (**)
iObject.GetDepthObject();
iObject.SetDepthObject(         Установить свойство (**)
DepthObject );
```

Синтаксис COM:

```
iObject->get_DepthObject(       Получить свойство
&DepthObject );
```

```
iObject->put_DepthObject(     Установить свойство
DepthObject );
```

Примечание.

1. Свойство доступно при типе построения отверстия "до грани", см.ksHoleCutTypeEnum.
2. Позволяет получать указатель на интерфейс объекта и устанавливать объект, задающий глубину выдавливания.

Diameter – Диаметр отверстия

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Diameter =                    Получить свойство (* )
iObject.Diameter;
iObject.Diameter =            Установить свойство (* )
Diameter;
Diameter =                    Получить свойство (**)
iObject.GetDiameter();
iObject.SetDiameter(          Установить свойство (**)
Diameter );
```

Синтаксис COM:

```
iObject->get_Diameter(        Получить свойство
&Diameter );
iObject->put_Diameter(        Установить свойство
Diameter );
```

Примечание.

Позволяет считывать и устанавливать диаметр отверстия.

Point3DParamSurface – Параметры пространственной точки на поверхности

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
Point3DParamSurface =                    Получить свойство(* )
Object.Point3DParamSurface
Point3DParamSurface =                    Получить свойство(**)
Object.GetPoint3DParamSurface()
```

Синтаксис COM:

Object.get_Point3DParamSurface(&Point3DParamSurface)	Получить свойство.

Примечание:

Свойство доступно только для чтения.

Свойство позволяет получить/установить параметры точки привязки отверстия на поверхности, используя интерфейс параметров точки на поверхности IPoint3DParamSurface.

Sketch – Эскиз

Интерфейс...

Тип данных: указатель на интерфейс эскиза ISketch

Синтаксис Automation:

Sketch = iObject.Sketch;	Получить свойство (*)
Sketch = iObject.GetSketch();	Получить свойство (**)

Синтаксис COM:

iObject->get_Sketch(&Sketch)	Получить свойство

Примечание.

1. Свойство позволяет получать эскиз для отверстия.
2. Свойство доступно только для чтения.
3. Эскиз создается при создании.

X – Координата центра отверстия по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = iObject.X;	Получить свойство (*)
iObject.X = X;	Установить свойство (*)
X = iObject.GetX();	Получить свойство (**)
iObject.SetX(X);	Установить свойство (**)

Синтаксис COM:

```
iObject->get_X( &X );        Получить свойство
iObject->put_X( X );         Установить свойство
```

Примечание.

Позволяет считывать и устанавливать координату центра отверстия по X.

Y – Координата центра отверстия по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y = iObject.Y;               Получить свойство (* )
iObject.Y = Y;               Установить свойство (* )
Y = iObject.GetY();          Получить свойство (**)
iObject.SetY( Y );           Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y( &Y );        Получить свойство
iObject->put_Y( Y );         Установить свойство
```

Примечание.

Позволяет считывать и устанавливать координату центра отверстия по Y.

Интерфейс ISheetMetalJalousies

Интерфейс коллекции элементов Жалюзи.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ISheetMetalJalousies

КОМПАС версия v18

ISheetMetalJalousies – свойства

SheetMetalJalousie – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalJalousie.

Синтаксис Automation:

SheetMetalJalousie = Object.SheetMetalJalousie(Index);	Получить свойство (*)
SheetMetalJalousie = Object.GetSheetMetalJalousie(Index);	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalJalousie(Index, &SheetMetalJalousie);	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalJalousies – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalJalousie * * Result);

Возвращаемое значение:

	- указатель на интерфейс ISheetMetalJalousie.

Примечания:

1. Метод позволяет создать новый интерфейс операции Жалюзи.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalJalousie

Интерфейс операции Жалюзи.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalJalousie

Данный интерфейс можно получить с помощью метода коллекции операций пластина ISheetMetalJalousies::Add или свойства ISheetMetalJalousies::SheetMetalJalousie.

КОМПАС версия v18

ISheetMetalJalousie – свойства

BuildingType – Способ построения

Интерфейс...

Тип данных: Значение из перечисления ksJalousieBuildingTypeEnum.

Синтаксис Automation:

```
BuildingType = Object.BuildingType;          Получить свойство (* )
Object.BuildingType = BuildingType;          Установить свойство (* )
BuildingType = Object.GetBuildingType();     Получить свойство (**)
Object.SetBuildingType( BuildingType );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType( &BuildingType );    Получить свойство
Object.put_BuildingType(  BuildingType );    Установить свойство
```

Direction – Направление построения: TRUE – Прямое, FALSE – Обратное

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction                 Получить свойство (* )
Object.Direction = Direction;                Установить свойство (* )
Direction = Object.GetDirection();           Получить свойство (**)
Object.SetDirection( Direction );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction );          Получить свойство
Object.put_Direction(  Direction );          Установить свойство
```

FormEnd – Форма торца

Интерфейс...

Тип данных: Значение из перечисления ksJalousieFormEndEnum.

Синтаксис Automation:

```
FormEnd = Object.FormEnd;                Получить свойство (* )
FormEnd = Object.GetFormEnd();           Установить свойство (* )
FormEnd = Object.GetFormEnd();           Получить свойство (**)
Object.SetFormEnd( FormEnd );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_FormEnd( &FormEnd );          Получить свойство
Object.put_FormEnd(  FormEnd );          Установить свойство
```

Height – Высота

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Height = Object.Height;                  Получить свойство (* )
Object.Height = Height;                  Установить свойство (* )
Height = Object.GetHeight();             Получить свойство (**)
Object.SetHeight( Height );              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Height( &Height );            Получить свойство
Object.put_Height(  Height );            Установить свойство
```

HeightType – Способ задания высоты

Интерфейс...

Тип данных: Значение из перечисления ksJalousieHeightTypeEnum.

Синтаксис Automation:

```
HeightType = Object.HeightType;          Получить свойство (* )
Object.HeightType = HeightType;          Установить свойство (* )
HeightType = Object.GetHeightType();     Получить свойство (**)
Object.SetHeightType( HeightType );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HeightType( &HeightType );      Получить свойство
Object.put_HeightType(  HeightType );      Установить свойство
```

Radius – Радиус скругления основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius;                    Получить свойство (* )
Object.Radius = Radius;                    Установить свойство (* )
Radius = Object.GetRadius();               Получить свойство (**)
Object.SetRadius( Radius );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius );              Получить свойство
Object.put_Radius(  Radius );              Установить свойство
```

Round – Скругление основания

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Round = Object.Round;                      Получить свойство (* )
Object.Round = Round;                      Установить свойство (* )
Round = Object.GetRound();                 Получить свойство (**)
Object.SetRound( Round );                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Round( &Round );                Получить свойство
Object.put_Round(  Round );                Установить свойство
```

Side – Положение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Side = Object.Side;                        Получить свойство (* )
```

```
Object.Side = Side;                    Установить свойство (* )
Side = Object.GetSide();               Получить свойство (**)
Object.SetSide( Side );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Side( &Side );              Получить свойство
Object.put_Side(  Side );              Установить свойство
```

Значение свойства:

```
TRUE                                   - справа
FALSE                                  - слева.
```

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Sketch = Object.Sketch;                Получить свойство (* )
Object.Sketch = Sketch;                Установить свойство (* )
Sketch = Object.GetSketch();           Получить свойство (**)
Object.SetSketch( Sketch );            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch );          Получить свойство
Object.put_Sketch(  Sketch );          Установить свойство
```

Width – Ширина

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width;                  Получить свойство (* )
Object.Width = Width;                  Установить свойство (* )
Width = Object.GetWidth();             Получить свойство (**)
Object.SetWidth( Width );              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width );          Получить свойство
Object.put_Width(  Width );          Установить свойство
```

Интерфейс ISheetMetalLineBend

Справка системы КОМПАС...

kompas.chm::/808_94_5_Sgib_po_linii.htm

Интерфейс параметров Сгиба по линии.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalLineBend

Примечание:

Интерфейс можно получить с помощью метода коллекции сгибов ISheetMetalLineBends::Add или свойства ISheetMetalLineBends::SheetMetalLineBend.

ISheetMetalLineBend – свойства

Angle – Угол сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = iObject.Angle;          Получить свойство (* )
iObject.Angle = Angle;          Установить свойство (* )
Angle = iObject.GetAngle;       Получить свойство (**)
iObject.SetAngle( Angle );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Angle( &Angle );          Получить свойство
iObject->put_Angle( Angle );           Установить свойство
```

Примечание.

Позволяет считывать и устанавливать угол сгиба.

AngleType – Способ задания угла – угол сгиба / дополняющий угол

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AngleType = iObject.AngleType;	Получить свойство (*)
iObject.AngleType = AngleType;	Установить свойство (*)
AngleType = iObject.GetAngleType;	Получить свойство (**)
iObject.SetAngleType(AngleType);	Установить свойство (**)

Синтаксис COM:

iObject->get_AngleType(&AngleType);	Получить свойство
iObject->put_AngleType(AngleType);	Установить свойство

Значения свойства:

TRUE	- угол интерпретируется как угол сгиба,
FALSE	- угол интерпретируется как дополняющий угол.

Примечание.

Позволяет считывать и устанавливать способ задания угла.

BendCoefficient – Коэффициент нейтрального слоя

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendCoefficient = iObject.BendCoefficient;	Получить свойство (*)
iObject.BendCoefficient = BendCoefficient;	Установить свойство (*)
BendCoefficient = iObject.GetBendCoefficient;	Получить свойство (**)
iObject.SetBendCoefficient(BendCoefficient);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendCoefficient(&BendCoefficient);	Получить свойство
iObject->put_BendCoefficient(BendCoefficient);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - коэффициент нейтрального слоя. Позволяет считывать и устанавливать коэффициент нейтрального слоя. Get - свойство будет возвращать коэффициент нейтрального слоя базового листового тела, пока не будет установлен коэффициент нейтрального слоя для сгиба по линии.

BendLeftSideFixed – Признак фиксации левой стороны

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BendLeftSideFixed = iObject.BendLeftSideFixed;	Получить свойство (*)
iObject.BendLeftSideFixed = BendLeftSideFixed;	Установить свойство (*)
BendLeftSideFixed = iObject.GetBendLeftSideFixed;	Получить свойство (**)
iObject.SetBendLeftSideFixed(BendLeftSideFixed);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendLeftSideFixed(&BendLeftSideFixed);	Получить свойство
iObject->put_BendLeftSideFixed(BendLeftSideFixed);	Установить свойство

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Примечание.

Позволяет установить и получить признак фиксации левой стороны.

BendReduction – Уменьшение сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendReduction = iObject.BendReduction;	Получить свойство (*)
iObject.BendReduction = BendReduction;	Установить свойство (*)
BendReduction = iObject.GetBendReduction;	Получить свойство (**)
iObject.SetBendReduction(BendReduction);	Установить свойство (**)

Синтаксис COM:

iObject->get_BendReduction(&BendReduction);	Получить свойство
iObject->put_BendValue(BendValue);	Установить свойство

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - величина сгиба. Позволяет считывать и устанавливать величину сгиба. Get - свойство будет возвращать величину сгиба базового листового тела, пока не будет установлена "величина сгиба" для сгиба по линии.

BendTablePath – Уменьшение сгиба

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

BendTablePath = iObject.BendTablePath;	Получить свойство (*)
BendTablePath = iObject.GetBendTablePath;	Получить свойство (**)

Синтаксис COM:

iObject->get_BendTablePath(&BendTablePath);	Получить свойство

Примечание.

Свойство доступно только для чтения. Таблица сгибов одна на документ. Путь к таблице сгибов выбирается у базового листового тела.

BendType – Способ формирования сгиба

Интерфейс...

Тип данных: из перечисления ksBendTypeEnum

Синтаксис Automation:

```
BendType = iObject.BendType;          Получить свойство (* )
iObject.BendType = BendType;          Установить свойство (* )
BendType = iObject.GetBendType;       Получить свойство (**)
iObject.SetBendType( BendType );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_BendType( &BendType );   Получить свойство
iObject->put_BendType( BendType );    Установить свойство
```

Примечание.

Позволяет считывать и устанавливать способ формирования сгиба.

BendValue – Величина сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BendValue = iObject.BendValue;          Получить свойство (* )
iObject.BendValue = BendValue;          Установить свойство (* )
BendValue = iObject.GetBendValue;       Получить свойство (**)
iObject.SetBendValue( BendValue );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_BendValue( &BendValue );   Получить свойство
iObject->put_BendValue( BendValue );    Установить свойство
```

Примечание.

Чтобы свойство было доступно, необходимо установить тип расчета длины развёртки - величина сгиба. Позволяет считывать и устанавливать величину сгиба. Get - свойство будет возвращать величину сгиба базового листового тела, пока не будет установлена "величина сгиба" для сгиба по линии.

Direction – Направление построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = iObject.Direction;                  Получить свойство (* )
iObject.Direction = Direction;                  Установить свойство (* )
Direction = iObject.GetDirection;               Получить свойство (**)
iObject.SetDirection( Direction );              Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Direction( &Direction );     Получить свойство
iObject->put_Direction( Direction );      Установить свойство
```

Значения свойства:

```
TRUE                    - прямое направление,
FALSE                   - обратное направление.
```

Примечание.

Позволяет установить и получить направление построения сгиба.

DismissalAngleType – Способ освобождения угла сгиба

Интерфейс...

Тип данных: из перечисления ksBendAngleReleaseTypeEnum

Синтаксис Automation:

```
DismissalAngleType = iObject.DismissalAngleType;    Получить свойство (* )
iObject.DismissalAngleType = DismissalAngleType;    Установить свойство (* )
DismissalAngleType =                                Получить свойство (**)
iObject.GetDismissalAngleType;
iObject.SetDismissalAngleType(                      Установить свойство (**)
DismissalAngleType );
```

Синтаксис COM:

```
iObject->get_DismissalAngleType(        Получить свойство
&DismissalAngleType );
iObject->put_DismissalAngleType(        Установить свойство
DismissalAngleType );
```

Примечание.

Позволяет считывать и устанавливать способ освобождения угла сгиба.

Faces – Массив граней (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Faces = iObject.Faces;          Получить свойство (* )
iObject.Faces = Faces;          Установить свойство (* )
Faces = iObject.GetFaces;       Получить свойство (**)
iObject.SetFaces( Faces );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Faces( &Faces );           Получить свойство
iObject->put_Faces( Faces );            Установить свойство
```

Примечание.

Позволяет получать и устанавливать грань или массив граней, участвующих в операции Сгиб по линии. Массив граней возвращается в виде массива SAFEARRAY граней LPDISPATCH (VT_ARRAY | VT_DISPATCH).

InternalRadius – Внутренний радиус

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
InternalRadius =                Получить свойство (* )
iObject.InternalRadius;
iObject.InternalRadius =        Установить свойство (* )
InternalRadius;
InternalRadius =                Получить свойство (**)
iObject.GetInternalRadius;
iObject.SetInternalRadius(      Установить свойство (**)
InternalRadius );
```

Синтаксис COM:

```
iObject->get_InternalRadius( &InternalRadius );    Получить свойство
iObject->put_InternalRadius( InternalRadius );     Установить свойство
```

Значения свойства:

```
TRUE     - внутренний радиус,
FALSE    - наружный радиус.
```

Примечание.

Позволяет получать и устанавливать признак "внутренний радиус".

Line – Прямолинейный объект (отрезок эскиза или ломаной, ребро или ось)

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
Line = iObject.Line;          Получить свойство (* )
iObject.Line = Line;          Установить свойство (* )
Line = iObject.GetLine;       Получить свойство (**)
iObject.SetLine( Line );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Line( &Line );   Получить свойство
iObject->put_Line( Line );    Установить свойство
```

Примечание.

Позволяет получать и устанавливать прямолинейный объект (отрезок эскиза или ломанной, ребро или ось), относительно которого сгибается листовое тело. Прямолинейный объект и грань (массив граней), участвующие в операции "сгиб по линии", должны располагаться в одной плоскости.

Radius – Радиус сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = iObject.Radius;                    Получить свойство (* )
iObject.Radius = Radius;                    Установить свойство (* )
Radius = iObject.GetRadius;                 Получить свойство (**)
iObject.SetRadius( Radius );                Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Radius(          Получить свойство
&Radius );
iObject->put_Radius(          Установить свойство
Radius );
```

Примечание.

Позволяет получать и устанавливать радиус сгиба.

Straighten – Разогнуть сгиб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Straighten = iObject.Straighten;            Получить свойство (* )
iObject.Straighten = Straighten;            Установить свойство (* )
Straighten = iObject.GetStraighten;         Получить свойство (**)
iObject.SetStraighten( Straighten );        Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Straighten( &Straighten );    Получить свойство
iObject->put_Straighten( Straighten );     Установить свойство
```

Значения свойства:

```
TRUE                          - угол разогнут,
FALSE                         - угол согнут.
```

Примечание.

Позволяет считывать и устанавливать способ формирования сгиба.

UnfoldType – Способ определения длины развертки

Интерфейс...

Тип данных: из перечисления ksUnfoldTypeEnum

Синтаксис Automation:

```
UnfoldType = iObject.UnfoldType;          Получить свойство (* )
iObject.UnfoldType = UnfoldType;          Установить свойство (* )
UnfoldType = iObject.GetUnfoldType;       Получить свойство (**)
iObject.SetUnfoldType( UnfoldType );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_UnfoldType(                  Получить свойство
&UnfoldType );
iObject->put_UnfoldType(                  Установить свойство
UnfoldType );
```

Примечание.

Позволяет считывать и устанавливать способ определения длины развертки. Если задан способ определения длины развертки - из таблицы, то будут недоступны свойства установки (Set) для коэффициента нейтрального слоя, величины сгиба и уменьшения сгиба, т.к они будут браться из таблицы.

Интерфейс ISheetMetalRibs

Интерфейс коллекции элементов Ребро усиления.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalRibs

КОМПАС версия v18

ISheetMetalRibs – свойства

SheetMetalRib – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalRib.

Синтаксис Automation:

```
SheetMetalRib = Object.SheetMetalRib( Index );      Получить свойство (* )
SheetMetalRib = Object.GetSheetMetalRib( Index );   Получить свойство (**)
```

Синтаксис COM:

Object.get_SheetMetalRib(Index, &SheetMetalRib);	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalRibs – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalRib ** Result);

Возвращаемое значение:

	- указатель на интерфейс ISheetMetalRib.

Примечания:

1. Метод позволяет создать новый интерфейс операции **Пластина**.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalRuledShell

Интерфейс операции Обечайка.

Иерархия:

IDispatch

ISheetMetalRuledShell

Данный интерфейс можно получить из коллекции обечаек, полученной с помощью свойства ISheetMetalContainer::SheetMetalRuledShells, с помощью метода ISheetMetalBodies::Add или свойства ISheetMetalBodies::SheetMetalBody.

Интерфейс является дополнительным для ISheetMetalBody.

ISheetMetalRuledShell – свойства

DraftOutward – Признак уклона наружу

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DraftOutward = Object.DraftOutward(Normal)	Получить свойство (*)
Object.DraftOutward(Normal) = DraftOutward	Установить свойство (*)
DraftOutward = Object.GetDraftOutward(Normal)	Получить свойство (**)
Object.SetDraftOutward(Normal, DraftOutward)	Установить свойство (**)

Синтаксис COM:

Object.get_DraftOutward(Normal, &DraftOutward)	Получить свойство
Object.put_DraftOutward(Normal, DraftOutward)	Установить свойство

Входные параметры:

normal TRUE	- угол наклона в прямом направлении.
FALSE	- угол наклона в обратном направлении.

DraftValue – Угол уклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

DraftValue = Object.DraftValue(Normal)	Получить свойство (*)
Object.DraftValue(Normal) = DraftValue	Установить свойство (*)
DraftValue = Object.GetDraftValue(Normal)	Получить свойство (**)
Object.SetDraftValue(Normal, DraftValue)	Установить свойство (**)

Синтаксис COM:

Object.get_DraftValue(Normal, &DraftValue)	Получить свойство
Object.put_DraftValue(Normal, DraftValue)	Установить свойство

Входные параметры:

normal TRUE	- угол наклона в прямом направлении.

FALSE - угол наклона в обратном направлении.

GapDraftPosition – Угловое смещение зазора вдоль кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

GapDraftPosition = Object.GapDraftPosition	Получить свойство (*)
Object.GapDraftPosition = GapDraftPosition	Установить свойство (*)
GapDraftPosition = Object.GetGapDraftPosition()	Получить свойство (**)
Object.SetGapDraftPosition(GapDraftPosition)	Установить свойство (**)

Синтаксис COM:

Object.get_GapDraftPosition(&GapDraftPosition)	Получить свойство
Object.put_GapDraftPosition(GapDraftPosition)	Установить свойство

GapOffsetLength – Величина смещения зазора вдоль кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

GapOffsetLength = Object.GapOffsetLength	Получить свойство (*)
Object.GapOffsetLength = GapOffsetLength	Установить свойство (*)
GapOffsetLength = Object.GetGapOffsetLength()	Получить свойство (**)
Object.SetGapOffsetLength(GapOffsetLength)	Установить свойство (**)

Синтаксис COM:

Object.get_GapOffsetLength(&GapOffsetLength)	Получить свойство
Object.put_GapOffsetLength(GapOffsetLength)	Установить свойство

GapOffsetU – Смещение зазора в % от длины кривой

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
GapOffsetU = Object.GapOffsetU          Получить свойство (* )
Object.GapOffsetU = GapOffsetU          Установить свойство (* )
GapOffsetU = Object.GetGapOffsetU()     Получить свойство (**)
Object.SetGapOffsetU( GapOffsetU )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_GapOffsetU( &GapOffsetU )    Получить свойство
Object.put_GapOffsetU(  GapOffsetU )    Установить свойство
```

GapValue – Зазор

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
GapValue = Object.GapValue              Получить свойство (* )
Object.GapValue = GapValue              Установить свойство (* )
GapValue = Object.GetGapValue()         Получить свойство (**)
Object.SetGapValue( GapValue )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_GapValue( &GapValue )        Получить свойство
Object.put_GapValue(  GapValue )        Установить свойство
```

KeepRadius – Постоянный радиус

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
KeepRadius = Object.KeepRadius          Получить свойство (* )
Object.KeepRadius = KeepRadius          Установить свойство (* )
KeepRadius = Object.GetKeepRadius()     Получить свойство (**)
Object.SetKeepRadius( KeepRadius )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_KeepRadius( &KeepRadius )    Получить свойство
Object.put_KeepRadius(  KeepRadius )    Установить свойство
```

OffsetGapType – Тип смещения зазора

Интерфейс...

Тип данных: Значение из перечисления ksOffsetGapType.

Синтаксис Automation:

```
OffsetGapType = Object.OffsetGapType          Получить свойство (* )
Object.OffsetGapType = OffsetGapType          Установить свойство (* )
OffsetGapType = Object.GetOffsetGapType()     Получить свойство (**)
Object.SetOffsetGapType( OffsetGapType )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OffsetGapType( &OffsetGapType )    Получить свойство
Object.put_OffsetGapType(  OffsetGapType )    Установить свойство
```

RuledBorder – Тип кромки оснований

Интерфейс...

Тип данных: из перечисления ksRuledBorderEnum.

Синтаксис Automation:

```
RuledBorder = Object.RuledBorder              Получить свойство (* )
Object.RuledBorder = RuledBorder              Установить свойство (* )
RuledBorder = Object.GetRuledBorder()         Получить свойство (**)
Object.SetRuledBorder( RuledBorder )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_RuledBorder( &RuledBorder )        Получить свойство
Object.put_RuledBorder(  RuledBorder )        Установить свойство
```

RuledJoint – Тип кромки стыка

Интерфейс...

Тип данных: из перечисления ksRuledJointEnum.

Синтаксис Automation:

```
RuledJoint = Object.RuledJoint                Получить свойство (* )
Object.RuledJoint = RuledJoint                Установить свойство (* )
RuledJoint = Object.GetRuledJoint()           Получить свойство (**)
Object.SetRuledJoint( RuledJoint )            Установить свойство (**)
```

Синтаксис COM:

Object.get_RuledJoint(&RuledJoint)	Получить свойство
Object.put_RuledJoint(RuledJoint)	Установить свойство

SegmentationMethod – Способ сегментации эскиза

Интерфейс...

Тип данных: из перечисления ksSegmentationMethodEnum

Синтаксис Automation:

SegmentationMethod = Object.SegmentationMethod	=	Получить свойство (*)
Object.SegmentationMethod = SegmentationMethod	=	Установить свойство (*)
SegmentationMethod = Object.GetSegmentationMethod()	=	Получить свойство (**)
Object.SetSegmentationMethod(SegmentationMethod)		Установить свойство (**)

Синтаксис COM:

Object.get_SegmentationMethod(&SegmentationMethod)	Получить свойство
Object.put_SegmentationMethod(SegmentationMethod)	Установить свойство

SegmentationSplitValue – Величина, определяющая разбиения, интерпретация зависит от типа разбиения: это либо количество сегментов, либо длина сегмента

Интерфейс...

Тип данных: double

Синтаксис Automation:

SegmentationSplitValue = Object.SegmentationSplitValue	=	Получить свойство (*)
Object.SegmentationSplitValue = SegmentationSplitValue	=	Установить свойство (*)
SegmentationSplitValue = Object.GetSegmentationSplitValue()	=	Получить свойство (**)

Object.SetSegmentationSplitValue(SegmentationSplitValue)	Установить свойство (**)

Синтаксис COM:

Object.get_SegmentationSplitValue(&SegmentationSplitValue)	Получить свойство
Object.put_SegmentationSplitValue(SegmentationSplitValue)	Установить свойство

UseSegmentation – Сегментация

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseSegmentation = Object.UseSegmentation	=	Получить свойство (*)
Object.UseSegmentation = UseSegmentation	=	Установить свойство (*)
UseSegmentation = Object.GetUseSegmentation()	=	Получить свойство (**)
Object.SetUseSegmentation(UseSegmentation)		Установить свойство (**)

Синтаксис COM:

Object.get_UseSegmentation(&UseSegmentation)	Получить свойство
Object.put_UseSegmentation(UseSegmentation)	Установить свойство

Интерфейс ISheetMetalLinearRuledShell

Интерфейс операции Линейчатая обечайка.

Иерархия:

IDispatch

ISheetMetalLinearRuledShell

Данный интерфейс можно получить из коллекции обечаек, полученной с помощью свойства ISheetMetalContainer::SheetMetalLinearRuledShells, с помощью метода ISheetMetalBodies::Add или свойства ISheetMetalBodies::SheetMetalBody.

Интерфейс является дополнительным для ISheetMetalBody, ISheetMetalRuledShell.

Версия Компас v18.1

ISheetMetalLinearRuledShell– свойства

AutoSegmentation – Автоматическое разбиение

Интерфейс...

Тип данных: BOOL

Интерфейс...

Синтаксис Automation:

AutoSegmentation = Object.AutoSegmentation	=	Получить свойство (*)
Object.AutoSegmentation = AutoSegmentation	=	Установить свойство (*)
AutoSegmentation = Object.GetAutoSegmentation()	=	Получить свойство (**)
Object.SetAutoSegmentation(AutoSegmentation)		Установить свойство (**)

Синтаксис COM:

Object.get_AutoSegmentation(&AutoSegmentation)	Получить свойство
Object.put_AutoSegmentation(AutoSegmentation)	Установить свойство

CurvesCount – Количество кривых в сегментации

Интерфейс...

Тип данных: long

Синтаксис Automation:

CurvesCount = Object.CurvesCount		Получить свойство (*)
CurvesCount = Object.GetCurvesCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_CurvesCount(&CurvesCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CurveSegmentationMethod – Способ сегментации эскиза

Интерфейс...

Тип данных: из перечисления ksSegmentationMethodEnum

Синтаксис Automation:

CurveSegmentationMethod = Object.CurveSegmentationMethod(CurveIndex	=	Получить свойство (*)
Object.CurveSegmentationMethod(CurveIndex) = CurveSegmentationMethod		Установить свойство (*)
CurveSegmentationMethod = Object.GetCurveSegmentationMethod (CurveIndex)	=	Получить свойство (**)
Object.SetCurveSegmentationMethod (CurveIndex, CurveSegmentationMethod)		Установить свойство (**)

Синтаксис COM:

Object.get_CurveSegmentationMethod(CurveIndex, &CurveSegmentationMethod)	Получить свойство
Object.put_CurveSegmentationMethod(CurveIndex, CurveSegmentationMethod)	Установить свойство

Входные параметры:

long CurveIndex	- индекс кривой.

CurveSegmentationSplitValue – Величина, определяющая разбиения, интерпретация зависит от типа разбиения: количество сегментов либо длина сегмента

Интерфейс...

Тип данных: double

Синтаксис Automation:

CurveSegmentationSplitValue = Object.CurveSegmentationSplitValue(CurveIndex)	Получить свойство (*)
Object.CurveSegmentationSplitValue(CurveIndex) = CurveSegmentationSplitValue	Установить свойство (*)
CurveSegmentationSplitValue = Object.GetCurveSegmentationSplitValue(CurveIndex)	Получить свойство (**)
Object.SetCurveSegmentationSplitValue(CurveIndex, CurveSegmentationSplitValue)	Установить свойство (**)

Синтаксис COM:

Object.get_CurveSegmentationSplitValue(CurveIndex, &CurveSegmentationSplitValue)	Получить свойство
Object.put_CurveSegmentationSplitValue(CurveIndex, CurveSegmentationSplitValue)	Установить свойство

Входные параметры:

long CurveIndex	- индекс кривой.

CurveUseSegmentation – Сегментация

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CurveUseSegmentation = Object.CurveUseSegmentation(CurveIndex)	Получить свойство (*)
Object.CurveUseSegmentation(CurveIndex) = CurveUseSegmentation	Установить свойство (*)
CurveUseSegmentation = Object.GetCurveUseSegmentation(CurveIndex)	Получить свойство (**)
Object.SetCurveUseSegmentation(CurveIndex, CurveUseSegmentation)	Установить свойство (**)

Синтаксис COM:

Object.get_CurveUseSegmentation(CurveIndex, &CurveUseSegmentation)	Получить свойство
Object.put_CurveUseSegmentation(CurveIndex, CurveUseSegmentation)	Установить свойство

Входные параметры:

long CurveIndex	- индекс кривой.

EdgesCount – Количество ребер

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesCount = Object.EdgesCount	Получить свойство (*)
EdgesCount = Object.GetEdgesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesCount(&EdgesCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Sketch2 – Основание 2

Интерфейс...

Тип данных: Указатель на интерфейс ISketch

Синтаксис Automation:

Sketch2 = Object.Sketch2	Получить свойство (*)
Object.Sketch2 = Sketch2	Установить свойство (*)
Sketch2 = Object.GetSketch2()	Получить свойство (**)
Object.SetSketch2(Sketch2)	Установить свойство (**)

Синтаксис COM:

Object.get_Sketch2(&Sketch2)	Получить свойство
Object.put_Sketch2(Sketch2)	Установить свойство

UseCommonSegmentationParameters – Единые параметры сегментации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseCommonSegmentationParameters = Object.UseCommonSegmentationParameters	=	Получить свойство (*)
Object.UseCommonSegmentationParameters = UseCommonSegmentationParameters	=	Установить свойство (*)
UseCommonSegmentationParameters = Object.GetUseCommonSegmentationParameters()	=	Получить свойство (**)
Object.SetUseCommonSegmentationParameters(UseCommonSegmentationParameters)		Установить свойство (**)

Синтаксис COM:

Object.get_UseCommonSegmentationParameters(&UseCommonSegmentationParameters)	Получить свойство
Object.put_UseCommonSegmentationParameters(UseCommonSegmentationParameters)	Установить свойство

ISheetMetalLinearRuledShell – методы

AddNewEdge – Добавление ребра после указанного индексом

Интерфейс...

Синтаксис Automation:

BOOL AddNewEdge(long IndexAt);

Синтаксис COM:

HRESULT AddNewEdge(long IndexAt, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

IndexAt - индекс.

DeleteEdge – Удаление ребра

Интерфейс...

Синтаксис Automation :

BOOL DeleteEdge(long Index);

Синтаксис COM :

HRESULT DeleteEdge(long Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Index - индекс.

GetEdgePointParam – Получение параметров точки ребра

Интерфейс...

Синтаксис Automation:

BOOL GetEdgePointParam(long EdgeIndex, BOOL StartPoint, double * X, double * Y, double * Z, double * T, IModelObject * * AssociateVertex);

Синтаксис COM:

HRESULT GetEdgePointParam(long EdgeIndex, BOOL StartPoint, double * X, double * Y, double * Z, double * T, IModelObject * * AssociateVertex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

EdgeIndex - индекс ребра.
StartPoint - TRUE - начальная точка,
- FALSE - конечная точка.

Выходные параметры:

X, Y, Z	- координаты.
T	- параметр кривой.
AssociateVertex	- ассоциативный объект

GetEdgePointParams – Получение параметров ребер

Интерфейс...

Синтаксис Automation:

BOOL GetEdgePointParams(VARIANT * Points1, VARIANT * T1, VARIANT * AssociateVertexes1, VARIANT * Points2, VARIANT * T2, VARIANT * AssociateVertexes2);

Синтаксис COM :

HRESULT GetEdgePointParams(VARIANT * Points1, VARIANT * T1, VARIANT * AssociateVertexes1, VARIANT * Points2, VARIANT * T2, VARIANT * AssociateVertexes2, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

Points1	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 начальных координат вершин.
T1	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 параметров вершин.
AssociateVertexes1	- массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH вершин первой направляющей, связанных с вершинами.
Points2	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 координат вершин ребер на второй направляющей.
T2	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 параметров вершин ребер на первой направляющей.
AssociateVertexes2	массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH вершин второй направляющей, связанных с вершинами.

SetEdgePointParam – Изменение положения точки ребра

Интерфейс...

Синтаксис Automation :

BOOL SetEdgePointParam(long EdgeIndex, BOOL StartPoint, double X, double Y, double Z, double * T, IModelObject * AssociateVertex);

Синтаксис COM :

HRESULT SetEdgePointParam(long EdgeIndex, BOOL StartPoint, double X, double Y, double Z, double * T, IModelObject * AssociateVertex, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

EdgeIndex	- индекс ребра.
StartPoint	- TRUE - начальная точка, - FALSE - конечная точка.
X, Y, Z	- координаты.
T	- параметр кривой.
AssociateVertex	- ассоциативный объект

Интерфейс ISheetMetalRib

Интерфейс операции **Ребро усиления**.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalRib

КОМПАС версия v18

ISheetMetalRib – свойства

ArchCoefficient – Относительная глубина прогиба в%

Интерфейс...

Тип данных: double

Синтаксис Automation:

ArchCoefficient Object.ArchCoefficient	=	Получить свойство (*)

Object.ArchCoefficient = ArchCoefficient	=	Установить свойство (*)
ArchAngle = Object.GetArchCoefficient()	=	Получить свойство (**)
Object.SetArchCoefficient(ArchCoefficient)		Установить свойство (**)

Синтаксис COM:

Object.get_ArchCoefficient(&ArchCoefficient)	Получить свойство
Object.put_ArchCoefficient(ArchCoefficient)	Установить свойство

Версия Компас v18.1

ArchCreate – Создавать прогиб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ArchCreate = Object.ArchCreate	=	Получить свойство (*)
Object.ArchCreate = ArchCreate	=	Установить свойство (*)
ArchCreate = Object.GetArchCreate()	=	Получить свойство (**)
Object.SetArchCreate(ArchCreate)		Установить свойство (**)

Синтаксис COM:

Object.get_ArchCreate(&ArchCreate)	Получить свойство
Object.put_ArchCreate(ArchCreate)	Установить свойство

Версия Компас v18.1

ArchMeasure – Способ задания глубины прогиба

Интерфейс...

Тип данных: из перечисления ksArchMeasureEnum

Синтаксис Automation:

ArchMeasure	=	Получить свойство (*)
Object.ArchMeasure		
Object.ArchMeasure	=	Установить свойство (*)
ArchMeasure		
ArchMeasure	=	Получить свойство (**)
Object.GetArchMeasure()		
Object.SetArchMeasure(		Установить свойство (**)
ArchMeasure)		

Синтаксис COM:

Object.get_ArchMeasure(	Получить свойство
&ArchMeasure)	
Object.put_ArchMeasure(	Установить свойство
ArchMeasure)	

ArchLength – Глубина прогиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

ArchLength	=	Получить свойство (*)
Object.ArchLength		
Object.ArchLength	=	Установить свойство (*)
ArchLength		
ArchLength	=	Получить свойство (**)
Object.GetArchLength()		
Object.SetArchLength(		Установить свойство (**)
ArchLength)		

Синтаксис COM:

Object.get_ArchLength(	Получить
&ArchLength)	свойство
Object.put_ArchLength(	Установить
ArchLength)	свойство

ArchRadius – Радиус прогиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ArchRadius = Object.ArchRadius                  Получить свойство (* )
Object.ArchRadius = ArchRadius                  Установить свойство (* )
ArchRadius = Object.GetArchRadius()             Получить свойство (**)
Object.SetArchRadius( ArchRadius )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ArchRadius(                  Получить свойство
&ArchRadius )
Object.put_ArchRadius(                  Установить свойство
ArchRadius )
```

Версия Компас v18.1

Angle1 – Угол наклона профиля или угол ребра

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle1 = Object.Angle1;         Получить свойство (* )
Object.Angle1 = Angle1;         Установить свойство (* )
Angle1 = Object.GetAngle1();    Получить свойство (**)
Object.SetAngle1( Angle1 );     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Ang                  Получить
le1( &Angle1 );                 свойство
Object.put_Ang                  Установить
le1(  Angle1 );                 свойство
```

Angle2 – Угол наклона сечения

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle2 = Object.Angle2;         Получить свойство (* )
Object.Angle2 = Angle2;         Установить свойство (* )
Angle2 = Object.GetAngle2();    Получить свойство (**)
Object.SetAngle2( Angle2 );     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle2(   &Angle2    Получить свойство
);
Object.put_Angle2(  Angle2 );    Установить свойство
```

BendEdge – Ребро сгиба

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
BendEdge = Object.BendEdge( NewVal );          Получить свойство (* )
BendEdge = Object.GetBendEdge( NewVal );       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_BendEdge(          Получить свойство
NewVal, &BendEdge );
```

Примечание:

Свойство доступно только для чтения.

BendFace – Грань сгиба

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
BendFace = Object.BendFace( NewVal );                    Получить свойство (* )
BendFace = Object.GetBendFace( NewVal );                 Получить свойство (**)
```

Синтаксис COM:

```
Object.get_BendFace(          Получить свойство
NewVal, &BendFace );
```

Примечание:

Свойство доступно только для чтения.

BendObject – Сгиб

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

BendObject = Object.BendObject;	Получить свойство (*)
BendObject = Object.GetBendObject();	Получить свойство (**)

Синтаксис COM:

Object.get_BendObject(&BendObject);	Получить свойство

Примечание:

Свойство доступно только для чтения.

BuildingType – Способ построения

Интерфейс...

Тип данных: Значение из перечисления ksSHRibBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType;	Получить свойство (*)
Object.BuildingType = BuildingType;	Установить свойство (*)
BuildingType = Object.GetBuildingType();	Получить свойство (**)
Object.SetBuildingType(BuildingType);	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType);	Получить свойство
Object.put_BuildingType(BuildingType);	Установить свойство

CutingType – Форма сечения

Интерфейс...

Тип данных: Значение из перечисления ksSHRibCutingTypeEnum.

Синтаксис Automation:

CutingType = Object.CutingType;	Получить свойство (*)
Object.CutingType = CutingType;	Установить свойство (*)

```
CutingType = Object.GetCutingType();        Получить свойство (**)
Object.SetCutingType( CutingType );         Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CutingType(          Получить свойство
&CutingType ));
Object.put_CutingType(          Установить свойство
CutingType );
```

Direction – Направление смещения вдоль кривой сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction;               Получить свойство (* )
Object.Direction = Direction;               Установить свойство (* )
Direction = Object.GetDirection();          Получить свойство (**)
Object.SetDirection( Direction );           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction );     Получить свойство
Object.put_Direction(  Direction );     Установить свойство
```

Значение свойства:

```
TRUE                                    - прямое,
FALSE                                    - обратное.
```

Lenght1 – Длина 1 или глубина ребра

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lenght1 = Object.Lenght1;               Получить свойство (* )
Object.Lenght1 = Lenght1;               Установить свойство (* )
Lenght1 = Object.GetLenght1();          Получить свойство (**)
Object.SetLenght1( Lenght1 );           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Lenght1( &Lenght1 );     Получить свойство
Object.put_Lenght1( Lenght1 );      Установить свойство
```

Lenght2 – Длина 2

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lenght2 = Object.Lenght2;                 Получить свойство (* )
Object.Lenght2 = Lenght2;                 Установить свойство (* )
Lenght2 = Object.GetLenght2();            Получить свойство (**)
Object.SetLenght1( Lenght2 );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Lenght2( &Lenght2 );     Получить свойство
Object.put_Lenght2(  Lenght2 );     Установить свойство
```

Offset – Величина смещения вдоль кривой сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Offset = Object.Offset;             Получить свойство (* )
Object.Offset = Offset;             Установить свойство (* )
Offset = Object.GetOffset();        Получить свойство (**)
Object.SetOffset( Offset);          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Offset( &Offset );       Получить свойство
Object.put_Offset(  Offset );       Установить свойство
```

OffsetType – Тип задания смещения вдоль кривой сгиба

Интерфейс...

Тип данных: Значение из перечисления ksPoint3DCurveParamTypeEnum.

Синтаксис Automation:

```
OffsetType =                    Получить свойство (* )
Object.OffsetType;
Object.OffsetType =             Установить свойство (* )
OffsetType;
OffsetType =                    Получить свойство (**)
Object.GetOffsetType();
Object.SetOffsetType(           Установить свойство (**)
OffsetType );
```

Синтаксис COM:

```
Object.get_OffsetType(          Получить свойство
&OffsetType );
Object.put_OffsetType(          Установить свойство
OffsetType );
```

Radius1 – Радиус ребра

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius1 = Object.Radius1;       Получить свойство (* )
Object.Radius1 = Radius1;       Установить свойство (* )
Radius1 =                       Получить свойство (**)
Object.GetRadius1();
Object.SetRadius1(              Установить свойство (**)
Radius1 );
```

Синтаксис COM:

```
Object.get_Radius1(             Получить свойство
&Radius1 );
Object.put_Radius1(             Установить свойство
Radius1 );
```

Radius2 – Радиус скругления основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius2 = Object.Radius2;       Получить свойство (* )
```

```
Object.Radius2 = Radius2;     Установить свойство (* )
Radius2 =                     Получить свойство (**)
Object.GetRadius2();
Object.SetRadius2(            Установить свойство (**)
Radius2 );
```

Синтаксис COM:

```
Object.get_Radius2(           Получить свойство
&Radius2 );
Object.put_Radius2(           Установить свойство
Radius2 );
```

Round – Скругление основания

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Round = Object.Round;         Получить свойство (* )
Object.Round = Round;         Установить свойство (* )
Round =                       Получить свойство (**)
Object.GetRound();
Object.SetRound( Round );     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Round( &Round );       Получить свойство
Object.put_Round(  Round );       Установить свойство
```

Width – Ширина основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width;         Получить свойство (* )
Object.Width = Width;         Установить свойство (* )
Width = Object.GetWidth();    Получить свойство (**)
Object.SetWidth( Width );     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width );       Получить свойство
```

Object.put_Width(Width);	Установить свойство

ISheetMetalRib – методы

AutoInitBendObjects – Автоматическая инициализация объектов сгиба

Интерфейс...

Синтаксис Automation:

BOOL AutoInitBendObjects(IModelObject * InitObject);

Синтаксис COM:

HRESULT AutoInitBendObjects(IModelObject * InitObject, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

InitObject	- грань или ребро сгиба.

CalculateOptimalParams – Вычислить оптимальные параметры профиля по сгибу

Интерфейс...

Синтаксис Automation:

BOOL CalculateOptimalParams();

Синтаксис COM:

HRESULT CalculateOptimalParams(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Метод пересчитывает глубину, длины 1, 2 и угол наклона профиля в зависимости от выбранного сгиба и типа профиля.

InitBendObjects – Инициализация объектов сгиба

Интерфейс...

Синтаксис Automation:

BOOL InitBendObjects(IModelObject * BendObject, IModelObject * BendFace, IModelObject * BendEdge);

Синтаксис COM:

HRESULT InitBendObjects(IModelObject * BendObject, IModelObject * BendFace, IModelObject * BendEdge, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

BendObject	- операция листового тела - сгиб (Сгиб, Сгиб по линии, Сгиб по эскизу),
BendFace	- грань сгиба,
BendEdge	- ребро сгиба.

Интерфейс ISheetMetalSketchBends

Интерфейс коллекции элементов Сгиб по эскизу.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalSketchBends

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalSketchBends.

КОМПАС версия v18

ISheetMetalSketchBends – свойства

SheetMetalSketchBend – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalSketchBend.

Синтаксис Automation:

SheetMetalSketchBend = Object.SheetMetalSketchBend(Index)	Получить свойство (*)
SheetMetalSketchBend = Object.GetSheetMetalSketchBend(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalSketchBend(Index, &SheetMetalSketchBend)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalSketchBends – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalSketchBend ** Result);

Возвращаемое значение:

	- указатель на интерфейс ISheetMetalSketchBend.

Примечания:

1. Метод позволяет создать новый интерфейс операции сгиб по эскизу.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalSketchBend

Интерфейс параметров операции Сгиб по эскизу.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalSketchBend

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям ISheetMetalSketchBends::Add или свойства ISheetMetalSketchBends::SheetMetalSketchBend.

КОМПАС версия v18

ISheetMetalSketchBend – свойства

BendCoefficient – Коэффициент нейтрального слоя

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BendCoefficient = Получить свойство (* )
Object.BendCoefficient;
Object.BendCoefficient = Установить свойство (* )
BendCoefficient;
BendCoefficient = Получить свойство (**)
Object.GetBendCoefficient();
Object.SetBendCoefficient( Установить свойство (**)
BendCoefficient );
```

Синтаксис COM:

```
Object.get_BendCoefficient( Получить
&BendCoefficient ); свойство
Object.put_BendCoefficient( Установить
BendCoefficient ); свойство
```

BendReduction – Уменьшение сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
BendReduction = Получить свойство (* )
Object.BendReduction;
Object.BendReduction = Установить свойство (* )
BendReduction;
BendReduction = Получить свойство (**)
Object.GetBendReduction();
```

```
Object.SetBendReduction(        Установить свойство (**)
BendReduction );
```

Синтаксис COM:

```
Object.get_BendReduction(       Получить
&BendReduction );               свойство
Object.put_BendReduction(       Установить
BendReduction );                свойство
```

BendRelease – Тип освобождения сгиба

Интерфейс...

Тип данных: Значение из перечисления ksBendReleaseTypeEnum.

Синтаксис Automation:

```
BendRelease                 =   Получить свойство (* )
Object.BendRelease;
Object.BendRelease          =   Установить свойство (* )
BendRelease;
BendRelease                 =   Получить свойство (**)
Object.GetBendRelease();
Object.SetBendRelease(          Установить свойство (**)
BendRelease );
```

Синтаксис COM:

```
Object.get_BendRelease(         Получить
&BendRelease );                 свойство
Object.put_BendRelease(         Установить
BendRelease );                  свойство
```

BendTablePath – Имя файла таблицы сгибов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
BendTablePath               =   Получить свойство (* )
Object.BendTablePath
BendTablePath               =   Получить свойство (**)
Object.GetBendTablePath()
```

Синтаксис COM:

Object.get_BendTablePath(&BendTablePath)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BendValue – Величина сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

BendValue Object.BendValue;	=	Получить свойство (*)
Object.BendValue BendValue;	=	Установить свойство (*)
BendValue Object.GetBendValue();	=	Получить свойство (**)
Object.SetBendValue(BendValue);		Установить свойство (**)

Синтаксис COM:

Object.get_BendValue(&BendValue);	Получить свойство
Object.put_BendValue(BendValue);	Установить свойство

BuildingType – Способ построения сгиба по эскизу

Интерфейс...

Тип данных: из перечисления ksSketchBendBuildingTypeEnum

Синтаксис Automation:

BuildingType Object.BuildingType;	=	Получить свойство (*)
Object.BuildingType BuildingType;	=	Установить свойство (*)
BuildingType Object.GetBuildingType();	=	Получить свойство (**)
Object.SetBuildingType(BuildingType);		Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType);	Получить свойство
Object.put_BuildingType(BuildingType);	Установить свойство

ClosingAngle – Угол при замыкании углов

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingAngle = Object.ClosingAngle(Type);	=	Получить свойство (*)
Object.ClosingAngle(Type) = ClosingAngle;	=	Установить свойство (*)
ClosingAngle = Object.GetClosingAngle(Type);	=	Получить свойство (**)
Object.SetClosingAngle(Type, ClosingAngle);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingAngle(Type, &ClosingAngle);	Получить свойство
Object.put_ClosingAngle(Type, ClosingAngle);	Установить свойство

Входные параметры:

Type	- тип замыкания операции сгиб по эскизу листового тела из перечисления ksClosingTypeEnum.

ClosingClosedType – Способ замыкания углов

Интерфейс...

Тип данных: Значение из перечисления ksClosingClosedTypeEnum.

Синтаксис Automation:

ClosingClosedType = Object.ClosingClosedType(Type);		Получить свойство (*)
Object.ClosingClosedType(Type) = ClosingClosedType;		Установить свойство (*)
ClosingClosedType = Object.GetClosingClosedType(Type);		Получить свойство (**)
Object.SetClosingClosedType(Type, ClosingClosedType);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingClosedType (Type, &ClosingClosedType);	Получить свойство
Object.put_ClosingClosedType (Type, ClosingClosedType);	Установить свойство

Входные параметры:

Type	- тип замыкания операции сгиб по эскизу листового тела из перечисления ksClosingTypeEnum.

ClosingCorneringType – Обработка угла при замыкании

Интерфейс...

Тип данных: Значение из перечисления ksClosingCorneringEnum.

Синтаксис Automation:

ClosingCorneringType = Object.ClosingCorneringType(Type);		Получить свойство (*)
Object.ClosingCorneringType(Type) = ClosingCorneringType;		Установить свойство (*)
ClosingCorneringType = Object.GetClosingCorneringType(Type);		Получить свойство (**)
Object.SetClosingCorneringType(Type, ClosingCorneringType);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingCorneringType(Type, &ClosingCorneringType);	Получить свойство
Object.put_ClosingCorneringType(Type, ClosingCorneringType);	Установить свойство

Входные параметры:

Type	- тип замыкания операции сгиб по эскизу листового тела из перечисления ksClosingTypeEnum.

ClosingEnable – Включить замыкание углов

Интерфейс...

Синтаксис Automation:

ClosingEnable = Object.ClosingEnable(Type);	Получить свойство (*)
Object.ClosingEnable(Type) = ClosingEnable;	Установить свойство (*)
ClosingEnable = Object.GetClosingEnable(Type);	Получить свойство (**)
Object.SetClosingEnable(Type, ClosingEnable);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingEnable(Type, &ClosingEnable);	Получить свойство
Object.put_ClosingEnable(Type, ClosingEnable);	Установить свойство

Входные параметры:

Type	- тип замыкания операции сгиб по эскизу листового тела из перечисления ksClosingTypeEnum.

ClosingGapValue – Значение зазора при замыкании углов

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingGapValue = Object.ClosingGapValue(Type);	=	Получить свойство (*)
Object.ClosingGapValue(Type) = ClosingGapValue;		Установить свойство (*)
ClosingGapValue = Object.GetClosingGapValue(Type);	=	Получить свойство (**)
Object.SetClosingGapValue(Type, ClosingGapValue);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingGapValue(Type, &ClosingGapValue);	Получить свойство
Object.put_ClosingGapValue(Type, ClosingGapValue);	Установить свойство

Входные параметры:

Type	- тип замыкания операции сгиб по эскизу листового тела из перечисления ksClosingTypeEnum.

ClosingHoleDiameter – Диаметр отверстия при замыкании угла

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingHoleDiameter = Object.ClosingHoleDiameter;	=	Получить свойство (*)
Object.ClosingHoleDiameter = ClosingHoleDiameter;	=	Установить свойство (*)
ClosingHoleDiameter = Object.GetClosingHoleDiameter();	=	Получить свойство (**)

Object.SetClosingHoleDiameter(ClosingHoleDiameter);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHoleDiameter(&ClosingHoleDiameter);	Получить свойство
Object.put_ClosingHoleDiameter(ClosingHoleDiameter);	Установить свойство

ClosingHoleOffset – Смещение отверстия при замыкании угла

Интерфейс...

Тип данных: double

Синтаксис Automation:

ClosingHoleOffset = Object.ClosingHoleOffset;	=	Получить свойство (*)
Object.ClosingHoleOffset = ClosingHoleOffset;	=	Установить свойство (*)
ClosingHoleOffset = Object.GetClosingHoleOffset();	=	Получить свойство (**)
Object.SetClosingHoleOffset(ClosingHoleOffset);		Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHoleOffset(&ClosingHoleOffset);	Получить свойство
Object.put_ClosingHoleOffset(ClosingHoleOffset);	Установить свойство

ClosingHolePlacement – Размещение отверстия при круговой обработке угла

Интерфейс...

Тип данных: Значение из перечисления ksClosingHolePlacementEnum.

Синтаксис Automation:

ClosingHolePlacement = Object.ClosingHolePlacement;	=	Получить свойство (*)

Object.ClosingHolePlacement = ClosingHolePlacement;	Установить свойство (*)
ClosingHolePlacement = Object.GetClosingHolePlacement();	Получить свойство (**)
Object.SetClosingHolePlacement(ClosingHolePlacement);	Установить свойство (**)

Синтаксис COM:

Object.get_ClosingHolePlacement(&ClosingHolePlacement);	Получить свойство
Object.put_ClosingHolePlacement(ClosingHolePlacement);	Установить свойство

Direction – Направление построения

Интерфейс...

Тип данных: Значение из перечисления ksDirectionTypeEnum.

Синтаксис Automation:

Direction = Object.Direction;	Получить свойство (*)
Object.Direction = Direction;	Установить свойство (*)
Direction = Object.GetDirection();	Получить свойство (**)
Object.SetDirection(Direction);	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction);	Получить свойство
Object.put_Direction(Direction);	Установить свойство

DismissalAngleType – Способ освобождения угла сгиба

Интерфейс...

Тип данных: Значение из перечисления ksBendAngleReleaseTypeEnum.

Синтаксис Automation:

DismissalAngleType = Object.DismissalAngleType;	=	Получить свойство (*)
Object.DismissalAngleType = DismissalAngleType;	=	Установить свойство (*)
DismissalAngleType = Object.GetDismissalAngleType ();	=	Получить свойство (**)
Object.SetDismissalAngleType (DismissalAngleType);		Установить свойство (**)

Синтаксис COM:

Object.get_DismissalAngleTyp e(&DismissalAngleType);	Получить свойство
Object.put_DismissalAngleTyp e(DismissalAngleType);	Установить свойство

DismissalDepth – Глубина разгрузки сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

DismissalDepth = Object.DismissalDepth;	=	Получить свойство (*)
Object.DismissalDepth = DismissalDepth;	=	Установить свойство (*)
DismissalDepth = Object.GetDismissalDepth();	=	Получить свойство (**)
Object.SetDismissalDepth(DismissalDepth);		Установить свойство (**)

Синтаксис COM:

Object.get_DismissalDepth(&DismissalDepth);	Получить свойство
Object.put_DismissalDepth(DismissalDepth);	Установить свойство

DismissalWidth – Ширина разгрузки сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DismissalWidth = Object.DismissalWidth;     Получить свойство (* )
Object.DismissalWidth = DismissalWidth;     Установить свойство (* )
DismissalWidth = Object.GetDismissalWidth(); Получить свойство (**)
Object.SetDismissalWidth( DismissalWidth );  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DismissalWidth( &DismissalWidth );   Получить свойство
Object.put_DismissalWidth( DismissalWidth );    Установить свойство
```

DismissalWithWidth – Учитывать ширину

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
DismissalWithWidth = Object.DismissalWithWidth;     Получить свойство (* )
Object.DismissalWithWidth = DismissalWithWidth;     Установить свойство (* )
DismissalWithWidth = Object.GetDismissalWithWidth(); Получить свойство (**)
Object.SetDismissalWithWidth ( DismissalWithWidth );  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DismissalWithWidth( &DismissalWithWidth );   Получить свойство
Object.put_DismissalWithWidth(  DismissalWithWidth );   Установить свойство
```

Edges – Массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges;	Получить свойство (*)
Object.Edges = Edges;	Установить свойство (*)
Edges = Object.GetEdges();	Получить свойство (**)
Object.SetEdges(Edges);	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges);	Получить свойство
Object.put_Edges(Edges);	Установить свойство

Примечание:

Массив ребер, вдоль которых должен располагаться сгиб.

InternalRadius – Внутренний радиус

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InternalRadius = Object.InternalRadius;	=	Получить свойство (*)
Object.InternalRadius = InternalRadius;	=	Установить свойство (*)
InternalRadius = Object.GetInternalRadius();	=	Получить свойство (**)
Object.SetInternalRadius(InternalRadius);		Установить свойство (**)

Синтаксис COM:

Object.get_InternalRadius(&InternalRadius);	Получить свойство
Object.put_InternalRadius(InternalRadius);	Установить свойство

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Sketch = Object.Sketch;          Получить свойство (* )
Object.Sketch = Sketch;          Установить свойство (* )
Sketch = Object.GetSketch();     Получить свойство (**)
Object.SetSketch( Sketch );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch );    Получить свойство
Object.put_Sketch(  Sketch );    Установить свойство
```

Примечание:

Эскиз определяет профиль сгиба.

Straighten – Разогнуть

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Straighten = Object.Straighten;  Получить свойство (* )
Object.Straighten = Straighten;  Установить свойство (* )
Straighten =                     Получить свойство (**)
Object.GetStraighten();
Object.SetStraighten(            Установить свойство (**)
Straighten );
```

Синтаксис COM:

```
Object.get_Straighten(           Получить свойство
&Straighten );
Object.put_Straighten(           Установить свойство
Straighten );
```

Radius – Радиус сгиба

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius = Object.Radius;          Получить свойство (* )
Object.Radius = Radius;          Установить свойство (* )
Radius = Object.GetRadius();     Получить свойство (**)
Object.SetRadius( Radius );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius( &Radius );    Получить свойство
Object.put_Radius(  Radius );    Установить свойство
```

Width1 – Ширина в прямом направлении

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width1 = Object.Width1;          Получить свойство (* )
Object.Width1 = Width1;          Установить свойство (* )
Width1 = Object.GetWidth1();     Получить свойство (**)
Object.SetWidth1( Width1 );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width1( &Width1 );    Получить свойство
Object.put_Width1(  Width1 );    Установить свойство
```

Width2 – Ширина в обратном направлении

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width2 = Object.Width2;          Получить свойство (* )
Object.Width2 = Width2;          Установить свойство (* )
Width2 = Object.GetWidth2();     Получить свойство (**)
```

Object.SetWidth2(Width2);	Установить свойство (**)

Синтаксис COM:

Object.get_Width2(&Width2);	Получить свойство
Object.put_Width2(Width2);	Установить свойство

WithoutAngleRelease – Без освобождения углов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithoutAngleRelease = Object.WithoutAngleRelease;	=	Получить свойство (*)
Object.WithoutAngleRelease = WithoutAngleRelease;	=	Установить свойство (*)
WithoutAngleRelease = Object.GetWithoutAngleRelease();	=	Получить свойство (**)
Object.SetWithoutAngleRelease(WithoutAngleRelease);		Установить свойство (**)

Синтаксис COM:

Object.get_WithoutAngleRelease(&WithoutAngleRelease);	Получить свойство
Object.put_WithoutAngleRelease(WithoutAngleRelease);	Установить свойство

WithoutBendRelease – Без освобождения сгиба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithoutBendRelease = Object.WithoutBendRelease;	=	Получить свойство (*)
Object.WithoutBendRelease = WithoutBendRelease;	=	Установить свойство (*)

WithoutBendRelease = Object.GetWithoutBendRelease();	=	Получить свойство (**)
Object.SetWithoutBendRelease(WithoutBendRelease);		Установить свойство (**)

Синтаксис COM:

Object.get_WithoutBendRelease(&WithoutBendRelease);	Получить свойство
Object.put_WithoutBendRelease(WithoutBendRelease);	Установить свойство

UnfoldType – Способ определения длины развертки

Интерфейс...

Тип данных: Значение из перечисления ksUnfoldTypeEnum.

Синтаксис Automation:

UnfoldType = Object.UnfoldType;	=	Получить свойство (*)
Object.UnfoldType = UnfoldType;	=	Установить свойство (*)
UnfoldType = Object.GetUnfoldType();	=	Получить свойство (**)
Object.SetUnfoldType(UnfoldType);		Установить свойство (**)

Синтаксис COM:

Object.get_UnfoldType(&UnfoldType);	Получить свойство
Object.put_UnfoldType(UnfoldType);	Установить свойство

Интерфейс ISheetMetalPlates

Интерфейс коллекции элементов Пластина.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalPlates

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalPlates.

КОМПАС версия v18

ISheetMetalPlates – свойства

SheetMetalPlate – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalPlate.

Синтаксис Automation:

SheetMetalPlate = Object.SheetMetalPlate(Index)		Получить свойство (*)
SheetMetalPlate Object.GetSheetMetalPlate(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalPlate(Index, &SheetMetalPlate)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalPlates – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalPlate * * Result);

Возвращаемое значение:

- указатель на интерфейс ISheetMetalPlate.

Примечания:

1. Метод позволяет создать новый интерфейс операции “пластина”.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalPlate

Интерфейс операции Пластина.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalPlate

Данный интерфейс можно получить с помощью метода коллекции операций “пластина” ISheetMetalPlates::Add или свойства ISheetMetalPlates::SheetMetalPlate.

КОМПАС версия v18

ISheetMetalPlate – свойства

IsUserThickness – Признак толщины, заданной пользователем

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsUserThickness = Object.IsUserThickness;	=	Получить свойство (*)
Object.IsUserThickness = IsUserThickness;	=	Установить свойство (*)
IsUserThickness = Object.GetIsUserThickness();	=	Получить свойство (**)
Object.SetIsUserThickness(IsUserThickness);		Установить свойство (**)

Синтаксис COM:

Object.get_IsUserThickness(&IsUserThickness);	Получить свойство
Object.put_IsUserThickness(IsUserThickness)	Установить свойство

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Sketch = Object.Sketch;          Получить свойство (* )
Object.Sketch = Sketch;          Установить свойство (* )
Sketch = Object.GetSketch();     Получить свойство (**)
Object.SetSketch( Sketch );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch );    Получить
                                 свойство
Object.put_Sketch(  Sketch )     Установить
                                 свойство
```

Thickness – Толщина пластины

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Thickness = Object.Thickness;    Получить свойство (* )
Object.Thickness = Thickness;    Установить свойство (* )
Thickness                  =     Получить свойство (**)
Object.GetThickness();
Object.SetThickness(             Установить свойство (**)
Thickness );
```

Синтаксис COM:

```
Object.get_Thickness(            Получить
&Thickness );                    свойство
Object.put_Thickness(            Установить
Thickness )                      свойство
```

ThicknessObject – Объект (листовое тело), задающий толщину

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

ThicknessObject = Object.ThicknessObject;	=	Получить свойство (*)
Object.ThicknessObject = ThicknessObject;	=	Установить свойство (*)
ThicknessObject = Object.GetThicknessObject();	=	Получить свойство (**)
Object.SetThicknessObject(ThicknessObject)		Установить свойство (**)

Синтаксис COM:

Object.get_ThicknessObject(&ThicknessObject);	Получить свойство
Object.put_ThicknessObject(ThicknessObject)	Установить свойство

Интерфейс ISheetMetalPressFormings

Интерфейс коллекции элементов Открытая или Закрытая штамповка.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalPressFormings

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalPressFormings.

КОМПАС версия v18

ISheetMetalPressFormings – свойства

SheetMetalPressForming – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalPressForming.

Синтаксис Automation:

SheetMetalPressForming = Object.SheetMetalPressForming (Index)	=	Получить свойство (*)

SheetMetalPressForming = Object.GetSheetMetalPressForming(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalPressForming(Index, &SheetMetalPressForming)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalPressFormings – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum Type, ISheetMetalPressForming * * Result);

Возвращаемое значение:

	- указатель на интерфейс ISheetMetalPressForming.

Входные параметры:

Type	- тип объекта из перечисления ksObj3dTypeEnum.

Примечания:

1. Метод позволяет создать новый интерфейс операций Открытая или Закрытая штамповка.
2. Допустимыми значениями Type являются o3d_sheetMetalDrawnCutout (Закрытая штамповка), o3d_sheetMetalDimpleCutout (Открытая штамповка)

3. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalPressForming

Интерфейс операции Открытая /Закрытая штамповка.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalPressForming

Данный интерфейс можно получить с помощью метода коллекции операций пластина ISheetMetalPressFormings::Add или свойства ISheetMetalPressFormings::SheetMetalPressForming.

КОМПАС версия v18

ISheetMetalPressForming – свойства

Angle – Угол уклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle;          Получить свойство (* )
Object.Angle = Angle;          Установить свойство (* )
Angle = Object.GetAngle();     Получить свойство (**)
Object.SetAngle( Angle )       Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle );    Получить свойство
Object.put_Angle(  Angle )     Установить свойство
```

Direction – Направление построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction;    Получить свойство (* )
```

```
Object.Direction = Direction;       Установить свойство (* )
Direction                    =      Получить свойство (**)
Object.GetDirection();
Object.SetDirection( Direction      Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_Direction(               Получить
&Direction );                       свойство
Object.put_Direction(               Установить
Direction )                         свойство
```

Значение свойства:

```
TRUE                                - прямое,
FALSE                               - обратное.
```

Height – Высота

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Height = Object.Height;             Получить свойство (* )
Object.Height = Height;             Установить свойство (* )
Height = Object.GetHeight();        Получить свойство (**)
Object.SetHeight( Height )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Height( &Height );       Получить свойство
Object.put_Height( Height )         Установить свойство
```

HeightType – Способ задания высоты

Интерфейс...

Тип данных:

Синтаксис Automation:

```
HeightType                   =      Получить свойство (* )
Object.HeightType;
Object.HeightType            =      Установить свойство (* )
HeightType;
```

```
HeightType                  =   Получить свойство (**)
Object.GetHeightType();
Object.SetHeightType(           Установить свойство (**)
HeightType )
```

Синтаксис COM:

```
Object.get_HeightType(              Получить
&HeightType );                      свойство
Object.put_HeightType(              Установить
HeightType )                        свойство
```

Radius1 – Радиус скругления ребер

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius1 = Object.Radius1;       Получить свойство (* )
Object.Radius1 = Radius1;       Установить свойство (* )
Radius1                 =   Получить свойство (**)
Object.GetRadius1();
Object.SetRadius1( Radius1 )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius1( &Radius1 )      Получить свойство
Object.put_Radius1(  Radius1 )      Установить свойство
```

Radius2 – Радиус скругления основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius2 = Object.Radius2;       Получить свойство (* )
Object.Radius2 = Radius2;       Установить свойство (* )
Radius2                 =   Получить свойство (**)
Object.GetRadius2();
Object.SetRadius2( Radius2 )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius2( &Radius2 )      Получить свойство
```

```
Object.put_Radius2(  Radius2 )          Установить свойство
```

Radius3 – Радиус скругления дна в закрытой штамповке

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius3 = Object.Radius3;          Получить свойство (* )
Object.Radius3 = Radius3;          Установить свойство (* )
Radius3                       =    Получить свойство (**)
Object.GetRadius3();
Object.SetRadius3( Radius3 )       Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius3( &Radius3 )          Получить свойство
Object.put_Radius3(  Radius3 )          Установить свойство
```

Round – Скругление основания

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Round = Object.Round;              Получить свойство (* )
Object.Round = Round;              Установить свойство (* )
Round = Object.GetRound();         Получить свойство (**)
Object.SetRound( Round );          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Round( &Round )      Получить свойство
Object.put_Round(  Round )      Установить свойство
```

RoundBottom – Скругление дна в закрытой штамповке

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
RoundBottom                   =    Получить свойство (* )
Object.RoundBottom ;
```

Object.RoundBottom RoundBottom;	=	Установить свойство (*)
RoundBottom Object.GetRoundBottom();	=	Получить свойство (**)
Object.SetRoundBottom(RoundBottom);		Установить свойство (**)

Синтаксис COM:

Object.get_RoundBottom(&RoundBottom)	Получить свойство
Object.put_RoundBottom(RoundBottom)	Установить свойство

RoundEdges – Скругление ребер

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

RoundEdges Object.RoundEdges ;	=	Получить свойство (*)
Object.RoundEdges RoundEdges;	=	Установить свойство (*)
RoundEdges Object.GetRoundEdges();	=	Получить свойство (**)
Object.SetRoundEdges(RoundEdges);		Установить свойство (**)

Синтаксис COM:

Object.get_RoundEdges(&RoundEdges)	Получить свойство
Object.put_RoundBottom(RoundBottom)	Установить свойство

Side – Неподвижная грань

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Side = Object.Side;	Получить свойство (*)
Object.Side = Side;	Установить свойство (*)

```
Side = Object.GetSide();          Получить свойство (**)
Object.SetSide( Side )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Side( &Side );         Получить свойство
Object.put_Side(  Side )          Установить свойство
```

Значение свойства:

```
TRUE                              - наружу,
FALSE                             - внутрь.
```

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Sketch = Object.Sketch;           Получить свойство (* )
Object.Sketch = Sketch;           Установить свойство (* )
Sketch = Object.GetSketch();      Получить свойство (**)
Object.SetSketch( Sketch )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch );     Получить свойство
Object.put_Sketch(  Sketch )      Установить свойство
```

ThicknessDirection – Направление добавления толщины стенок штамповки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ThicknessDirection          =   Получить свойство (* )
Object.ThicknessDirection;
Object.ThicknessDirection   =   Установить свойство (* )
ThicknessDirection;
ThicknessDirection          =   Получить свойство (**)
Object.GetThicknessDirection(
);
```

Object.SetThicknessDirection(ThicknessDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ThicknessDirection (&ThicknessDirection);	Получить свойство
Object.put_ThicknessDirection (ThicknessDirection)	Установить свойство

Значение свойства:

TRUE	- внутрь,
FALSE	- наружу.

Интерфейс ISheetMetalShoulders

Интерфейс коллекции элементов Буртик.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISheetMetalShoulders

Данный интерфейс можно получить, используя свойство контейнера объектов гибки ISheetMetalContainer::SheetMetalShoulders.

КОМПАС версия v18

ISheetMetalShoulders – свойства

SheetMetalShoulder – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISheetMetalShoulder.

Синтаксис Automation:

SheetMetalShoulder Object.SheetMetalShoulder(Index)	=	Получить свойство (*)
SheetMetalShoulder Object.GetSheetMetalShoulder(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetMetalShoulder(Index, &SheetMetalShoulder)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ISheetMetalShoulders – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISheetMetalShoulder * * Result);

Возвращаемое значение:

	- указатель на интерфейс ISheetMetalShoulder.

Примечания:

1. Метод позволяет создать новый интерфейс операций Буртик.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ISheetMetalShoulder

Интерфейс операции Буртик.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISheetMetalShoulder

Данный интерфейс можно получить с помощью метода коллекции операций пластина ISheetMetalShoulders::Add или свойства ISheetMetalShoulders::SheetMetalShoulder.

КОМПАС версия v18

ISheetMetalShoulder – свойства

BuildingType – Способ построения

Интерфейс...

Тип данных: Значение из перечисления ksShoulderBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType;	=	Получить свойство (*)
Object.BuildingType = BuildingType;	=	Установить свойство (*)
BuildingType = Object.GetBuildingType();	=	Получить свойство (**)
Object.SetBuildingType(BuildingType)		Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType);	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

CutingType – Форма сечения

Интерфейс...

Тип данных: Значение из перечисления ksShoulderCutingTypeEnum.

Синтаксис Automation:

CutingType = Object.CutingType;	=	Получить свойство (*)
Object.CutingType = CutingType;	=	Установить свойство (*)
CutingType = Object.GetCutingType();	=	Получить свойство (**)
Object.SetCutingType(CutingType)		Установить свойство (**)

Синтаксис COM:

Object.get_CutingType(&CutingType);	Получить свойство
Object.put_CutingType(CutingType)	Установить свойство

Direction – Направление построения: TRUE – Прямое, FALSE – Обратное

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction;	Получить свойство (*)
Object.Direction = Direction;	Установить свойство (*)
Direction = Object.GetDirection();	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction);	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Height – Высота

Интерфейс...

Тип данных: double

Синтаксис Automation:

Height = Object.Height;	Получить свойство (*)
Object.Height = Height;	Установить свойство (*)
Height = Object.GetHeight();	Получить свойство (**)
Object.SetHeight(Height)	Установить свойство (**)

Синтаксис COM:

Object.get_Height(&Height);	Получить свойство
Object.put_Height(Height)	Установить свойство

GapValue – Зазор

Интерфейс...

Тип данных: double

Синтаксис Automation:

GapValue = Object.GapValue;	Получить свойство (*)

```
Object.GapValue = GapValue;      Установить свойство (* )
GapValue                    =    Получить свойство (**)
Object.GetGapValue();
Object.SetGapValue( GapValue     Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_GapValue(                    Получить
&GapValue );                            свойство
Object.put_GapValue(                    Установить
GapValue )                              свойство
```

Radius1 – Радиус буртика

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius1 = Object.Radius1;        Получить свойство (* )
Object.Radius1 = Radius1;        Установить свойство (* )
Radius1                     =    Получить свойство (**)
Object.GetRadius1();
Object.SetRadius1( Radius1 )     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius1( &Radius1 );      Получить свойство
Object.put_Radius1(  Radius1 )       Установить свойство
```

Radius2 – Радиус скругления основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius2 = Object.Radius2;        Получить свойство (* )
Object.Radius2 = Radius2;        Установить свойство (* )
Radius2                     =    Получить свойство (**)
Object.GetRadius2();
Object.SetRadius2( Radius2 )     Установить свойство (**)
```

Синтаксис COM:

Object.get_Radius2(&Radius2);	Получить свойство
Object.put_Radius2(Radius2)	Установить свойство

Round – Скругление основания

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Round = Object.Round;	Получить свойство (*)
Object.Round = Round;	Установить свойство (*)
Round = Object.GetRound();	Получить свойство (**)
Object.SetRound(Round)	Установить свойство (**)

Синтаксис COM:

Object.get_Round(&Round);	Получить свойство
Object.put_Round(Round)	Установить свойство

Sketch – Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Sketch = Object.Sketch;	Получить свойство (*)
Object.Sketch = Sketch;	Установить свойство (*)
Sketch = Object.GetSketch();	Получить свойство (**)
Object.SetSketch(Sketch)	Установить свойство (**)

Синтаксис COM:

Object.get_Sketch(&Sketch);	Получить свойство
Object.put_Sketch(Sketch)	Установить свойство

ShoulderType – Тип. Способ обработки концов буртика

Интерфейс...

Тип данных: Значение из перечисления ksShoulderTypeEnum.

Синтаксис Automation:

```
ShoulderType = Object.ShoulderType;     Получить свойство (* )
Object.ShoulderType = ShoulderType;     Установить свойство (* )
ShoulderType = Object.GetShoulderType(); Получить свойство (**)
Object.SetShoulderType( ShoulderType )  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ShoulderType( &ShoulderType );   Получить свойство
Object.put_ShoulderType( ShoulderType )     Установить свойство
```

Width1 – Ширина основания

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width1 = Object.Width1;          Получить свойство (* )
Object.Width1 = Width1;          Установить свойство (* )
Width1 = Object.GetWidth1();     Получить свойство (**)
Object.SetWidth1( Width1 )       Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width1( &Width1 );    Получить свойство
Object.put_Width1( Width1 )      Установить свойство
```

Width2 – Ширина дна

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width2 = Object.Width2;          Получить свойство (* )
Object.Width2 = Width2;          Установить свойство (* )
Width2 = Object.GetWidth2();     Получить свойство (**)
Object.SetWidth2( Width2 )       Установить свойство (**)
```

Синтаксис COM:

Object.get_Width2(&Width2);	Получить свойство
Object.put_Width2(Width2)	Установить свойство

Интерфейс ISheetMetalUndercut

Интерфейс операции Подсечка.

Иерархия:

IDispatch

ISheetMetalUndercut

Интерфейс является дополнительным к интерфейсу ISheetMetalLineBend для операций подсечка создаваемых через коллекцию ISheetMetalContainer::SheetMetalUndercuts.

Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

КОМПАС версия v18

ISheetMetalUndercut – свойства

Distance – Расстояние

Интерфейс...

Тип данных: double

Синтаксис Automation:

Distance = Object.Distance	Получить свойство (*)
Object.Distance = Distance	Установить свойство (*)
Distance = Object.GetDistance()	Получить свойство (**)
Object.SetDistance (Distance)	Установить свойство (**)

Синтаксис COM:

Object.get_Distance(&Distance)	Получить свойство
Object.put_Distance(Distance)	Установить свойство

DistanceType – Задание размера

Интерфейс...

Тип данных: Значение из перечисления ksUndercutDistanceTypeEnum.

Синтаксис Automation:

DistanceType = Object.DistanceType	Получить свойство (*)

Object.DistanceType = DistanceType	Установить свойство (*)
DistanceType = Object.GetDistanceType()	Получить свойство (**)
Object.SetDistanceType (DistanceType)	Установить свойство (**)

Синтаксис COM:

Object.get_DistanceType(&DistanceType)	Получить свойство
Object.put_DistanceType(DistanceType)	Установить свойство

WithAddMaterial – С добавлением материала

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

WithAddMaterial = Object.WithAddMaterial	=	Получить свойство (*)
Object.WithAddMaterial = WithAddMaterial	=	Установить свойство (*)
WithAddMaterial = Object.GetWithAddMaterial()	=	Получить свойство (**)
Object.SetWithAddMaterial(WithAddMaterial)		Установить свойство (**)

Синтаксис COM:

Object.get_WithAddMaterial(&WithAddMaterial)	Получить свойство
Object.put_WithAddMaterial(WithAddMaterial)	Установить свойство

Оформление

Контейнер

Интерфейс ISymbols3DContainer

Интерфейс контейнера условных обозначений 3D.

Описание:

Позволяет получить коллекции 3D размеров и обозначений.

Примечание:

Дополнительный интерфейс компонента. Данный интерфейс можно получить у компонента IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ISymbols3DContainer – свойства

AngleDimensions3D – Интерфейс коллекции угловых размеров 3D

Интерфейс...

Тип данных: указатель на интерфейс IAngleDimensions3D

Синтаксис Automation:

AngleDimensions3D = Object.AngleDimensions3D		Получить свойство (*)
AngleDimensions3D = Object.GetAngleDimensions3D()		Получить свойство (**)

Синтаксис COM:

Object.get_AngleDimensions3D(&AngleDimensions3D)		Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать угловые и угловые с обрывом размеры.
2. Свойство доступно только для чтения.

Bases3D – Интерфейс коллекции обозначений базы 3D

Интерфейс...

Тип данных: указатель на интерфейс IBases3D

Синтаксис Automation:

Bases3D Object.Bases3D	=	Получить свойство (*)
Bases3D Object.GetBases3D()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Bases3D(&Bases3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать обозначения базы 3D.
2. Свойство доступно только для чтения.

DiametralDimensions3D – Интерфейс коллекции диаметральных размеров 3D

Интерфейс...

Тип данных: указатель на интерфейс IDiametralDimensions3D

Синтаксис Automation:

DiametralDimensions3D = Object.DiametralDimensions3D	Получить свойство (*)
DiametralDimensions3D = Object.GetDiametralDimensions3D()	Получить свойство (**)

Синтаксис COM:

Object.get_DiametralDimensions3D(&DiametralDimensions3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать диаметральные размеры.
2. Свойство доступно только для чтения.

Leaders3D – Интерфейс коллекции линий-выносок 3D

Интерфейс...

Тип данных: указатель на интерфейс ILeaders3D

Синтаксис Automation:

Leaders3D = Object.Leaders3D	Получить свойство (*)
Leaders3D = Object.GetLeaders3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Leaders3D(&Leaders3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать линии-выноски.
2. Свойство доступно только для чтения.

LineDimensions3D – Интерфейс коллекции линейных размеров 3D

Интерфейс...

Тип данных: указатель на интерфейс ILineDimensions3D

Синтаксис Automation:

LineDimensions3D = Object.LineDimensions 3D	Получить свойство (*)
LineDimensions3D = Object.GetLineDimensi ons3D()	Получить свойство (**)

Синтаксис COM:

Object.get_LineDimen sions3D(&LineDimensions3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать линейные размеры.
2. Свойство доступно только для чтения.

RadialDimensions3D – Интерфейс коллекции радиальных размеров 3D

Интерфейс...

Тип данных: указатель на интерфейс IRadialDimensions3D

Синтаксис Automation:

RadialDimensions3D = Object.RadialDimensio ns3D	Получить свойство (*)

RadialDimensions3D = Object.GetRadialDimensions3D()	Получить свойство (**)

Синтаксис COM:

Object.get_RadialDimensions3D(&RadialDimensions3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать радиальные размеры.
2. Свойство доступно только для чтения.

Roughs3D – Интерфейс коллекции обозначений шероховатости 3D

Интерфейс...

Тип данных: указатель на интерфейс IRoughs3D

Синтаксис Automation:

Roughs3D Object.Roughs3D	=	Получить свойство (*)
Roughs3D Object.GetRoughs3D()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Roughs3D(&Roughs3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать обозначения шероховатости 3D.
2. Свойство доступно только для чтения.

Threads – Коллекция условных обозначений резьбы

Интерфейс...

Тип данных: Указатель на интерфейс IThreads

Синтаксис Automation:

Threads Object.Threads	=	Получить свойство (*)

Threads Object.GetThreads()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Roughs3D(&Roughs3D)	Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

Tolerances3D – Интерфейс коллекции обозначений допусков формы 3D

Интерфейс...

Тип данных: указатель на интерфейс ITolerances3D

Синтаксис Automation:

Tolerances3D Object.Tolerances3D	=	Получить свойство (*)
Tolerances3D Object.GetTolerances3D()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Tolerances3D(&Tolerances3D)	Получить свойство (*)

Примечание:

1. Свойство позволяет получать и создавать обозначения допусков формы 3D.
2. Свойство доступно только для чтения.

ISymbols3DContainer – методы

CreateGenerativeDimensions – Отобразить размеры эскизов и операций

Интерфейс...

Синтаксис Automation:

BOOL CreateGenerativeDimensions(VARIANT Objects, BOOL CreateZeroDimensions);

Синтаксис COM:

HRESULT CreateGenerativeDimensions(VARIANT Objects, BOOL CreateZeroDimensions , BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- объект или массив объектов (VT_DISPATCH или VT_ARRAY | VT_DISPATCH), для которых нужно отобразить размеры,
CreateZeroDimensions	- создавать размеры с нулевым значением.

GetDimensionVariable – Получить переменную, связанную с размером

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDimensionVariable(LPDISPATCH Dimension);

Синтаксис COM:

HRESULT GetDimensionVariable(IModelObject * Dimension, IVariable7 * Result);

Возвращаемое значение:

указатель на интерфейс переменной IVariable7.

Входные параметры:

Dimensions	- указатель на размер, для которого надо получить переменную.

Размеры

Интерфейс IAngleDimensions3D

Справка системы КОМПАС: Команда Угловой размер (документ-модель)

`kompas.chm::/CM_ANGLE_DIMENSION_3D.htm`

Интерфейс коллекции угловых размеров 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObject

IAngleDimensions3D

Описание:

Интерфейс позволяет получить и создавать угловые размеры в 3D документе.

Коллекция содержит угловые и угловые с обрывом размеры.

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols3DContainer::AngleDimensions3D.

IAngleDimensions3D – свойства

AngleDimension3D –Указатель на угловой размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс углового размера IAngleDimension3D

Синтаксис Automation:

AngleDimension iObject.AngleDimension3D(Index);	=	Получить свойство (*)
AngleDimension iObject.GetAngleDimension3D(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AngleDimension3D(Index, &AngleDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IAngleDimensions3D – методы

Add – Добавить угловой размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IAngleDimension3D** Result);

Возвращаемое значение:

- указатель на интерфейс углового размера IAngleDimension3D.

Интерфейс IDiametralDimensions3D

Справка системы КОМПАС: Команда Диаметральный размер (документ-модель)

kompas.chm::/CM_DIAMETR_DIMENSION_3D.htm

Интерфейс коллекции диаметральных размеров 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObject

IDiametralDimensions3D

Описание:

Интерфейс позволяет получить и создавать диаметральные размеры в 3D документе.

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols3DContainer::DiametralDimensions3D.

IDiametralDimensions3D – свойства

DiametralDimension3D – Указатель на диаметральный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс диаметрального размера IDiametralDimension3D

Синтаксис Automation:

DiametralDimension = iObject.DiametralDimension3D(Index);		Получить свойство (*)
DiametralDimension = iObject.GetDiametralDimension3D(Index);		Получить свойство (**)

Синтаксис COM:

iObject->get_DiametralDimension3D(Index, &DiametralDimension)	Получить свойство

Входные параметры:

Index (Variant)		Индекс размера в коллекции. Поддерживаются следующие типы:
	- VT_I4	- индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IDiametralDimensions3D – методы

Add – Добавить диаметральный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IDiametralDimension3D** Result);

Возвращаемое значение:

	- Указатель на интерфейс диаметрального размера IDiametralDimension3D.

Интерфейс ILineDimensions3D

Справка системы КОМПАС: Команда Линейный размер (документ-модель)

`kompas.chm::/CM_LINE_DIMENSION_3D_PLANE.htm`

Команда Линейный размер от отрезка до точки (документ-модель)

`kompas.chm::/CM_LINE_DIMENSION_3D.htm`

Интерфейс коллекции линейных размеров 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObject

ILineDimensions3D

Описание:

Интерфейс позволяет получить и создавать линейные размеры (от отрезка до точки и на плоскости) в 3D документе.

Коллекция содержит линейные размеры (от отрезка до точки и на плоскости).

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols3DContainer::LineDimensions3D.

ILineDimensions3D – свойства

LineDimension3D – Указатель на линейный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс линейного размера IBaseLineDimension3D

Синтаксис Automation:

LineDimension = iObject.LineDimension3D(Index);	=	Получить свойство (*)
LineDimension = iObject.GetLineDimension3D(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LineDimension3D(Index, &LineDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ILineDimensions3D – методы

Add – Добавить линейный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum DimType, [out, retval] IBaseLineDimension3D** Result);

Входной параметр:

DimType	- тип объекта из перечисления ksObj3dTypeEnum.

Возвращаемое значение:

- указатель на интерфейс линейного размера IBaseLineDimension3D.

Примечание:

Для создания линейного размера от отрезка до точки нужно задать значение параметра DimType, равное o3d_baselineDimension3D.

Для создания линейного размера на плоскости нужно задать значение параметра DimType, равное o3d_lineDimension3D.

Интерфейс IRadialDimensions3D

Справка системы КОМПАС: Команда Радиальный размер (документ-модель)

kompas.chm::/CM_RADIUS_DIMENSION_3D.htm

Интерфейс коллекции радиальных размеров 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObject

IRadialDimensions3D

Описание:

Интерфейс позволяет получить и создавать радиальные размеры в 3D документе.

Данный интерфейс можно получить у контейнера условных обозначений, используя свойство ISymbols3DContainer::RadialDimensions3D.

IRadialDimensions3D – свойства

RadialDimension3D – Указатель на радиальный размер, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс радиального размера IRadialDimension3D

Синтаксис Automation:

RadialDimension iObject.RadialDimension3D(Index);	=	Получить свойство (*)
RadialDimension iObject.GetRadialDimension3D(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_RadialDimension3D(Index, &RadialDimension)	Получить свойство

Входные параметры:

Index (Variant)	- Индекс размера в коллекции. Поддерживаются следующие типы: - VT_I4 - индекс размера.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IRadialDimensions3D – методы

Add – Добавить радиальный размер в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATH Add( );

Синтаксис COM:

HRESULT Add([out, retval] IRadialDimension3D** Result);

Возвращаемое значение:

	- указатель на интерфейс радиального размера IRadialDimension3D.

Интерфейс IAngleDimension3D

Справка системы КОМПАС: Команда Угловой размер (документ-модель)

`kompas.chm::/CM_ANGLE_DIMENSION_3D.htm`

Интерфейс параметров углового размера 3D.

Иерархия:

IKompasAPIObject

IModelObject

IAngleDimension3D

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства углового размера в 3D документе.

Интерфейс можно получить у коллекции угловых размеров в 3D документе, используя свойство IAngleDimensions3D::AngleDimension3D или метод IAngleDimensions3D::Add.

После задания параметров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

При создании углового размера используются прямолинейные и плоские объекты, которые являются сторонами угла.

Прямолинейные объекты:

- отрезок в эскизе,
- сегмент ломаной,
- вспомогательная ось,
- ребро тела или поверхности.

Плоские объекты:

- координатная плоскость,
- вспомогательная плоскость,
- грань тела или поверхности.

3. Свойство IDimensionText::NominalValue возвращает угол.

IAngleDimension3D – свойства

DimensionType – Тип углового размера

Интерфейс...

Тип данных: из перечисления ksAngleDimTypeEnum

Синтаксис Automation:

DimensionType = Object.DimensionType	=	Получить свойство (*)
Object.DimensionType = DimensionType	=	Установить свойство (*)
DimensionType = Object.GetDimensionType()	=	Получить свойство (**)
Object.SetDimensionType(DimensionType)		Установить свойство (**)

Синтаксис COM:

Object.get_Dimension Type(&DimensionType)	Получить свойство (*)
Object.put_Dimension Type(DimensionType)	Установить свойство (*)

Length – Длина выносной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Length = Object.Length    Получить свойство (* )
Object.Length = Length    Установить свойство (* )
Object.Length = Length    Получить свойство (**)
Object.SetLength(         Установить свойство (**)
Length )
```

Синтаксис COM:

```
Object.get_Length( &Length )    Получить свойство (* )
Object.put_Length(  Length )    Установить свойство (* )
```

Object1 – Первый объект

Интерфейс...

Справка системы КОМПАС: угловой размер

`kompas.chm::/CM_ANGLE_DIMENSION_3D.htm`

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

```
Object1 = Object.Object1          Получить свойство (* )
Object.Object1 = Object1          Установить свойство (* )
Object1 = Object.GetObject1()     Получить свойство (**)
Object.SetObject1( Object1 )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Object1(    Получить свойство (* )
&Object1 )
Object.put_Object1(    Установить свойство (* )
Object1 )
```

Примечание:

При создании углового размера используются прямолинейные и плоские объекты, которые являются сторонами угла.

Прямолинейные объекты:

- отрезок в эскизе,
- сегмент ломаной,
- вспомогательная ось,

- ребро тела или поверхности.

Плоские объекты:

- координатная плоскость,
- вспомогательная плоскость,
- грань тела или поверхности.

Object2 – Второй объект

Интерфейс...

Справка системы КОМПАС: угловой размер,

kompas.chm::/CM_ANGLE_DIMENSION_3D.htm

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Object2 = Object.Object2	Получить свойство (*)
Object.Object2 = Object2	Установить свойство (*)
Object2 = Object.GetObject2()	Получить свойство (**)
Object.SetObject2(Object2)	Установить свойство (**)

Синтаксис COM:

Object.get_Object2(&Object2)	Получить свойство (*)
Object.put_Object2(Object2)	Установить свойство (*)

Примечание:

При создании углового размера используются прямолинейные и плоские объекты, которые являются сторонами угла.

Прямолинейные объекты:

- отрезок в эскизе,
- сегмент ломаной,
- вспомогательная ось,
- ребро тела или поверхности.

Плоские объекты:

- координатная плоскость,
- вспомогательная плоскость,
- грань тела или поверхности.

IAngleDimension3D – методы

GetCenterPoint – Получить координаты центра

Интерфейс...

Синтаксис Automation:

BOOL GetCenterPoint(double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetCenterPoint(double * X,
double * Y,
double * Z,
BOOL * Result);

Выходные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Inverse – Изменить направление размерных линий

Интерфейс...

Синтаксис Automation:

BOOL Inverse( );

Синтаксис COM:

HRESULT Inverse(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет изменять направление размерных линий на противоположное, сохраняя их длину.

SetCenterPoint – Установить координаты центра

Интерфейс...

Синтаксис Automation:

BOOL SetCenterPoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetCenterPoint(double X,
double Y,

double Z,
BOOL * Result);

Входные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет задавать новое положение размера на оси поверхности.

Изменение положения размера доступно для незафиксированного размера, у которого измеряемым объектом является цилиндрическая, коническая или тороидальная поверхность. Заданная точка будет спроецирована на ось поверхности.

Интерфейс IBaseLineDimension3D

Справка системы КОМПАС: Команда Линейный размер от отрезка до точки (документ-модель)

`kompas.chm::/CM_LINE_DIMENSION_3D.htm`

Интерфейс параметров линейного размера от отрезка до точки 3D.

Иерархия:

IKompasAPIObject

IModelObject

IBaseLineDimension3D

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства линейного размера от отрезка до точки в 3D документе.

Интерфейс можно получить у коллекции линейных размеров в 3D документе, используя свойство ILineDimensions3D::LineDimension3D или метод ILineDimensions3D::Add.

После задания параметров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

1. Интерфейс позволяет проставить линейный размер между прямолинейным и точечным объектами. При простановке линейного размера расстояния от ребра до точки в качестве первого объекта можно использовать только прямолинейные объекты, в качестве второ-

го объекта можно использовать только точечные объекты. При простановке линейного размера на плоскости можно использовать и прямолинейные и точечные объекты.

Прямолинейные объекты:

- отрезок в эскизе,
- сегмент ломаной,
- координатная ось,
- вспомогательная ось,
- ребро тела или поверхности.

Точечные объекты:

- точка в эскизе,
- пространственная точка,
- начало координат,
- вершина пространственной кривой,
- вершина тела или поверхности.

2. Свойство IDimensionText::NominalValue возвращает расстояние.

IBaseLineDimension3D – свойства

Length – Длина выносной линии

Интерфейс...

Тип данных: double

Синтаксис COM:

Object.get_Length(&Length)	Получить свойство (*)
Object.put_Length(Length)	Установить свойство (*)

Синтаксис Automation:

Length = Object.Length	Получить свойство (*)
Object.Length = Length	Установить свойство (*)
Length = Object.GetLength()	Получить свойство (**)
Object.SetLength(Length)	Установить свойство (**)

Object1 – Первый объект

Интерфейс...

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Object1 = Object.Object1	Получить свойство (*)
Object.Object1 = Object1	Установить свойство (*)
Object1 = Object.GetObject1()	Получить свойство (**)

Object.SetObject1(Object1)	Установить свойство (**)

Синтаксис COM:

Object.get_Object1(&Object1)	Получить свойство (*)
Object.put_Object1(Object1)	Установить свойство (*)

Примечание:

При создании линейного размера через по ребру и точке в качестве первого объекта можно использовать только прямолинейные объекты. При создании линейного размера на плоскости можно использовать и прямолинейные и точечные объекты.

Object2 – Второй объект

Интерфейс...

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Object2 = Object.Object2	Получить свойство (*)
Object.Object2 = Object2	Установить свойство (*)
Object2 = Object.GetObject2()	Получить свойство (**)
Object.SetObject2(Object2)	Установить свойство (**)

Синтаксис COM:

Object.get_Object2(&Object2)	Получить свойство (*)
Object.put_Object2(Object2)	Установить свойство (*)

Примечание:

При создании линейного размера через по ребру и точке в качестве первого объекта можно использовать только прямолинейные объекты. При создании линейного размера на плоскости можно использовать и прямолинейные и точечные объекты.

Интерфейс ILineDimension3D

Справка системы КОМПАС: Команда Линейный размер (документ-модель)

`kompas.chm::/CM_LINE_DIMENSION_3D_PLANE.htm`

Интерфейс параметров линейного размера на плоскости 3D.

Иерархия:

IKompasAPIObject

IModelObject

ILineDimension3D

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства линейного размера в 3D документе.

Интерфейс можно получить у коллекции линейных размеров в 3D документе, используя свойство ILineDimensions3D::LineDimension3D или метод ILineDimensions3D::Add.

После задания парамеров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

В качестве направляющей плоскости можно использовать:

- координатную плоскость,
- вспомогательную плоскость,
- грань тела или поверхности.

ILineDimension3D – свойства

Plane – Базовая плоскость

Интерфейс...

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Plane = Object.Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство (*)
Object.put_Plane(Plane)	Установить свойство (*)

Примечание:

В качестве направляющей плоскости можно использовать:

- координатную плоскость,
- вспомогательную плоскость,
- грань тела или поверхности.

Интерфейс IDiametralDimension3D

Справка системы КОМПАС: Команда Диаметральный размер (документ-модель)

`kompas.chm::/CM_DIAMETR_DIMENSION_3D.htm`

Интерфейс параметров диаметрального размера 3D.

Иерархия:

IKompasAPIObject

IModelObject

IDiametralDimension3D

IDimensionText

IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства диаметрального размера в 3D документе.

Интерфейс можно получить у коллекции диаметральных размеров, используя свойство IDiametralDimensions3D::DiametralDimension3D или метод IDiametralDimensions3D::Add.

После задания параметров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

1. При создании радиального и диаметрального размеров используются следующие объекты:

- окружность (дуга окружности) в эскизе,
- ребро тела или поверхности, имеющее форму окружности (дуги окружности),
- грань тела или поверхности, имеющая цилиндрическую, коническую, сферическую или тороидальную форму.

При выборе в качестве базового объекта окружности в эскизе размер проставляется в плоскости эскиза.

При выборе в качестве базового объекта ребра тела или поверхности размер проставляется в плоскости, в которой находится выбранное ребро.

Положение размера можно задавать произвольно или фиксировать.

Способы задания произвольного положения размера:

- установите координаты центра с помощью функции SetCenterPoint. Координаты будут спроецированы на ось грани.
- укажите точку на поверхности или ребре с помощью функции SetSurfacePoint.

Для фиксации положения размера укажите точечный объект или плоский объект, параллельный плоскости простановки размера. Размер будет проставлен в плоскости, проходящей через объект фиксации. Между размером и объектом фиксации формируется ассоциативная связь. Благодаря этой связи размер следует за объектом фиксации при изменении положения последнего.

2. Свойство IDimensionText::NominalValue возвращает диаметр.

IDiametralDimension3D – свойства

Angle – Угол наклона размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство (*)
Object.put_Angle(Angle)	Установить свойство (*)

Примечание:

Свойство позволяет задавать угол наклона размерной линии в плоскости размера.

DimensionType – Тип диаметрального размера (Полная размерная линия\Размерная линия с обрывом)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DimensionType = Object.DimensionType	Получить свойство (*)
Object.DimensionType = DimensionType	Установить свойство (*)
DimensionType = Object.GetDimensionType()	Получить свойство (**)
Object.SetDimensionType(DimensionType)	Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)	Получить свойство (*)
Object.put_DimensionType(DimensionType)	Установить свойство (*)

Object1 – Измеряемый объект

Интерфейс...

Справка системы КОМПАС: выбор базового объекта

kompas.chm::/192_24_1_2_Ukazanie_obwekta_dlj.htm

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Object1 = Object.Object1	Получить свойство (*)
Object.Object1 = Object1	Установить свойство (*)
Object1 = Object.GetObject1()	Получить свойство (**)
Object.SetObject1(Object1)	Установить свойство (**)

Синтаксис COM:

Object.get_Object1(&Object1)	Получить свойство (*)
Object.put_Object1(Object1)	Установить свойство (*)

Примечание:

При создании радиального и диаметрального размеров используются следующие объекты:

- окружность (дуга окружности) в эскизе,
- ребро тела или поверхности, имеющее форму окружности (дуги окружности),
- грань тела или поверхности, имеющая цилиндрическую, коническую, сферическую или тороидальную форму.

При выборе в качестве базового объекта окружности в эскизе размер проставляется в плоскости эскиза.

При выборе в качестве базового объекта ребра тела или поверхности размер проставляется в плоскости, в которой находится выбранное ребро.

PlaneObject – Фиксирующий объект

Интерфейс...

Справка системы КОМПАС: выбор базового объекта

`kompas.chm::/192_24_1_2_Ukazanie_obwekta_dlj.htm`

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

PlaneObject = Object.PlaneObject	Получить свойство (*)
Object.PlaneObject = PlaneObject	Установить свойство (*)
PlaneObject = Object.GetPlaneObject()	Получить свойство (**)
Object.SetPlaneObject(PlaneObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PlaneObject(&PlaneObject)	Получить свойство (*)
Object.put_PlaneObject(PlaneObject)	Установить свойство (*)

Примечание:

Положение размера можно зафиксировать. Для фиксации положения размера укажите точечный объект или плоский объект, параллельный плоскости простановки размера.

Размер будет проставлен в плоскости, проходящей через объект фиксации. Между размером и объектом фиксации формируется ассоциативная связь. Благодаря этой связи размер следует за объектом фиксации при изменении положения последнего.

Для снятия фиксации установите значение свойства == NULL.

IDiametralDimension3D – методы

GetCenterPoint – Получить координаты центра

Интерфейс...

Синтаксис Automation:

```
BOOL GetCenterPoint( double * X,
double * Y,
double * Z );
```

Синтаксис COM:

```
HRESULT GetCenterPoint( double * X,
double * Y,
double * Z,
BOOL * Result );
```

Выходные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetSurfacePoint – Получить точку пересечения размера с поверхностью или ребром

Интерфейс...

Синтаксис Automation:

```
BOOL GetSurfacePoint( double * X,
double * Y,
double * Z );
```

Синтаксис COM:

```
HRESULT GetSurfacePoint( double * X,
double * Y,
double * Z,
BOOL * Result );
```

Выходные параметры:

X, Y, Z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetCenterPoint – Установить координаты центра

Интерфейс...

Синтаксис Automation:

BOOL SetCenterPoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetCenterPoint(double X,
double Y,
double Z,
BOOL * Result);

Входные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет задавать новое положение размера на оси поверхности.

Изменение положения размера доступно для незафиксированного размера, у которого измеряемым объектом является цилиндрическая, коническая или тороидальная поверхность. Заданная точка будет спроецирована на ось поверхности.

SetSurfacePoint – Установить точку пересечения размера с поверхностью или ребром

Интерфейс...

Справка системы КОМПАС: особенности простановки размера

kompas.chm::/ploskost_razmera.htm

Синтаксис Automation:

BOOL SetSurfacePoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetSurfacePoint(double X,
double Y,
double Z,
BOOL * Result);

Входные параметры:

X, Y, Z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет задавать новое положение размера относительно грани или ребра. При указании точки на поверхности точка центра пересчитается.

Интерфейс IRadialDimension3D

Справка системы КОМПАС: Команда Радиальный размер (документ-модель)

kompas.chm::/CM_RADIUS_DIMENSION_3D.htm

Интерфейс параметров радиального размера 3D.

Иерархия:

IKompasAPIObject
    IModelObject
        IRadialDimension3D
IDimensionText
IDimensionParams

Описание:

Интерфейс позволяет получить и задать свойства радиального размера в 3D документе.

Интерфейс можно получить у коллекции радиальных размеров, используя свойство IRadialDimensions3D::RadialDimension3D или метод IRadialDimensions3D::Add.

После задания параметров размера требуется вызвать метод IModelObject::Update.

Интерфейсы IDimensionText и IDimensionParams являются дополнительными. Их можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

1. При создании радиального и диаметрального размеров используются следующие объекты:

- окружность (дуга окружности) в эскизе,
- ребро тела или поверхности, имеющее форму окружности (дуги окружности),
- грань тела или поверхности, имеющая цилиндрическую, коническую, сферическую или тороидальную форму.

При выборе в качестве базового объекта окружности в эскизе размер проставляется в плоскости эскиза.

При выборе в качестве базового объекта ребра тела или поверхности размер проставляется в плоскости, в которой находится выбранное ребро.

Положение размера можно задавать произвольно или фиксировать.

Способы задания произвольного положения размера:

- установите координаты центра с помощью функции SetCenterPoint. Координаты будут спроецированы на ось грани.
- укажите точку на поверхности или ребре с помощью функции SetSurfacePoint.

Для фиксации положения размера укажите точечный объект или плоский объект, параллельный плоскости простановки размера. Размер будет проставлен в плоскости, проходящей через объект фиксации. Между размером и объектом фиксации формируется ассоциативная связь. Благодаря этой связи, размер следует за объектом фиксации при изменении положения последнего.

2. Свойство IDimensionText::NominalValue возвращает радиус.

IRadialDimension3D – свойства

Angle – Угол наклона размерной линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство (*)
Object.put_Angle(Angle)	Установить свойство (*)

Примечание:

Свойство позволяет задавать угол наклона размерной линии в плоскости размера.

DimensionType – Тип радиального размера (От центра\не от центра)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DimensionType = Object.DimensionType	=	Получить свойство (*)
Object.DimensionType = DimensionType	=	Установить свойство (*)
DimensionType = Object.GetDimensionType()	=	Получить свойство (**)
Object.SetDimensionType(DimensionType)		Установить свойство (**)

Синтаксис COM:

Object.get_DimensionType(&DimensionType)		Получить свойство (*)
Object.put_DimensionType(DimensionType)		Установить свойство (*)

Object1 – Измеряемый объект

Интерфейс...

Справка системы КОМПАС: выбор базового объекта

`kompas.chm::/192_24_1_2_Ukazanie_obwekta_dlj.htm`

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

Object1 = Object.Object1	Получить свойство (*)
Object.Object1 = Object1	Установить свойство (*)
Object1 = Object.GetObject1()	Получить свойство (**)
Object.SetObject1(Object1)	Установить свойство (**)

Синтаксис COM:

Object.get_Object1(&Object1)	Получить свойство (*)
Object.put_Object1(Object1)	Установить свойство (*)

Примечание:

При создании радиального и диаметрального размеров используются следующие объекты:

- окружность (дуга окружности) в эскизе,
- ребро тела или поверхности, имеющее форму окружности (дуги окружности),
- грань тела или поверхности, имеющая цилиндрическую, коническую, сферическую или тороидальную форму.

При выборе в качестве базового объекта окружности в эскизе размер проставляется в плоскости эскиза.

При выборе в качестве базового объекта ребра тела или поверхности размер проставляется в плоскости, в которой находится выбранное ребро.

PlaneObject – Фиксирующий объект

Интерфейс...

Справка системы КОМПАС: выбор базового объекта

`kompas.chm::/192_24_1_2_Ukazanie_obwekta_dlj.htm`

Тип данных: указатель на интерфейс модельного объекта IModelObject

Синтаксис Automation:

PlaneObject = Object.PlaneObject	Получить свойство (*)
Object.PlaneObject = PlaneObject	Установить свойство (*)
PlaneObject = Object.GetPlaneObject()	Получить свойство (**)
Object.SetPlaneObject(PlaneObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PlaneObject(&PlaneObject)	Получить свойство (*)
Object.put_PlaneObject(PlaneObject)	Установить свойство (*)

Примечание:

Положение размера можно зафиксировать. Для фиксации положения размера укажите точечный объект или плоский объект, параллельный плоскости простановки размера. Размер будет проставлен в плоскости, проходящей через объект фиксации. Между размером и объектом фиксации формируется ассоциативная связь. Благодаря этой связи размер следует за объектом фиксации при изменении положения последнего.

Для снятия фиксации установите значение свойства == NULL.

IRadialDimension3D – методы

GetCenterPoint – Получить координаты центра

Интерфейс...

Синтаксис Automation:

BOOL GetCenterPoint(double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetCenterPoint(double * X,
double * Y,
double * Z,
BOOL * Result);

Выходные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetSurfacePoint – Получить точку пересечения размера с поверхностью или ребром

Интерфейс...

Синтаксис Automation:

BOOL GetSurfacePoint(double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetSurfacePoint(double * X,
double * Y,
double * Z,
BOOL * Result);

Выходные параметры:

X, Y, Z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetCenterPoint – Установить координаты центра

Интерфейс...

Синтаксис Automation:

BOOL SetCenterPoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetCenterPoint(double X,
double Y,
double Z,
BOOL * Result);

Входные параметры:

X, Y, Z	- координаты точки на оси поверхности или центр окружности или дуги.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет задавать новое положение размера на оси поверхности.

Изменение положения размера доступно для незафиксированного размера, у которого измеряемым объектом является цилиндрическая, коническая или тороидальная поверхность. Заданная точка будет спроецирована на ось поверхности.

SetSurfacePoint – Установить точку пересечения размера с поверхностью или ребром

Интерфейс...

Синтаксис Automation:

BOOL SetSurfacePoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetSurfacePoint(double X,
double Y,
double Z,
BOOL * Result);

Входные параметры:

X, Y, Z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция позволяет задавать новое положение размера относительно грани или ребра. При указании точки на поверхности точка центра пересчитается.

Обозначения

Интерфейс IBases3D

Справка системы КОМПАС: Команда База (документ-модель)

`kompas.chm::/CM_BASE_3D.htm`

Интерфейс коллекции обозначений базы 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

IBases3D

Описание:

Интерфейс позволяет получить обозначение базы 3D.

Примечание:

Получить интерфейс коллекции можно используя свойство контейнера условных обозначений ISymbols3DContainer::Bases3D.

IBases3D – свойства

Base3D – Обозначение базы, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс IBase3D

Синтаксис Automation:

```
Base3D = iObject.Base3D( Index );        Получить свойство (* )
Base3D = iObject.GetBase3D( Index );     Получить свойство (**)
```

Синтаксис COM:

iObject->get_Base3D(&Base3D)	Index,	Получить свойство

Входные параметры:

Index	VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IBases3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IBase3D ** Result);

Возвращаемое значение:

- Указатель на интерфейс базы 3D IBase3D.

Интерфейс ILeaders3D

Справка системы КОМПАС...

KOMPAS.chm::/CM_LEADER_3D.htm

Интерфейс коллекции линий-выносок 3D.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ILeaders3D

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера условных обозначений 3D ISymbols3DContainer::Leaders3D.

ILeaders3D – свойства

Leader3D – Указатель на элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IBaseLeader3D

Синтаксис Automation:

```
Leader3D = Object.Leader3D( Index )        Получить свойство (* )
Leader3D = Object.GetLeader3D( Index )     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Leader3D(    Index,             Получить свойство
&Leader3D )
```

Входные параметры:

```
Index                   - индекс объекта.
```

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

ILeaders3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum DimType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum DimType, IBaseLeader3D ** Result);

Входные параметры:

```
Dim     - тип линии-выноски.
Typ
e
```

Типы линий-выносок:

```
o3d_leader3D        86    Линия-выноска 3D,
```

o3d_markLeader3D	87	Знак маркировки 3D,
o3d_positionLeader3D	89	Обозначение позиции 3D,
o3d_brandLeader3D	90	Знак клеймения 3D.

Возвращаемое значение:

- Указатель на интерфейс линии выноски IBaseLeader3D.

Возвращаемое значение:

- Указатель на интерфейс присоединительной точки IConjunctivePoint.

Примечание:

1. Метод позволяет создать новый интерфейс присоединительной точки.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IModelObject::Update.

Интерфейс IRoughs3D

Справка системы КОМПАС: Команда Шероховатость (документ-модель)

`kompas.chm::/CM_ROUGH_3D.htm`

Интерфейс коллекции обозначений шероховатости 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

IRoughs3D

Описание:

Позволяет получать и создавать обозначение 3D шероховатости.

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера условных обозначений ISymbols3DContainer::Roughs3D.

IRoughs3D – свойства

Rough3D – Обозначение шероховатости 3D, заданное по индексу

Интерфейс...

Тип данных: указатель на интерфейс IRough3D

Синтаксис Automation:

Rough3D = iObject.Rough3D(Index);	Получить свойство (*)
Rough3D = iObject.GetRough3D(Index);	Получить свойство (**)

Синтаксис COM:

iObject->get_Rough3D(&Rough3D)	Index,	Получить свойство

Входные параметры:

Index	VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IRoughs3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IRough3D ** Result);

Возвращаемое значение:

- Указатель на интерфейс шероховатости 3D IRough3D.

Интерфейс ITolerances3D

Справка системы КОМПАС: Команда Допуск формы (документ - модель)

`kompas.chm::/CM_TOLERANCE_3D.htm`

Интерфейс коллекции допусков формы 3D.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

ITolerances3D

Описание:

Позволяет получать и создавать обозначения допуска формы 3D.

Примечание:

Получить интерфейс коллекции можно используя свойство контейнера условных обозначений ISymbols3DContainer::Tolerances3D.

ITolerances3D – свойства

Tolerance3D – Указатель на допуск формы 3D, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ITolerance

Синтаксис Automation:

Tolerance = iObject.Tolerance(Index);		Получить свойство (*)
Tolerance iObject.GetTolerance(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_Tolerance(&Tolerance)	Index,	Получить свойство

Входные параметры:

Index	- VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ITolerances3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ITolerance3D ** Result);

Возвращаемое значение:

- Указатель на интерфейс допуска формы Itolerance3D.

Интерфейс IBase3D

Справка системы КОМПАС: Команда База (документ-модель)

kompas.chm::/CM_BASE_3D.htm

Интерфейс обозначения базы 3D.

Иерархия:

IKompasAPIObject

IModelObject

IBase3D

Примечание:

Интерфейс можно получить у коллекции обозначений базы 3D, используя свойство IBases3D::Base3D или метод IBases3D::Add.

IBase3D – свойства

AutoSorted – Автосортировка

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- включить опцию,
FALSE	- отключить опцию.

Синтаксис Automation:

AutoSorted = Object.AutoSorted	Получить свойство (*)
Object.AutoSorted = AutoSorted	Установить свойство (*)
AutoSorted = Object.GetAutoSorted()	Получить свойство (**)
Object.SetAutoSorted(AutoSorted)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSorted(&AutoSorted)	Получить свойство
Object.put_AutoSorted(AutoSorted)	Установить свойство

Примечание.

1. По умолчанию значение свойства равно TRUE.
2. При отключенной опции доступно редактирование текста 3D базы с помощью свойства IBase3D::Text.

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

BasePlane = Object.BasePlane	Получить свойство (*)
Object.BasePlane = BasePlane	Установить свойство (*)
BasePlane = Object.GetBasePlane()	Получить свойство (**)
Object.SetBasePlane(BasePlane)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePlane(&BasePlane)	Получить свойство
Object.put_BasePlane(BasePlane)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать тип плоскости, в которой располагается объект, из перечисления ksObj3dTypeEnum.

BaseObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство

Примечание.

Свойство доступно только для чтения.

DrawType – Способ отрисовки обозначения базы

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- перпендикулярно к объекту,
FALSE	- произвольное расположение.

Синтаксис Automation:

DrawType = Object.DrawType	Получить свойство (*)
Object.DrawType = DrawType	Установить свойство (*)
DrawType = Object.GetDrawType()	Получить свойство (**)
Object.SetDrawType(DrawType)	Установить свойство (**)

Синтаксис COM:

Object.get_DrawType(&DrawType)	Получить свойство
Object.put_DrawType(DrawType)	Установить свойство

Примечание.

По умолчанию значение свойства равно TRUE.

PositionObject – Положение плоскости обозначения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

PositionObject = Object.PositionObject	Получить свойство (*)

```
Object.PositionObject = PositionObject          Установить свойство (* )
PositionObject = Object.GetPositionObject()     Получить свойство (**)
Object.SetPositionObject( PositionObject )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PositionObject(     Получить свойство
&PositionObject )
Object.put_PositionObject(     Установить свойство
PositionObject )
```

Примечание.

Свойство позволяет устанавливать и получать объект, на котором будет располагаться плоскость.

Text – Текст базы

Интерфейс...

Синтаксис Automation:

```
Text = Object.Text              Получить свойство (* )
Text = Object.GetText()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Text( &Text )        Получить свойство
```

Примечание.

1. Свойство доступно только для чтения.
2. Свойство позволяет получить указатель на интерфейс текста IText для редактирования текста 3D базы. Указатель на интерфейс IText вернётся только в случае отключенной опции "Автосортировка". Если автосортировка текста включена, свойство вернёт NULL.

IBase3D – методы

GetPosition – Получить точку, задающую положение плоскости объекта

Интерфейс...

Синтаксис Automation:

```
BOOL GetPosition ( double * x,
double * x,
double * z );
```

Синтаксис COM:

HRESULT GetPosition(double * x,
double * y
double * z
BOOL* Result);

Выходные параметры:

x, y, z	- координаты точки, задающей положения объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetBranchBeginPoint – Получить точку, привязки обозначения базы на поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetBranchBeginPoint(double * x,
double * x,
double * z);

Синтаксис COM:

HRESULT GetBranchBeginPoint(double * x,
double * y,
double * z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки привязки обозначения базы на поверхности.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetBranchEndPoint – Получить точку положения надписи обозначения базы

Интерфейс...

Синтаксис Automation:

BOOL GetBranchEndPoint(double * x,
double * x,
double * z);

Синтаксис COM:

HRESULT GetBranchEndPoint(double * x,
double * y,
double * z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки положения надписи обозначения базы.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetBranchBeginPoint – Установить точку привязки обозначения базы на поверхности

Интерфейс...

Синтаксис Automation:

BOOL SetBranchBeginPoint(double x,
double x,
double z);

Синтаксис COM:

HRESULT SetBranchBeginPoint(double x,
double y,
double z,
IModelObject * Object,
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки привязки обозначения базы на поверхности.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetBranchEndPoint – Установить точку положения надписи обозначения базы

Интерфейс...

Синтаксис Automation:

BOOL SetBranchEndPoint(double x,
double x,
double z);

Синтаксис COM:

HRESULT SetBranchEndPoint(double x,
double y,
double z,
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки положения надписи обозначения базы.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetPosition – Получить точку, задающую положение плоскости объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPosition (double x,
double x,
double z);

Синтаксис COM:

HRESULT SetPosition(double x,
double y
double z
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки, задающей положения объекта;

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IBaseLeader3D

Справка системы КОМПАС: команда Линия-выноска (документ-модель)

KOMPAS.chm::/CM_LEADER_3D.htm

Интерфейс обозначения линии-выноски 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IBaseLeader3D

ILeader

IPositionLeader

IBrandLeader

IMarkLeader

IBranchs3D

IChangeLeader

Примечание:

1. Интерфейс можно получить у коллекции линий выносок 3D, используя свойство ILeaders3D::Leader3D или метод ILeaders3D::Add.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительные интерфейсы:

Тип линии-выноски	ksObj3dTypeEnum	Интерфейс,
Простая линия-выноска	o3d_leader3D	ILeader,
Линия-выноска для обозначения позиции	o3d_positionLeader3D	IPositionLeader,
Линия-выноска для для обозначения клеймения	o3d_brandLeader3D	IBrandLeader,

Линия-выноска для обозначения маркирования	o3d_markLeader3D	IMarkLeader,
Знак изменения		IChangeLeader,
Ответвления		IBranchs3D.

3. После задания параметров обозначения требуется вызвать метод IModelObject::Update.

IBaseLeader3D – свойства

ArrowType – Тип стрелки линии-выноски

Интерфейс...

Тип данных: из перечисления ksArrowEnum

Синтаксис Automation:

ArrowType = Object.ArrowType	Получить свойство (*)
Object.ArrowType = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType()	Получить свойство (**)
Object.SetArrowType(ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(&ArrowType)	Получить свойство
Object.put_ArrowType(ArrowType)	Установить свойство

Входные параметры:

Index	- индекс ответвления (long).

Примечание.

Свойство позволяет устанавливать и получать тип стрелки ответвления.

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Значения свойства:

o3d_unknown	0	Неизвестный
o3d_planeXOY	1	Плоскость XOY
o3d_planeXOZ	2	Плоскость XOZ
o3d_planeYOZ	3	Плоскость YOZ

Синтаксис Automation:

BasePlane = Object.BasePlane	Получить свойство (*)
Object.BasePlane = BasePlane	Установить свойство (*)
BasePlane = Object.GetBasePlane()	Получить свойство (**)
Object.SetBasePlane(BasePlane)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePlane(&BasePlane)	Получить свойство
Object.put_BasePlane(BasePlane)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать тип плоскости из перечисления ksObj3dTypeEnum, в которой располагается объект.

PositionObject – Объект, задающий положение плоскости обозначения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

PositionObject = Object.PositionObject	Получить свойство (*)
Object.PositionObject = PositionObject	Установить свойство (*)

PositionObject = Object.GetPositionObject()	Получить свойство (**)
Object.SetPositionObject(PositionObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PositionObject(&PositionObject)	Получить свойство
Object.put_PositionObject(PositionObject)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать объект, на котором будет располагаться плоскость объекта.

IBaseLeader3D – методы

GetPosition – Получить точку, задающую положение объекта

Интерфейс...

Синтаксис Automation:

```
BOOL GetPosition ( double * x,
double * y,
double * z );
```

Синтаксис COM:

```
HRESULT GetPosition( double * x,
double * y
double * z
BOOL   * Result);
```

Выходные параметры:

x, y, z	- координаты точки, задающей положение объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetPosition – Установить точку, задающую положение объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPosition (double x,
double y,
double z);

Синтаксис COM:

HRESULT SetPosition(double x,
double y
double z
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки, задающей положение объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IRough3D

Справка системы КОМПАС: Команда Шероховатость (документ-модель)

`kompas.chm::/CM_ROUGH_3D.htm`

Интерфейс обозначения шероховатости 3D.

Иерархия:

IDispatch
    IKompasAPIObject
        IModelObject
            IRough3D

IRoughParams

Примечание:

1. Интерфейс можно получить у коллекции обозначений шероховатости 3D, используя свойство IRoughs3D::Rough3D или метод IRoughs3D::Add.
2. После задания параметров объекта требуется вызвать метод IModelObject_Update.
3. Интерфейс IRoughParams является дополнительным, его можно получить с помощью метода IUnknown::QueryInterface.

IRough3D – свойства

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

BasePlane = Object.BasePlane	Получить свойство (*)
Object.BasePlane = BasePlane	Установить свойство (*)
BasePlane = Object.GetBasePlane()	Получить свойство (**)
Object.SetBasePlane(BasePlane)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePlane(&BasePlane)	Получить свойство
Object.put_BasePlane(BasePlane)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать тип плоскости, в которой располагается объект, из перечисления ksObj3dTypeEnum.

PositionObject – Положение плоскости обозначения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

PositionObject = Object.PositionObject	Получить свойство (*)
Object.PositionObject = PositionObject	Установить свойство (*)
PositionObject = Object.GetPositionObject()	Получить свойство (**)
Object.SetPositionObject(PositionObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PositionObject(&PositionObject)	Получить свойство
Object.put_PositionObject(PositionObject)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать объект, на котором будет располагаться плоскость объекта.

IRough3D – методы

BaseObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство

Примечание.

Свойство доступно только для чтения.

BasePosition – Установить точку начала выносной линии или привязки знака

Интерфейс...

Синтаксис Automation:

```
BOOL SetBasePosition( double * x,
double * x,
double * z );
```

Синтаксис COM:

```
HRESULT SetBasePosition( double * x,
double * y,
double * z,
IModelObject * Object,
BOOL   * Result);
```

Входные параметры:

x, y, z	- координаты точки начала выносной линии или привязки знака.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetBasePosition – Получить точку начала выносной линии или привязки знака

Интерфейс...

Синтаксис Automation:

BOOL GetBasePosition(double * x,
double * x,
double * z);

Синтаксис COM:

HRESULT GetBasePosition(double * x,
double * y,
double * z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки начала выносной линии или привязки знака.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetPosition – Получить точку, задающую положение плоскости объекта

Интерфейс...

Синтаксис Automation:

BOOL GetPosition (double * x,
double * x,
double * z);

Синтаксис COM:

HRESULT GetPosition(double * x,
double * y

double * z
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки, задающей положения объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetShelfPosition – Получить точку начала полки

Интерфейс...

Синтаксис Automation:

BOOL GetShelfPosition(double * x,
double * x,
double * z);

Синтаксис COM:

HRESULT GetShelfPosition(double * x,
double * y,
double * z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки начала полки.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetPosition – Установить точку, задающую положение плоскости объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPosition (double x,
double x,
double z);

Синтаксис COM:

HRESULT SetPosition(double * x,
double * y
double * z
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки, задающей положения объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetShelfPosition – Установить точку начала полки

Интерфейс...

Синтаксис Automation:

BOOL SetShelfPosition(double x,
double x,
double z);

Синтаксис COM:

HRESULT SetShelfPosition(double x,
double y,
double z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки начала полки.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс ITolerance3D

Справка системы КОМПАС: Команда Допуск формы (документ - модель)

kompas.chm::/CM_TOLERANCE_3D.htm

Интерфейс обозначения допуска формы 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ITolerance3D

IToleranceParam

IBranchs3D

ITable

Примечание:

1. Интерфейс можно получить у коллекции обозначений допуска формы 3D, используя свойство ITolerances3D::Tolerance3D или метод ITolerances3D::Add.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительные интерфейсы:

- IToleranceParam - интерфейс параметров допуска формы,
- IBranchs3D - интерфейс для работы с ответвлениями в 3D,
- ITable - интерфейс таблицы.

3. После задания параметров обозначения требуется вызвать метод IModelObject::Update.

ITolerance3D – свойства

BranchPos – Положение ножки

Интерфейс...

Тип данных: из перечисления ksTablePointEnum

Синтаксис Automation:

BranchPos = Object.BranchPos(Index)	Получить свойство (*)
Object.BranchPos(Index) = BranchPos	Установить свойство (*)
BranchPos = Object.GetBranchPos(Index)	Получить свойство (**)
Object.SetBranchPos(Index, BranchPos)	Установить свойство (**)

Синтаксис COM:

Object.get_BranchPos(Index, &BranchPos)	Получить свойство
Object.put_BranchPos(Index, BranchPos)	Установить свойство

Входные параметры:

long Index	- индекс ответвления.

ArrowType – Тип стрелки ответвления

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- стрелка,
FALSE	- треугольник.

Синтаксис Automation:

ArrowType = Object.ArrowType(Index)	Получить свойство (*)
Object.ArrowType(Index) = ArrowType	Установить свойство (*)
ArrowType = Object.GetArrowType(Index)	Получить свойство (**)
Object.SetArrowType(Index, ArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ArrowType(Index, &ArrowType)	Получить свойство
Object.put_ArrowType(Index, ArrowType)	Установить свойство

Входные параметры:

Index	- индекс ответвления (long).

Примечание.

Свойство позволяет устанавливать и получать тип стрелки ответвления.

BasePlane – Базовая плоскость

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Значения свойства:

o3d_unknown	0	Неизвестный
o3d_planeXOY	1	Плоскость XOY
o3d_planeXOZ	2	Плоскость XOZ
o3d_planeYOZ	3	Плоскость YOZ

Синтаксис Automation:

BasePlane = Object.BasePlane	Получить свойство (*)
Object.BasePlane = BasePlane	Установить свойство (*)
BasePlane = Object.GetBasePlane()	Получить свойство (**)
Object.SetBasePlane(BasePlane)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePlane(&BasePlane)	Получить свойство
Object.put_BasePlane(BasePlane)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать тип плоскости из перечисления ksObj3dTypeEnum, в которой располагается объект.

PositionObject – Положение плоскости обозначения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

PositionObject = Object.PositionObject	Получить свойство (*)
Object.PositionObject = PositionObject	Установить свойство (*)
PositionObject = Object.GetPositionObject()	Получить свойство (**)
Object.SetPositionObject(PositionObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PositionObject(&PositionObject)	Получить свойство
Object.put_PositionObject(PositionObject)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать объект, на котором будет располагаться плоскость объекта.

ToleranceArrowType – Тип стрелки ответвления

Интерфейс...

Тип данных: из перечисления ksToleranceArrowType

Синтаксис Automation:

ToleranceArrowType = Object.ToleranceArrowType(Index)	Получить свойство (*)
Object.ToleranceArrowType(Index) = ToleranceArrowType	Установить свойство (*)
ToleranceArrowType = Object.GetToleranceArrowType(Index)	Получить свойство (**)
Object.SetToleranceArrowType(Index, ToleranceArrowType)	Установить свойство (**)

Синтаксис COM:

Object.get_ToleranceArrowType(Index, &ToleranceArrowType)	Получить свойство
Object.put_ToleranceArrowType(Index, ToleranceArrowType)	Установить свойство

Входные параметры:

Index	- индекс ответвления.

ITolerance3D – методы

GetPosition – Получить точку, задающую положение объекта

Интерфейс...

Синтаксис Automation:

BOOL GetPosition (double * x,
double * y,
double * z);

Синтаксис COM:

HRESULT GetPosition(double * x,
double * y
double * z
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки, задающей положение объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetPosition – Установить точку, задающую положение объекта

Интерфейс...

Синтаксис Automation:

BOOL SetPosition (double x,
double y,
double z);

Синтаксис COM:

HRESULT SetPosition(double x,
double y
double z
BOOL * Result);

Входные параметры:

x, y, z	- координаты точки, задающей положение объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IBranchs3D

Справка системы КОМПАС...

KOMPAS.chm::/1024_113_2_2_Dobavlenie_i_udalenie_otv.htm

Интерфейс для работы с ответвлениями.

Иерархия:

IDispatch

IBranchs3D

Примечание:

Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface у интерфейсов IBaseLeader3D, ITolerance3D, IMarkLeader, IPositionLeader, IChangeLeader, IBrandLeader.

IBranchs3D – свойства

BranchCount – Число ответвлений

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchCount Object.BranchCount	=	Получить свойство (*)
BranchCount Object.GetBranchCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BranchCount(&BranchCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BranchEndPoints – Массив SAFEARRAY координат точек целеуказания

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

BranchEndPoints Object.BranchEndPoints	=	Получить свойство (*)
BranchEndPoints Object.GetBranchEndPoints()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BranchEndPoints(&BranchEndPoints)	Получить свойство

Свойство позволяет получить массив координат конечных точек ответвлений.

Примечание:

1. Свойство доступно только для чтения.
2. Координаты точек в массиве лежат в следующей последовательности: x0, y0, z0, x1, y1, z1,...xi, yi, zi.

BranchObject – Объект, на который указывает «ножка»

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BranchObject Object.BranchObject(Index)	=	Получить свойство (*)
BranchObject Object.GetBranchObject(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_BranchObject(Index, &BranchObject)	Получить свойство

Входные параметры

long Index	- номер ответвления.

Примечание:

Свойство доступно только для чтения.

BranchObjects – Массив SAFEARRAY объектов, на которые указывают «ножки»

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

BranchObjects Object.BranchObjects	=	Получить свойство (*)
BranchObjects Object.GetBranchObjects()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BranchObjects(&BranchObjects)	Получить свойство

Примечание:

1. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
2. Свойство доступно только для чтения.

BranchPoints – Массив SAFEARRAY координат точек ответвления

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

BranchPoints Object.BranchPoints(Index)	=	Получить свойство (*)
Object.BranchPoints(Index) BranchPoints	=	Установить свойство (*)
BranchPoints Object.GetBranchPoints(Index)	=	Получить свойство (**)
Object.SetBranchPoints(Index, BranchPoints)		Установить свойство (**)

Синтаксис COM:

Object.get_BranchPoints(Index, &BranchPoints)	Получить свойство
Object.put_BranchPoints(Index, BranchPoints)	Установить свойство

Входные параметры

long Index	- номер ответвления.

Свойство позволяет получить массив координат точек ответвления, заданного по номеру.

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, z0, x1, y1, z1,...xi, yi, zi.
2. При задании точек координаты будут спроецированы на плоскость объекта.

BranchPointsCount – Число точек в ответвлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

BranchPointsCount Object.BranchPointsCount	=	Получить свойство (*)
BranchPointsCount Object.GetBranchPointsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BranchPointsCount(&BranchPointsCount)	Получить свойство

Входные параметры

long Index	- номер ответвления.

Примечание:

Свойство доступно только для чтения.

IBranchs3D – методы

AddBranch – Добавить ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranch(VARIANT Points,
LPDISPATCH Object);

Синтаксис COM:

HRESULT AddBranch(VARIANT Points,
IModelObject * Object,
BOOL * Result);

Входные параметры:

Object	- объект, на который указывает ответвление,
Points	- массив SafeArray типа VT_ARRAY I VT_R8 координат точек.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Координаты точек в массиве лежат в следующей последовательности: x0, y0, z0, x1, y1, z1 ...xi, yi, zi.
2. При задании точек координаты будут спроецированы на плоскость объекта.

AddBranchByPoint – Добавить прямолинейное ответвление

Интерфейс...

Синтаксис Automation:

BOOL AddBranchByPoint(double X,
double Y,
double Z,
IModelObject * Object);

Синтаксис COM:

HRESULT AddBranchByPoint(double X,
double Y,
double Z,
IModelObject * Object,
BOOL * Result);

Входные параметры:

Object	- объект целеуказания,
x, y, z	- координаты точки целеуказания.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DeleteBranch – Удалить ответвление

Интерфейс...

Синтаксис Automation:

BOOL DeleteBranch(long Index);

Синтаксис COM:

HRESULT DeleteBranch(long Index, BOOL * Result);

Входные параметры:

Index	- индекс ответвления.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetBranchBeginPoint – Получить координаты начала полки или точки привязки

Интерфейс...

Синтаксис Automation:

BOOL GetBranchBeginPoint(double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetBranchBeginPoint(double * X,
double * Y,
double * Z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Метод позволяет получить координаты начала полки.

GetBranchEndPoint – Получить координаты конечной точки ответвления

Интерфейс...

Синтаксис Automation:

BOOL GetBranchBeginPoint(long Index,
double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetBranchBeginPoint(long Index,
double * X,
double * Y,
double * Z,
BOOL * Result);

Входные параметры:

Index	- индекс ответвления.

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Метод позволяет получить координаты конечной точки ответвления.

SetBranchBeginPoint – Установить координаты начала полки или точки привязки

Интерфейс...

Синтаксис Automation:

BOOL SetBranchBeginPoint(double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetBranchBeginPoint(double X,
double Y,

double Z,
BOOL * Result);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Метод позволяет установить координаты начала полки.

SetBranchEndPoint – Установить координаты конечной точки ответвления

Интерфейс...

Синтаксис Automation:

BOOL SetBranchEndPoint(long Index,
double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetBranchEndPoint(long Index,
double X,
double Y,
double Z,
BOOL * Result);

Входные параметры:

Index	- индекс ответвления,
x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха.
FALSE	- в случае неудачи.

Метод позволяет установить координаты конечной точки ответвления.

Интерфейс IUserDesignationCompObj

Составной объект для пользовательских объектов обозначений 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IUserDesignationCompObj

КОМПАС версия v18

IUserDesignationCompObj – методы

SetObjects – Задать объекты

Интерфейс...

Синтаксис Automation:

BOOL SetObjects(VARIANT Objects);

Синтаксис COM:

HRESULT SetObjects(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Objects	- массив SafeArray (VT_ARRAY | VT_DISPATCH) объектов.

Пересчет модели с учетом допусков

Интерфейс IToleranceRecalcsManager

Менеджер пересчета модели по допускам.

Иерархия:

IDispatch

IToleranceRecalcsManager

Является дополнительным к интерфейсу менеджера 3D документа (можно получить с помощью QueryInterface).

IToleranceRecalcsManager– свойства

CurrentRecalc – Интерфейс текущего пересчета

Интерфейс...

Тип данных: указатель на интерфейс IToleranceRecalc

Синтаксис Automation:

```
CurrentRecalc= Object.CurrentRecalc    Получить свойство (* )
CurrentRecalc=                         Получить свойство (**)
Object.GetCurrentRecalc()
```

Синтаксис COM:

```
CurrentRecalc          =          Получить свойство
Object.GetCurrentRecalc()
```

Примечание:

Свойство доступно только для чтения.

RecalcsCount – Количество пересчетов размеров

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
RecalcsCount= Object.RecalcsCount    Получить свойство (* )
RecalcsCount=                        Получить свойство (**)
Object.GetRecalcsCount()
```

Синтаксис COM:

```
RecalcsCount           =          Получить свойство
Object.GetRecalcsCount()
```

Примечание:

Свойство доступно только для чтения.

ToleranceMode – Включить\отключить режим пересчета по допуску

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ToleranceMode = Object.ToleranceMode;      Получить свойство (* )
Object.ToleranceMode = ToleranceMode;      Установить свойство (* )
ToleranceMode = Object.GetToleranceMode(); Получить свойство (**)
```

Object.SetToleranceMode(ToleranceMode); Установить свойство (*)

Синтаксис COM:

Object->get_ToleranceMode(&ToleranceMode); Получить свойство

Object->put_ToleranceMode(ToleranceMode); Установить свойство

IToleranceRecalcsManager– методы

AddRecalc – Создать пересчет размеров

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddRecalc();

Синтаксис COM:

HRESULT AddRecalc(IToleranceRecalc ** Result);

Возвращаемое значение:

- указатель на интерфейс пересчета размеров IToleranceRecalc.

AddRecalcCopy – Создать пересчет размеров по образцу

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddRecalcCopy(IToleranceRecalc * Source);

Синтаксис COM:

HRESULT AddRecalcCopy(IToleranceRecalc * Source, IToleranceRecalc ** Result);

Возвращаемое значение:

- указатель на интерфейс пересчета размеров IToleranceRecalc.

Входные параметры

Source	- указатель на копируемый пересчет размеров IToleranceRecalc.

GetRecalc – Получить пересчет размеров по имени или индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetRecalc(VARIANT Index);

Синтаксис COM:

HRESULT GetRecalc(VARIANT Index, IToleranceRecalc ** Result);

Возвращаемое значение:

- указатель на интерфейс пересчета размеров IToleranceRecalc.

Входные параметры

Index	- имя или индекс пересчета.

SaveRecalcModel – Сохранить пересчитанную копию модели

Интерфейс...

Синтаксис Automation:

BOOL SaveRecalcModel(BSTR FileName);

Синтаксис COM:

HRESULT SaveRecalcModel(BSTR FileName, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры

FileName	- имя файла, в который будет сохранена модель.

Интерфейс IToleranceRecalc

Интерфейс пересчета модели.

Иерархия:

IDispatch

IKompasAPIObject

IToleranceRecalc

Интерфейс можно получить у Менеджера пересчета модели с помощью методов IToleranceRecalcsManager::AddRecalc, IToleranceRecalcsManager::AddRecalcCopy, IToleranceRecalcsManager::GetRecalc и свойства IToleranceRecalcsManager::CurrentRecalc.

IToleranceRecalc– свойства

Coefficient – Коэффициент

Интерфейс...

Тип данных: double

Синтаксис Automation:

Coefficient = Object.Coefficient(Index);	Получить свойство (*)
Object.Coefficient(Index) = Coefficient;	Установить свойство (*)
Coefficient = Object.GetCoefficient(Index);	Получить свойство (**)
Object.SetCoefficient(Index, Coefficient);	Установить свойство (*)

Синтаксис COM:

Object.get_Coefficient(Index, &Coefficient);	Получить свойство
Object.put_Coefficient(Index, Coefficient);	Установить свойство

Входные параметры:

VARIANT	Index- имя или индекс пересчета.

Current – Текущий пересчет

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Current = Object.Current;	Получить свойство (*)
Object.Current = Current;	Установить свойство (*)
Current = Object.GetCurrent();	Получить свойство (**)
Object.SetCurrent(Current);	Установить свойство (*)

Синтаксис COM:

Object->get_Current(&Current);	Получить свойство
Object->put_Current(Current);	Установить свойство

Примечание:

Установить свойство можно только равным TRUE.

Id – Идентификатор

Интерфейс...

Тип данных: long

Синтаксис Automation:

Id = Object.Id	Получить свойство (*)
Id = Object.GetId()	Получить свойство (**)

Синтаксис COM:

Object.get_Id(&Id)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Item – Получить переменную или компонент

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Item = Object.Item(Index)	Получить свойство (*)
Item = Object.GetItem(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Item(Index, &Item)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ItemsCount – Количество объектов в пересчете

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
ItemsCount= Object.ItemsCount          Получить свойство (* )
ItemsCount= Object.GetItemsCount()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_ItemsCount( &ItemsCount )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Name – Имя

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = Object.Name;                    Получить свойство (* )
Object.Name = Name;                    Установить свойство (* )
Name = Object.GetName();               Получить свойство (**)
Object.SetName( Name );                Установить свойство (* )
```

Синтаксис COM:

```
Object->get_Name( &Name );             Получить свойство
Object->put_Name( Name );              Установить свойство
```

RecalcType – Способ пересчета

Интерфейс...

Тип данных: из перечисления ksToleranceRecalcsEnum

Синтаксис Automation:

```
RecalcType = Object.RecalcType( Index );       Получить свойство (* )
Object.RecalcType( Index ) = RecalcType;       Установить свойство (* )
RecalcType = Object.GetRecalcType( Index );    Получить свойство (**)
Object.SetRecalcType( Index, RecalcType );     Установить свойство (* )
```

Синтаксис COM:

```
Object.get_RecalcType( Index, &RecalcType );   Получить свойство
Object.put_RecalcType(  Index, RecalcType );   Установить свойство
```

Входные параметры:

VARIANT	Index - имя или индекс пересчета.

RecalcUserType – Способ пользовательского пересчета для вставки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
RecalcUserType = Object.RecalcUserType( Index );   Получить свойство (* )
Object.RecalcUserType( Index ) = RecalcUserType;   Установить свойство (* )
RecalcUserType = Object.GetRecalcUserType( Index );   Получить свойство (**)
Object.SetRecalcUserType( Index, RecalcUserType );   Установить свойство (* )
```

Синтаксис COM:

```
Object.get_RecalcUserType( Index, &RecalcUserType );   Получить свойство
Object.put_RecalcUserType( Index, RecalcUserType );   Установить свойство
```

Входные параметры:

VARIANT	Index - имя или индекс пересчета.

IToleranceRecalc– методы

AddAllVariables – Добавить все переменные

Интерфейс...

Синтаксис Automation:

BOOL AddAllVariables();

Синтаксис COM:

HRESULT AddAllVariables(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

AddItems – Добавить переменные, размер или компонент

Интерфейс...

Синтаксис Automation:

BOOL AddItems(IKompasAPIObject * Object);

Синтаксис COM:

HRESULT AddItems(IKompasAPIObject * Object, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Object - указатель на интерфейс переменной, размера или компонента.

DeleteItem – Удалить переменную или вариант пересчета компонента по имени, индексу или интерфейсу

Интерфейс...

Синтаксис Automation:

BOOL DeleteItem(VARIANT Index);

Синтаксис COM:

HRESULT DeleteItem(VARIANT Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

Index - имя или индекс пересчета.

Технические требования

Интерфейс ITechnicalDemand3D

Технические требования 3D.

Иерархия:

IKompasAPIObject

ITechnicalDemand3D

Описание:

Интерфейс позволяет задавать и получать параметры технических требований в 3D.

Примечание:

Интерфейс можно получить у интерфейса документа 3D с помощью свойства IKompasDocument3D::TechnicalDemand3D

ITechnicalDemand3D – свойства

Text – Текст

Интерфейс...

Тип данных: указатель на интерфейс текста IText

Синтаксис Automation:

Text = Object.Text	Получить свойство (*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

1. Свойство позволяет получать интерфейс текста технических требований.
2. Свойство доступно только для чтения.

ITechnicalDemand3D – методы

IsCreated – Признак наличия технических требований в документе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsCreated = Object.IsCreated	Получить свойство (*)

IsCreated = Получить свойство (**)
Object.GetIsCreated()

Синтаксис COM:

Object.get_IsCreated(Получить свойство
&IsCreated)

Примечание:

1. Свойство позволяет получать признак наличия технических требований в документе.
2. Свойство доступно только для чтения.

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет применить заданные параметры технических требований.

Неуказанная шероховатость

Интерфейс ISpecRough3D

Интерфейс неуказанной шероховатости 3D.

Иерархия:

IDispatch
IKompasAPIObject
ISpecRough3D

Интерфейс можно получить с помощью свойства IKompasDocument3D1::SpecRough.

ISpecRough3D – свойства

AddSign – Добавить знак в скобках

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AddSign = Object.AddSign;                    Получить свойство (* )
Object.AddSign = AddSign;                    Установить свойство (* )
AddSign = Object.GetAddSign();               Получить свойство (**)
Object.SetAddSign( AddSign );                Установить свойство (* )
```

Синтаксис COM:

```
Object.get_AddSign( &AddSign );              Получить свойство
Object.put_AddSign(  AddSign );              Установить свойство
```

IsCreated – Признак отображения в документе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsCreated = Object.IsCreated                 Получить свойство (* )
IsCreated = Object.GetIsCreated()            Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsCreated( &IsCreated )           Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Placement – Размещение

Интерфейс...

Тип данных: из перечисления ksSpecRoughPlacementEnum

Синтаксис Automation:

```
Placement = Object. Placement;               Получить свойство (* )
Object. Placement =  Placement;              Установить свойство (* )
Placement = Object.Get Placement();          Получить свойство (**)
Object.Set Placement(  Placement );          Установить свойство (* )
```

Синтаксис COM:

```
Object.get_ Placement( & Placement );        Получить свойство
Object.put_ Placement(   Placement );        Установить свойство
```

SignType – Тип значка

Интерфейс...

Тип данных: из перечисления ksRoughSignEnum

Синтаксис Automation:

```
 SignType = Object. SignType;                    Получить свойство (* )
Object. SignType =  SignType;                    Установить свойство (* )
 SignType = Object.Get SignType();               Получить свойство (**)
Object.Set SignType(  SignType );                Установить свойство (* )
```

Синтаксис COM:

```
Object.get_ SignType( & SignType );              Получить свойство
Object.put_ SignType(   SignType );              Установить свойство
```

Text – Текст

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
 Text = Object. Text;                            Получить свойство (* )
Object. Text =  Text;                            Установить свойство (* )
 Text = Object.Get Text();                       Получить свойство (**)
Object.Set Text(  Text );                        Установить свойство (* )
```

Синтаксис COM:

```
Object.get_ Text( & Text );                      Получить свойство
Object.put_ Text(   Text );                      Установить свойство
```

ISpecRough3D – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Update – Обновление данных

Интерфейс...

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Операции, массивы, эскизы, точки

Контейнер

Интерфейс IModelContainer

Интерфейс контейнера трехмерных объектов.

Иерархия:

IDispatch

IModelContainer

Описание.

Интерфейс позволяет работать с коллекциями 3D-объектов, входящих в состав 3D-объекта.

Примечание.

Интерфейс контейнера является дополнительным к интерфейсу IPart7. Данный интерфейс можно получить от IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);

IModelContainer – свойства

BilletsObsoletes – Коллекция деталей заготовок и зеркальных деталей

Интерфейс...

Тип данных: указатель на интерфейс коллекции деталей заготовок и зеркальных деталей IBilletsObsoletes

Синтаксис Automation:

```
BilletsObsoletes                =   Получить свойство (* )
Object.BilletsObsoletes;
BilletsObsoletes                =   Получить свойство (**)
Object.GetBilletsObsoletes();
```

Синтаксис COM:

```
Object.get_BilletsObsoletes(                    Получить свойство
&BilletsObsoletes);
```

Примечание:

Свойство доступно только для чтения.

BodyRepositions – Коллекция операций перепозиционирования тела, поверхности

Интерфейс...

Тип данных: Указатель на интерфейс IBodyRepositions.

Синтаксис Automation:

```
BodyRepositions                 =   Получить свойство (* )
Object.BodyRepositions;
BodyRepositions                 =   Получить свойство (**)
Object.GetBodyRepositions();
```

Синтаксис COM:

```
Object.get_BodyRepositions(                     Получить свойство
&BodyRepositions );
```

Примечание:

Свойство доступно только для чтения.

Booleans – Коллекция булевых операций

Интерфейс...

Тип данных: Указатель на интерфейс IBooleans.

Синтаксис Automation:

```
Booleans = Object.Booleans                    Получить свойство (* )
Booleans = Object.GetBooleans()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Booleans( &Booleans )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Chamfers – Коллекция операций Фаска

Интерфейс...

Тип данных: Указатель на интерфейс IChamfers.

Синтаксис Automation:

```
Chamfers = Object.Chamfers                    Получить свойство (* )
Chamfers = Object.GetChamfers()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Chamfers( &Chamfers )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

CollectionsGeometry – Коллекции геометрии

Интерфейс...

Тип данных: указатель на интерфейс ICollectionsGeometry.

Синтаксис Automation:

```
CollectionsGeometry                 =   Получить свойство (* )
Object.CollectionsGeometry;
CollectionsGeometry                 =   Получить свойство (**)
Object.GetCollectionsGeometry();
```

Синтаксис COM:

```
Object.get_CollectionsGeometry(                 Получить свойство
&CollectionsGeometry );
```

Примечание:

Свойство доступно только для чтения.

CopiesGeometry – Коллекция операций копирования геометрии

Интерфейс...

Тип данных: указатель на интерфейс ICopiesGeometry.

Синтаксис Automation:

```
CopiesGeometry                  =   Получить свойство (* )
Object.CopiesGeometry;
CopiesGeometry                  =   Получить свойство (**)
Object.GetCopiesGeometry();
```

Синтаксис COM:

```
Object.get_CopiesGeometry(                  Получить свойство
&CopiesGeometry );
```

Примечание:

Свойство доступно только для чтения.

Cuts – Коллекция операций Сечение

Интерфейс...

Тип данных: Указатель на интерфейс ICuts.

Синтаксис Automation:

```
Cuts = Object.Cuts                          Получить свойство (* )
Cuts = Object.GetCuts()                     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Cuts( &Cuts )                    Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Evolutions – Коллекция кинематических операций

Интерфейс...

Тип данных: Указатель на интерфейс IEvolutions.

Синтаксис Automation:

```
Evolutions = Object.Evolutions                    Получить свойство (* )
Evolutions = Object.GetEvolutions()               Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Evolutions( &Evolutions )              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Extrusions – Указатель на интерфейс коллекции операций выдавливания

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций выдавливания IExtrusions.

Синтаксис Automation:

```
Extrusions = iObject.Extrusions;          Получить свойство (* )
Extrusions = iObject.GetExtrusions();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Extrusions( &Extrusions      Получить свойство
);
```

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получать коллекции операций выдавливания.

FeaturePatterns – Указатель на интерфейс коллекции операций копирования

Интерфейс...

Тип данных: указатель на интерфейс IFeaturePatterns.

Синтаксис Automation:

```
FeaturePatterns                 =   Получить свойство (* )
Object.FeaturePatterns
FeaturePatterns                 =   Получить свойство (**)
Object.GetFeaturePatterns()
```

Синтаксис COM:

FeaturePatterns Object.GetFeaturePatterns()	=	Получить свойство

Свойство позволяет получить коллекцию всех операций копирования, входящих в состав данного объекта.

Примечание:

Свойство доступно только для чтения.

Fillets – Коллекция операций Скругление

Интерфейс...

Тип данных: Указатель на интерфейс IFillets.

Синтаксис Automation:

Fillets = Object.Fillets	Получить свойство (*)
Fillets = Object.GetFillets()	Получить свойство (**)

Синтаксис COM:

Object.get_Fillets(&Fillets)	Получить свойство

Примечание:

Свойство доступно только для чтения.

FullFillets – Коллекция операций полного скругления

Интерфейс...

Тип данных: Указатель на интерфейс IFullFillets

Синтаксис Automation:

FullFillets = Object.FullFillets	Получить свойство (*)
FullFillets = Object.GetFullFillets()	Получить свойство (**)

Синтаксис COM:

Object.get_FullFillets(&FullFillets)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия Компас v18.1

Holes3D – Коллекция отверстий 3D

Интерфейс...

Тип данных: Указатель на интерфейс IHoles3D.

Синтаксис Automation:

Holes3D = Object.Holes3D	Получить свойство (*)
Holes3D = Object.GetHoles3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Holes3D(&Holes3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Inclines – Коллекция операций Уклон

Интерфейс...

Тип данных: Указатель на интерфейс IInclines.

Синтаксис Automation:

Inclines = Object.Inclines	Получить свойство (*)
Inclines = Object.GetInclines()	Получить свойство (**)

Синтаксис COM:

Object.get_Inclines(&Inclines)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Lofts – Коллекция операций По сечениям

Интерфейс...

Тип данных: Указатель на интерфейс ILofts.

Синтаксис Automation:

Lofts = Object.Lofts	Получить свойство (*)
Lofts = Object.GetLofts()	Получить свойство (**)

Синтаксис COM:

Object.get_Lofts(&Lofts)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MacroObjects3D – Коллекция макроэлементов 3D

Интерфейс...

Тип данных: Указатель на интерфейс IMacroObjects3D.

Синтаксис Automation:

MacroObjects3D = Object.MacroObjects3D	Получить свойство (*)
MacroObjects3D = Object.GetMacroObjects3D()	Получить свойство (**)

Синтаксис COM:

Object.get_MacroObjects3D(&MacroObjects3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MoldCavities – Коллекция операций вычитания компонентов

Интерфейс...

Тип данных: Указатель на интерфейс IMoldCavities.

Синтаксис Automation:

MoldCavities = Object.MoldCavities	Получить свойство (*)
MoldCavities = Object.GetMoldCavities()	Получить свойство (**)

Синтаксис COM:

Object.get_MoldCavities(&MoldCavities)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Objects – Трехмерные объекты, входящие в состав данного объекта

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Objects = iObject.Objects(ObjType)		Получить свойство (*)
Objects	=	Получить свойство (**)
iObject.GetObjects(ObjType)		

Синтаксис COM:

iObject->get_Objects(&Objects)	Получить свойство

Входные параметры:

ObjType	- тип трехмерных объектов из перечисления KompasAPIObjectTypeEnum.

Примечание:

1. Свойство доступно только для чтения.
2. Коллекция объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH). Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Points3D – Указатель на интерфейс на интерфейс коллекции элементов Пространственная точка

Интерфейс...

Тип данных: указатель на интерфейс коллекции элементов Пространственная точка IPoints3D.

Синтаксис Automation:

Points3D = iObject.Points3D;	Получить свойство (*)
Points3D = iObject.GetPoints3D();	Получить свойство (**)

Синтаксис COM:

iObject->get_Points3D(&Points3D);	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получать коллекции элементов Пространственная точка.

Ribs – Коллекция операций Ребро жесткости

Интерфейс...

Тип данных: Указатель на интерфейс IRibs.

Синтаксис Automation:

Ribs = Object.Ribs	Получить свойство (*)
Ribs = Object.GetRibs()	Получить свойство (**)

Синтаксис COM:

Object.get_Ribs(&Ribs)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Rotateds – Указатель на интерфейс коллекции операций вращения

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций вращения IRotateds.

Синтаксис Automation:

Rotateds = Object.Rotateds	Получить свойство (*)
Rotateds = Object.GetRotateds()	Получить свойство (**)

Синтаксис COM:

Object.get_Rotateds(&Rotateds)	Получить свойство

Свойство позволяет получить коллекцию всех операций вращения, входящих в состав данного объекта.

Примечание:

Свойство доступно только для чтения.

Scalings3D – Коллекция операций масштабирования

Интерфейс...

Тип данных: указатель на интерфейс IScalings3D.

Синтаксис Automation:

Scalings3D = Object.Scalings3D	Получить свойство (*)
Scalings3D = Object.GetScalings3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Scalings3D(&Scalings3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Shells – Коллекция операций Оболочка

Интерфейс...

Тип данных: Указатель на интерфейс IShells.

Синтаксис Automation:

Shells = Object.Shells	Получить свойство (*)
Shells = Object.GetShells()	Получить свойство (**)

Синтаксис COM:

Object.get_Shells(&Shells)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Sketchs – Указатель на интерфейс коллекции объектов "Эскиз"

Интерфейс...

Тип данных: указатель на интерфейс коллекции элементов «Эскиз» ISketchs.

Синтаксис Automation:

Sketchs = iObject.Sketchs;	Получить свойство (*)
Sketchs = iObject.GetSketchs();	Получить свойство (**)

Синтаксис COM:

iObject->get_Sketchs(&Sketchs);	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Свойство позволяет получать коллекции элементов "Эскиз".

SurfaceThickenings – Указатель на интерфейс коллекции операций придания толщины

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций придания толщины ISurfaceThickenings.

Синтаксис Automation:

SurfaceThickenings = Object.SurfaceThickenings		Получить свойство (*)
SurfaceThickenings Object.GetSurfaceThickenings()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SurfaceThickenings(&SurfaceThickenings)	Получить свойство

Свойство позволяет получить коллекцию всех операций придания толщины, входящих в состав данного объекта.

Примечание:

Свойство доступно только для чтения.

UnionsComponents – Коллекция операций объединения компонентов

Интерфейс...

Тип данных: Указатель на интерфейс IUnionsComponents.

Синтаксис Automation:

UnionsComponents = Object.UnionsComponents		Получить свойство (*)
UnionsComponents Object.GetUnionsComponents()	=	Получить свойство (**)

Синтаксис COM:

Object.get_UnionsComponents(&UnionsComponents)	Получить свойство

Примечание:

Свойство доступно только для чтения.

UserObjects – Коллекции пользовательских объектов 3D

Интерфейс...

Тип данных: Указатель на интерфейс IUserObjects3D.

Синтаксис Automation:

UserObjects = Object.UserObjects	Получить свойство(*)
UserObjects = Object.GetUserObjects()	Получить свойство (**)

Синтаксис COM:

Object.get_UserObjects(&UserObjects)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IModelContainer – методы

AddObject – Создает новый элемент 3D модели

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddObject(ksObj3dTypeEnum ObjectType);

Синтаксис COM:

HRESULT AddObject(ksObj3dTypeEnum ObjectType, IModelObject * * Result);

Возвращаемое значение:

	- указатель на интерфейс IModelObject.

Входные параметры:

ObjectType	- тип объекта из перечисления ksObj3dTypeEnum.

Операции

Интерфейс IExtrusions

Справка системы КОМПАС...

KOMPAS.chm::/CM_BASE_EXTRUSION_SOLID.htm

Интерфейс коллекции операций выдавливания.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

IExtrusions

Примечание.

Получить интерфейс операций выдавливания можно, используя свойство контейнера трехмерных объектов IModelContainer::Extrusions.

IExtrusions – свойства

Extrusion – Операция выдавливания, заданная по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_BASE_EXTRUSION_SOLID.htm`

Тип данных: указатель на интерфейс IExtrusion

Синтаксис Automation:

Extrusion = iObject.Extrusion(index)	Получить свойство (*)
Extrusion = iObject.GetExtrusion(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Extrusion(&Extrusion)	index,	Получить свойство

Входные параметры:

VARIANT index	- индекс операции выдавливания.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

IExtrusions – методы

Add – Создать операцию выдавливание

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

LPDISPATH Add(ksObj3dTypeEnum ExtrusionType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum ExtrusionType, IExtrusion ** Result);

Возвращаемое значение:

Указатель на интерфейс операции выдавливание IExtrusion - в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый интерфейс операции выдавливания.
2. Допустимыми значениями ExtrusionType являются o3d_bossExtrusion, o3d_cutExtrusion.
3. После получения нового интерфейса нужно задать параметры операции выдавливание и вызвать метод IModelObject::Update.

Интерфейс IRotateds

Справка системы КОМПАС...

kompas.chm::/252_26_1_elem_vr.htm

Коллекция операций вращения.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IRotateds

Интерфейс позволяет получать и создавать операции вращения.

Примечание

Интерфейс можно получить, используя свойство контейнера трехмерных объектов IModelContainer::Rotateds (операции вращения и вырезания вращением) и свойство контейнера поверхностей ISurfaceContainer::RotatedSurfaces (поверхность вращения).

IRotateds – свойства

Rotated – Возвращает операцию вращения, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс IRotated

Синтаксис Automation:

Rotated = Object.Rotated(Index)	Получить свойство (*)
Rotated = Object.GetRotated(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Rotated(Index, &Rotated)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IRotateds – методы

Add – Создает новую операцию вращения и добавляет ее в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum RotatedType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum RotatedType, IRotated ** Result);

Возвращаемое значение:

Указатель на интерфейс операции вращения IRotatedIRotated.htm.

Входные параметры:

RotatedType - тип операции

Примечание:

Для создания операций вращения и вырезания вращением надо получать коллекцию у контейнера трехмерных объектов с помощью свойства IModelContainer::Rotateds, для создания поверхности вращения - у контейнера поверхностей с помощью свойства ISurfaceContainer::RotatedSurfaces.

Интерфейс IScalings3D

Справка системы КОМПАС...

`kompas.chm::/783_Glava88_Masshtabirovanie.htm`

Коллекция операций масштабирования.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IScalings3D

Интерфейс позволяет получать и создавать операции масштабирования.

Интерфейс можно получить у контейнера модельных объектов с помощью метода IModelContainer::Scalings3D

IScalings3D – свойства

Scaling3D – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IScaling3D

Синтаксис Automation:

Scaling3D = Object.Scaling3D(Index)	Получить свойство (*)
Scaling3D = Object.GetScaling3D(Index)	Получить свойство (**)

Синтаксис COM:

Scaling3D = Object.GetScaling3D(Index)	Получить свойство

Возвращает операцию масштабирования из коллекции по заданному индексу или имени.

Примечание:

Свойство доступно только для чтения.

IScalings3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IScaling3D ** Result);

Возвращаемое значение:

Указатель на интерфейс IScaling3D

Создает и добавляет новую операцию масштабирования в коллекцию.

Интерфейс ISurfaceThickenings

Справка системы КОМПАС...

`kompas.chm::/847_90_4_Pridan_tol_pov.htm`

Коллекция операций придания толщины поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

ISurfaceThickenings

ISurfaceThickenings – свойства

SurfaceThickening – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISurfaceThickening

Синтаксис Automation:

SurfaceThickening Object.SurfaceThickening(Index)	=	Получить свойство (*)
SurfaceThickening Object.GetSurfaceThickening(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SurfaceThickening(&SurfaceThickening)	Index,	Получить свойство

Свойство позволяет получить указатель на операцию придания толщины поверхности.

Примечание:

Свойство доступно только для чтения.

ISurfaceThickenings – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISurfaceThickening ** Result);

Возвращаемое значение:

Указатель на интерфейс операции придания толщины поверхности ISurfaceThickening.

Интерфейс IExtrusion

Справка системы КОМПАС...

KOMPAS.chm::/237_25_1_elem_vydavl.htm

Интерфейс операции выдавливания.

Иерархия:

IKompasAPIObject

IModelObject

IExtrusion

IChooseBodies7

IThinParameters

ICutExtrusion

IExtrusion1

Примечание.

1. Интерфейс можно получить с помощью метода коллекции операций выдавливания IExtrusions::Add или свойства IExtrusions::Extrusion.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) интерфейс позволяет получить следующие дополнительный интерфейсы:

- IChooseBodies7 - интерфейс для области применения тел в операции,
- IThinParameters - интерфейс параметров тонкой стенки,
- ICutExtrusion - интерфейс операции "Вырезать выдавливанием",
- IExtrusion1 - интерфейс параметров элемента выдавливания.

IExtrusion – свойства

Depth – Глубина выдавливания

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/696_84_1_2_Glubina_vydavlivanij.htm

Тип данных: double

Синтаксис Automation:

Depth = iObject.Depth(normal);	Получить свойство (*)
iObject.Depth(normal) = Depth;	Установить свойство (*)
Depth = iObject.GetDepth(normal);	Получить свойство (**)
iObject.SetDepth(normal, Depth);	Установить свойство (**)

Синтаксис COM:

iObject->get_Depth(normal, &Depth);	Получить свойство
iObject->put_Depth(normal, Depth);	Установить свойство

Входные параметры:

normal	TRUE - выдавливание в прямом направлении, FALSE - выдавливание в обратном направлении.

Примечание:

Свойство позволяет получить и установить глубину выдавливания.

DepthObject – Объект, задающий глубину выдавливания

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/243_25_5_1_Vyd_na_rasst.htm`

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

DepthObject = iObject.DepthObject(normal);	Получить свойство (*)
iObject.DepthObject(normal) = DepthObject;	Установить свойство (*)
DepthObject = iObject.GetDepthObject(normal);	Получить свойство (**)
iObject.SetDepthObject(normal, DepthObject);	Установить свойство (**)

Синтаксис COM:

iObject->get_DepthObject(normal, &DepthObject);	Получить свойство
iObject->put_DepthObject(normal, DepthObject);	Установить свойство

Входные параметры:

normal	TRUE - признак уклона наружу в прямом направлении, FALSE - признак уклона наружу в обратном направлении.

Примечание:

1. Свойство позволяет получать указатель на объект, задающий глубину.
2. Объект может быть задан, если направление на него совпадает с выбранным направлением (прямым или обратным) операции выдавливания IExtrusion::Direction.

Direction – Тип направления выдавливания

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/696_84_1_2_Glubina_vydavlivanij.htm

Тип данных: из перечисления DirectionTypeEnum

Синтаксис Automation:

Direction = iObject.Direction;	Получить свойство (*)
iObject.Direction = Direction;	Установить свойство (*)
Direction = iObject.GetDirection();	Получить свойство (**)
iObject.SetDirection(Direction);	Установить свойство (**)

Синтаксис COM:

iObject->get_Direction(&Direction);	Получить свойство
iObject->put_Direction(Direction);	Установить свойство

Примечание:

Свойство позволяет получить и установить тип направления выдавливания.

DraftOutward – Признак уклона внутрь

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/698_84_1_3_Ugol_uklona.htm#Raw48834

Тип данных: bool

Синтаксис Automation:

DraftOutward = iObject.DraftOutward(normal);	Получить свойство (*)
iObject.DraftOutward(normal) = DraftOutward;	Установить свойство (*)
DraftOutward = iObject.GetDraftOutward(normal);	Получить свойство (**)
iObject.SetDraftOutward(normal, DraftOutward);	Установить свойство (**)

Синтаксис COM:

iObject->get_DraftOutward(normal, &DraftOutward);	Получить свойство
iObject->put_DraftOutward(normal, DraftOutward);	Установить свойство

Входные параметры:

normal	TRUE - признак уклона наружу в прямом направлении, FALSE - признак уклона наружу в обратном направлении.

Примечание:

Свойство позволяет получить и установить признак уклона внутрь.

DraftValue – Угол наклона

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/698_84_1_3_Ugol_uklona.htm#Raw48834`

Тип данных: double

Синтаксис Automation:

DraftValue = iObject.DraftValue(normal);	Получить свойство (*)
iObject.DraftValue(normal) = DraftValue;	Установить свойство (*)

DraftValue = iObject.GetDraftValue(normal);	Получить свойство (**)
iObject.SetDraftValue(normal, DraftValue);	Установить свойство (**)

Синтаксис COM:

iObject->get_DraftValue(normal, &DraftValue);	Получить свойство
iObject->put_DraftValue(normal, DraftValue);	Установить свойство

Входные параметры:

normal	TRUE -угол наклона в прямом направлении, FALSE - угол наклона в обратном направлении.

Примечание:

Свойство позволяет получить и установить угол наклона.

ExtrusionType – Тип выдавливания

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/696_84_1_2_Glubina_vydavlivanij.htm`

Тип данных: из перечисления EndTypeEnum

Синтаксис Automation:

ExtrusionType = iObject.ExtrusionType(normal);	Получить свойство (*)
iObject.ExtrusionType(normal) = ExtrusionType;	Установить свойство (*)
ExtrusionType = iObject.GetExtrusionType(normal);	Получить свойство (**)
iObject.SetExtrusionType(normal, ExtrusionType);	Установить свойство (**)

Синтаксис COM:

iObject->get_ExtrusionType(&ExtrusionType);	Получить свойство
iObject->put_ExtrusionType(ExtrusionType);	Установить свойство

Входные параметры:

normal	TRUE - выдавливание в прямом направлении, FALSE - выдавливание в обратном направлении.

Примечание:

Свойство позволяет получить и установить тип выдавливания.

Sketch – Эскиз

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_BASE_EXTRUSION_SOLID.htm`

Тип данных: указатель на интерфейс эскиза ISketch

Синтаксис Automation:

Sketch = iObject.Sketch;	Получить свойство (*)
iObject.Sketch = Sketch;	Установить свойство (*)
Sketch = iObject.GetSketch();	Получить свойство (**)
iObject.SetSketch(Sketch);	Установить свойство (**)

Синтаксис COM:

iObject->get_Sketch(&Sketch);	Получить свойство
iObject->put_Sketch(Sketch);	Установить свойство

Примечание:

Свойство позволяет установить новый эскиз для операции и получить указатель на эскиз.

IExtrusion – методы

GetSideParameters – Получить параметры выдавливания в одном направлении

Интерфейс...

Синтаксис Automation:

```
BOOL GetSideParameters( BOOL normal,
End_Type * ExtrusionType,
double * Depth,
double * DraftValue,
BOOL * DraftOutward,
IModelObject * DepthObject );
```

Синтаксис COM:

```
HRESULT GetSideParameters( BOOL normal,
End_Type * ExtrusionType,
double * Depth,
double * DraftValue,
BOOL * DraftOutward,
IModelObject * DepthObject );
```

Входные параметры:

normal	направление, TRUE - в прямом направлении, FALSE - в обратном направлении.

Выходные параметры:

ExtrusionType	- тип выдавливания,
Depth	- глубина выдавливания,
DraftValue	- угол наклона,
DraftOutward	- признак уклона внутрь,
DepthObject	- указатель на объект, задающий глубину выдавливания.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет получать параметры операции выдавливание.
2. Параметр Depth позволяет задать на сколько выдавить за объект, задающий глубину выдавливания, или какое расстояние оставить до объекта, задающего глубину выдавлива-

ния - в зависимости от типа построения (для типов построения "до поверхности" и "до вершины" см. перечисление ksEndTypeEnum).

SetSideParameters – Установить параметры выдавливания в одном направлении

Интерфейс...

Синтаксис Automation:

BOOL SetSideParameters(BOOL normal,
End_Type * ExtrusionType,
double * Depth,
double * DraftValue,
BOOL * DraftOutward,
IModelObject * DepthObject);

Синтаксис COM:

HRESULT SetSideParameters(BOOL normal,
End_Type * ExtrusionType,
double * Depth,
double * DraftValue,
BOOL * DraftOutward,
IModelObject * DepthObject);

Входные параметры:

normal	направление, TRUE - в прямом направлении, FALSE - в обратном направлении,
ExtrusionType	- тип выдавливания,
Depth	- глубина выдавливания,
DraftValue	- угол наклона,
DraftOutward	- признак уклона внутрь,
DepthObject	- указатель на объект, задающий глубину выдавливания.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет получать параметры операции выдавливание.
2. Параметр Depth в позволяет задать на сколько выдавить за объект, задающий глубину выдавливания, или какое расстояние оставить до объекта, задающего глубину выдавливания - в зависимости от типа построения (для типов построения "до поверхности" и "до вершины" см. перечисление ksEndTypeEnum).

Интерфейс ICutExtrusion

Справка системы КОМПАС...

kompas.chm::/CM_CUT_EXTRUSION_SOLID.htm

Интерфейс операции "Вырезать выдавливанием".

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IExtrusion
ICutExtrusion

Примечание.

Интерфейс можно получить с помощью метода коллекции операций выдавливания IExtrusions::Add(Obj3dType ExtrusionType), передав тип 3D объекта - операция "вырезать выдавливанием" (o3d_cutExtrusion) или свойства IExtrusions::Extrusion.

ICutExtrusion – свойства

Cut – Признак результата операции (результирующего тела)

Интерфейс...

Тип данных: bool

Синтаксис Automation:

Cut = iObject.Cut;	Получить свойство(*)
iObject.Cut = Cut;	Установить свойство (*)
Cut = iObject.GetCut();	Получить свойство (**)
iObject.SetCut(Cut);	Установить свойство (**)

Синтаксис COM:

iObject->get_Cut(&Cut);	Получить свойство
iObject->put_Cut(Cut);	Установить свойство

Примечание:

Свойство позволяет установить и получить результат (результирующее тело) операции "Вырезать выдавливанием".

Интерфейс IExtrusionSurface

Справка системы КОМПАС...

kompas.chm::/885_103_2_Poverkhnostq_vydavliv.htm

Интерфейс параметров поверхности выдавливания

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IExtrusion

IExtrusionSurface

Интерфейс позволяет создавать и редактировать поверхности выдавливания.

Примечание:

Интерфейс можно получить с помощью свойства IExtrusions::Extrusion и метода IExtrusions::Add с типом o3d_ExtrusionSurface.

IExtrusionSurface – свойства

ClosedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedShell = Object.ClosedShell	Получить свойство(*)
Object.ClosedShell = ClosedShell	Установить свойство (*)
ClosedShell = Object.GetClosedShell()	Получить свойство (**)
Object.SetClosedShell(ClosedShell)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedShell(&ClosedShell)	Получить свойство
Object.put_ClosedShell(ClosedShell)	Установить свойство

Интерфейс IRotated

Справка системы КОМПАС...

`kompas.chm::/252_26_1_elem_vr.htm`

Интерфейс параметров операции вращения.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRotated

Интерфейс позволяет создавать и редактировать операции вращения. Имеет дополнительный интерфейс IThinParameters, который можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Интерфейс можно получить с помощью свойства IRotateds::Rotated и метода IRotateds::Add с типом o3d_bossRotated.

IRotated – свойства

Angle – Угол вращения

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle(Normal)	Получить свойство (*)
Object.Angle(Normal) = Angle	Установить свойство (*)
Angle = Object.GetAngle(Normal)	Получить свойство (**)
Object.SetAngle(Normal, Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(Normal, &Angle)	Получить свойство
Object.put_Angle(Normal, Angle)	Установить свойство

Входные параметры:

BOOL Normal	- направление.

Примечания:

1. Для вращения в двух направлениях угол требуется задать дважды — для прямого и обратного направления.
2. Если используется тип направления Средняя плоскость, то угол задается один раз. При этом он воспринимается системой как общий угол (т.е. в каждую сторону откладывается его половина).

AngleObject – Объект, задающий угол вращения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

AngleObject = Object.AngleObject(Normal)	Получить свойство (*)
Object.AngleObject(Normal) = AngleObject	Установить свойство (*)

AngleObject = Object.GetAngleObject(Normal)	Получить свойство (**)
Object.SetAngleObject(Normal, AngleObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AngleObject(Normal, &AngleObject)	Получить свойство
Object.put_AngleObject(Normal, AngleObject)	Установить свойство

Входные параметры

BOOL Normal	- направление.

Axis – Ось операции вращения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Axis = Object.Axis	Получить свойство (*)
Object.Axis = Axis	Установить свойство (**)
Object.Axis = Axis	Получить свойство (**)
Object.SetAxis(Axis)	Установить свойство (**)

Синтаксис COM:

Object.get_Axis(&Axis)	Получить свойство
Object.put_Axis(Axis)	Установить свойство

Свойство позволяет устанавливать и получать объект, задающий ось вращения. Ось может являться кривой 3D, ребром или браться из эскиза.

Примечание:

Объект, задающий ось, должен быть прямолинейным. Если используется ось из эскиза, то она должна быть изображена отрезком со стилем линии *Осевая* или объектом *Осевая линия*.

CutOffByPoint – Флаг «отсекать» для способа определения угла До точки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CutOffByPoint(Normal) = Object.CutOffByPoint	Получить свойство (*)
Object.CutOffByPoint(Normal) = CutOffByPoint	Установить свойство (*)
CutOffByPoint = Object.GetCutOffByPoint(Normal)	Получить свойство (**)

Object.SetCutOffByPoint(Normal, CutOffByPoint)	Установить свойство (**)

Синтаксис COM:

Object.get_CutOffByPoint(Normal, &CutOffByPoint)	Получить свойство
Object.put_CutOffByPoint(Normal, CutOffByPoint)	Установить свойство

Direction – Направление вращения

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (**)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Свойство позволяет устанавливать и получать направление, в котором производится вращение.

Profile – Сечение для операции вращения

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Profile = Object.Profile	Получить свойство (*)
Object.Profile = Profile	Установить свойство (**)
Profile = Object.GetProfile()	Получить свойство (**)
Object.SetProfile(Profile)	Установить свойство (**)

Синтаксис COM:

Object.get_Profile(&Profile)	Получить свойство
Object.put_Profile(Profile)	Установить свойство

Свойство позволяет устанавливать и получать объект, определяющий форму сечения операции. Это может быть эскиз, грань, кривая 3D.

Примечание:

Если в эскизе, задающем сечение, присутствует ось, и ось не задана свойством IRotated::Axis, то для построения операции будет использоваться ось из эскиза.

RotatedType – Способ построения угла вращения

Интерфейс...

Тип данных: из перечисления ksRotatedTypeEnum

Синтаксис Automation:

RotatedType = Object.RotatedType(Normal)	Получить свойство (*)
Object.RotatedType(Normal) = RotatedType	Установить свойство (**)
RotatedType = Object.GetRotatedType(Normal)	Получить свойство (**)
Object.SetRotatedType(Normal, RotatedType)	Установить свойство (**)

Синтаксис COM:

Object.get_RotatedType(Normal, &RotatedType)	Получить свойство
Object.put_RotatedType(Normal, RotatedType)	Установить свойство

Входные параметры:

BOOL Normal	- направление.

ToroidShapeType – Признак тороида

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ToroidShapeType = Object.ToroidShapeType	Получить свойство (*)
Object.ToroidShapeType = ToroidShapeType	Установить свойство (**)
ToroidShapeType = Object.GetToroidShapeType()	Получить свойство (**)
Object.SetToroidShapeType(ToroidShapeType)	Установить свойство (**)

Синтаксис COM:

Object.get_ToroidShapeType(&ToroidShapeType)	Получить свойство
Object.put_ToroidShapeType(ToroidShapeType)	Установить свойство

Свойство позволяет устанавливать и получать признак тороида.

Интерфейс ICutRotated

Справка системы КОМПАС...

kompas.chm::/252_26_1_elem_vr.htm

Интерфейс параметров операции «Вырезать вращением»

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRotated

ICutRotated

Интерфейс позволяет создавать и редактировать операции вырезания вращением. Имеет дополнительный интерфейс IThinParameters, который можно получить с помощью методаIUnknown::QueryInterface.

Примечание:

Интерфейс можно получить с помощью свойства IRotateds::Rotated и метода IRotateds::Add с типом o3d_cutRotated.

ICutRotated – свойства

Cut – Признак результата операции (результирующего тела)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Cut = Object.Cut	Получить свойство (*)
Object.Cut = Cut	Установить свойство (*)
Cut = Object.GetCut()	Получить свойство (**)
Object.SetCut(Cut)	Установить свойство (**)

Синтаксис COM:

Object.get_Cut(&Cut)	Получить свойство
Object.put_Cut(Cut)	Установить свойство

Значения свойства:

TRUE	- вычитание элементов,
FALSE	- пересечение элементов.

Интерфейс IRotatedSurface

Справка системы КОМПАС...

kompas.chm::/884_Glava103_Poverkhnosti.htm

Интерфейс параметров поверхности вращения

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRotated

IRotatedSurface

Интерфейс позволяет создавать и редактировать поверхности вращения.

Примечание:

Интерфейс можно получить с помощью свойства IRotateds::Rotated и метода IRotateds::Add с типом o3d_RotatedSurface.

IRotatedSurface – свойства

ClosedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedShell = Object.ClosedShell	Получить свойство (*)
Object.ClosedShell = ClosedShell	Установить свойство (*)
ClosedShell = Object.GetClosedShell()	Получить свойство (**)
Object.SetClosedShell(ClosedShell)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedShell(&ClosedShell)	Получить свойство
Object.put_ClosedShell(ClosedShell)	Установить свойство

Интерфейс IRotated1

Интерфейс операции вращения.

Справка системы КОМПАС...

kompas.chm::/252_26_1_elem_vr.htm

Иерархия:

IDispatch

IRotated1

Интерфейс является дополнительным к интерфейсу IRotated. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

КОМПАС версия v18

IRotated1 – Свойства

OperationResult – Результат операции

Интерфейс...

Тип данных: Значение из перечисления ksOperationResultEnum.

Синтаксис Automation:

OperationResult = Object.OperationResult	Получить свойство (*)
Object.OperationResult = OperationResult	Установить свойство (**)
OperationResult = Object.GetOperationResult()	Получить свойство (**)
Object.SetOperationResult(OperationResult)	Установить свойство (**)

Синтаксис COM:

Object.get_OperationResult(&OperationResult)	Получить свойство
Object.put_OperationResult(OperationResult)	Установить свойство

Интерфейс IShellSurface

Интерфейс операции Поверхность (поверхность выдавливания, вращения, по сечениям, кинематическая).

Иерархия:

IDispatch

IShellSurface

Интерфейс является дополнительным для поверхностей:

- выдавливания IExtrusion,
- вращения IRotated,
- по сечениям ILoft,
- кинематических IEvolution.

Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

КОМПАС версия v18

IShellSurface – свойства

ClosedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedShell = Object.ClosedShell	Получить свойство (*)
Object.ClosedShell = ClosedShell	Установить свойство (**)
ClosedShell = Object.GetClosedShell()	Получить свойство (**)
Object.SetClosedShell(ClosedShell)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedShell(&ClosedShell)	Получить свойство
Object.put_ClosedShell(ClosedShell)	Установить свойство

Интерфейс IScaling3D

Справка системы КОМПАС...

kompas.chm::/783_Glava88_Masshtabirovanie.htm

Интерфейс параметров операции масштабирования.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IScaling3D

Интерфейс позволяет изменять размеры тела или поверхности в трех направлениях согласно заданному коэффициенту.

Интерфейс можно получить у коллекции операций масштабирования с помощью свойства IScalings3D::Scaling3D и метода IScalings3D::Add.

IScaling3D – свойства

BasePoint – Центр масштабирования тела или поверхности

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BasePoint = Object.BasePoint	Получить свойство (*)
Object.BasePoint = BasePoint	Установить свойство (*)
BasePoint = Object.GetBasePoint()	Получить свойство (**)
Object.SetBasePoint(BasePoint)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePoint(&BasePoint)	Получить свойство
Object.put_BasePoint(BasePoint)	Установить свойство

Свойство позволяет устанавливать и получать центр масштабирования - вершину или точку в пространстве, выбираемую за неподвижную.

Scale – Масштаб (коэффициент масштабирования тела или поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

Scale = Object.Scale	Получить свойство (*)
Object.Scale = Scale	Установить свойство (*)
Scale = Object.GetScale()	Получить свойство (**)
Object.SetScale(Scale)	Установить свойство (**)

Синтаксис COM:

Object.get_Scale(&Scale)	Получить свойство
Object.put_Scale(Scale)	Установить свойство

Свойство позволяет устанавливать и получать коэффициент масштабирования.

Shell – Тело или поверхность для масштабирования

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Shell = Object.Shell	Получить свойство (*)
Object.Shell = Shell	Установить свойство (*)

Shell = Object.GetShell()	Получить свойство (**)
Object.SetShell(Shell)	Установить свойство (**)

Синтаксис COM:

Object.get_Shell(&Shell)	Получить свойство
Object.put_Shell(Shell)	Установить свойство

Свойство позволяет устанавливать и получать объект для масштабирования.

Интерфейс ISurfaceThickening

Справка системы КОМПАС...

kompas.chm::/847_90_4_Pridan_tol_pov.htm

Интерфейс операции придания толщины поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISurfaceThickening

IThinParameters

Интерфейс позволяет придать толщину граням тела или поверхности, т.е. добавить слой материала на эти грани. Сформированный слой материала представляет собой заполненное материалом пространство между исходной и эквидистантной поверхностями или двумя эквидистантными поверхностями.

Имеет дополнительный интерфейс IThinParameters.

ISurfaceThickening – свойства

Faces – Список граней

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces(Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces) Получить свойство
Object.put_Faces(Faces) Установить свойство

Свойство позволяет задать грань (тип VARIANT'a - VT_DISPATCH) или массив граней (тип VARIANT'a - VT_ARRAY | VT_DISPATCH) для выполнения операции.

Интерфейс IThread

Интерфейс условного обозначения резьбы.

Справка системы КОМПАС...

KOMPAS.chm::/CM_SYMBOLS_THREAD.htm

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IThread

IThreadsParameters

Интерфейс можно получить с помощью методов:

IThreads::Thread

IThreads::Add

Интерфейс имеет дополнительный интерфейс IThreadsParameters.

IThread – свойства

AutoDiameter – Автоопределение диаметра

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoDiameter = Object.AutoDiameter Получить свойство (*)
Object.AutoDiameter = AutoDiameter Установить свойство (*)
AutoDiameter = Получить свойство (**)
Object.GetAutoDiameter()
Object.SetAutoDiameter(AutoDiameter) Установить свойство (**)

Синтаксис COM:

Object.get_AutoDiameter(&AutoDiameter) Получить свойство
Object.put_AutoDiameter(AutoDiameter) Установить свойство

AutoLenght – Автоматическое определение длины резьбы (TRUE – длина определяется автоматически – до конечной границы, FALSE – длина задана явно)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AutoLenght = Object.AutoLenght          Получить свойство (* )
Object.AutoLenght = AutoLenght          Установить свойство (* )
AutoLenght = Object.GetAutoLenght()     Получить свойство (**)
 Object.SetAutoLenght( AutoLenght )     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AutoLenght( &AutoLenght )    Получить свойство
Object.put_AutoLenght(  AutoLenght )    Установить свойство
```

BaseObject – Ребро или грань, по которой строится резьба

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
 BaseObject = Object.BaseObject         Получить свойство (* )
Object.BaseObject = BaseObject          Установить свойство (* )
BaseObject = Object.GetBaseObject()     Получить свойство (**)
Object.SetBaseObject( BaseObject )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BaseObject( &BaseObject )    Получить свойство
Object.put_BaseObject(  BaseObject )    Установить свойство
```

BaseObjectAdjustment – Подгонка диаметра базового объекта под диаметр резьбы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
BaseObjectAdjustment                =   Получить свойство (* )
Object.BaseObjectAdjustment
```

Object.BaseObjectAdjustment BaseObjectAdjustment	=	Установить свойство (*)
BaseObjectAdjustment Object.GetBaseObjectAdjustment()	=	Получить свойство (**)
Object.SetBaseObjectAdjustment(BaseObjectAdjustment)		Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjectAdjustment(&BaseObjectAdjustment)	Получить свойство
Object.put_BaseObjectAdjustment(BaseObjectAdjustment)	Установить свойство

BaseObjectAdjustmentOffset1 – Дополнительная подгонка для сохранения глубины зенковки/скругления/фаски при подгонке диаметра базового объекта под диаметр резьбы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BaseObjectAdjustmentOffset1 Object.BaseObjectAdjustmentOffset1	=	Получить свойство (*)
Object.BaseObjectAdjustmentOffset1 BaseObjectAdjustmentOffset1	=	Установить свойство (*)
BaseObjectAdjustmentOffset1 Object.GetBaseObjectAdjustmentOffset1( )	=	Получить свойство (**)
Object.SetBaseObjectAdjustmentOffset1(BaseObjectAdjustmentOffset1)		Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjectAdjustmentOffset1(&BaseObjectAdjustmentOffset1)	Получить свойство
Object.put_BaseObjectAdjustmentOffset1(BaseObjectAdjustmentOffset1)	Установить свойство

BaseObjectAdjustmentOffset2 – Дополнительная подгонка для сохранения глубины неплоского торца отверстия при подгонке диаметра базового объекта под диаметр резьбы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BaseObjectAdjustmentOffset2 = Object.BaseObjectAdjustmentOffset2	=	Получить свойство (*)
Object.BaseObjectAdjustmentOffset2 = BaseObjectAdjustmentOffset2	=	Установить свойство (*)
BaseObjectAdjustmentOffset2 = Object.GetBaseObjectAdjustmentOffset2( )	=	Получить свойство (**)
Object.SetBaseObjectAdjustmentOffset2(BaseObjectAdjustmentOffset2)		Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjectAdjustmentOffset2(&BaseObjectAdjustmentOffset2)	Получить свойство
Object.put_BaseObjectAdjustmentOffset2(BaseObjectAdjustmentOffset2)	Установить свойство

Direction – Направление резьбы (TRUE – прямое, FALSE – обратное)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

InitialBorder – Начальная граница

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

InitialBorder = Object.InitialBorder	Получить свойство (*)
Object.InitialBorder = InitialBorder	Установить свойство (*)
InitialBorder = Object.GetInitialBorder()	Получить свойство (**)
Object.SetInitialBorder(InitialBorder)	Установить свойство (**)

Синтаксис COM:

Object.get_InitialBorder(&InitialBorder)	Получить свойство
Object.put_InitialBorder(InitialBorder)	Установить свойство

FinalBorder – Конечная граница

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

FinalBorder = Object.FinalBorder	Получить свойство (*)
Object.FinalBorder = FinalBorder	Установить свойство (*)
FinalBorder = Object.GetFinalBorder()	Получить свойство (**)
Object.SetFinalBorder(FinalBorder)	Установить свойство (**)

Синтаксис COM:

Object.get_FinalBorder(&FinalBorder)	Получить свойство
Object.put_FinalBorder(FinalBorder)	Установить свойство

LeftThread – Левосторонняя резьба

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

LeftThread = Object.LeftThread	Получить свойство (*)
Object.LeftThread = LeftThread	Установить свойство (*)
LeftThread = Object.GetLeftThread()	Получить свойство (**)
Object.SetLeftThread(LeftThread)	Установить свойство (**)

Синтаксис COM:

Object.get_LeftThread(&LeftThread)	Получить свойство
Object.put_LeftThread(LeftThread)	Установить свойство

Примечание

Включение признака левой резьбы влияет на формирование обозначения резьбы. В обозначение будут добавлены буквы "LH", например, "МК 20*1,5 LH".

Lenght – Длина резьбы (для варианта явного задания длины)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Lenght = Object.Lenght	Получить свойство (*)
Object.Lenght = Lenght	Установить свойство (*)
Lenght = Object.GetLenght()	Получить свойство (**)
Object.SetLenght(Lenght)	Установить свойство (**)

Синтаксис COM:

Object.get_Lenght(&Lenght)	Получить свойство
Object.put_Lenght(Lenght)	Установить свойство

Интерфейс IThreadPattern

Интерфейс параметров стандартной резьбы.

Иерархия:

IDispatch

IKompasAPIObject

IThreadPattern

Интерфейс можно получить методом ISystemSettings::ThreadPattern

IThreadPattern – свойства

ConeAngle – cone – Угол для конической резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConeAngle = Object.ConeAngle	Получить свойство (*)
ConeAngle = Object.GetConeAngle()	Получить свойство (**)

Синтаксис COM:

Object.get_ConeAngle(&ConeAngle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ConeL1 – L1 Смещение основной плоскости от торца наружной резьбы (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConeL1 = Object.ConeL1(Index)	Получить свойство (*)
ConeL1 = Object.GetConeL1(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_ConeL1(Index, &ConeL1)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ConeL2 – L2 Смещение основной плоскости от торца внутренней резьбы (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConeL2 = Object.ConeL2(Index)	Получить свойство (*)
ConeL2 = Object.GetConeL2(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_ConeL1(Index, &ConeL1)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Designation – Обозначение резьбы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Designation = Object.Designation(Index)	Получить свойство (*)

Designation= Object.GetDesignation(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Designation(Index, &Designation)	Получить свойство

Примечание:

Свойство доступно только для чтения.

HoleKoeff – koeff – Коэффициент резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

HoleKoeff = Object.HoleKoeff	Получить свойство (*)
HoleKoeff= Object.GetHoleKoeff()	Получить свойство (**)

Синтаксис COM:

Object.get_HoleKoeff(&HoleKoeff)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Lenght – L – Рабочая длина резьбы (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Lenght = Object.Lenght(Index)	Получить свойство (*)
Lenght= Object.GetLenght(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Lenght(Index, &Lenght)	Получить свойство

Примечание:

Свойство доступно только для чтения.

NominalDiameter – D – Номинальный диаметр резьбы (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

NominalDiameter = Object.NominalDiameter(Index)	=	Получить свойство (*)
NominalDiameter= Object.GetNominalDiameter(Index)		Получить свойство (**)

Синтаксис COM:

Object.get_NominalDiameter(&NominalDiameter)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

P – P – Шаг резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

P = Object.P(Index)	Получить свойство (*)
P= Object.GetP(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_P(Index, &P)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PatternsCount – Количество записей в таблице

Интерфейс...

Тип данных: long

Синтаксис Automation:

PatternsCount = Object.PatternsCount	Получить свойство (*)
PatternsCount= Object.PatternsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_PatternsCount(&PatternsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Standart – Обозначение стандарта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Standart = Object.Standart	Получить свойство (*)
Standart= Object.GetStandart()	Получить свойство (**)

Синтаксис COM:

Object.get_Standart(&Standart)	Получить свойство

Примечание:

Свойство доступно только для чтения.

TableName – Имя таблицы данных

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

TableName = Object.TableName	Получить свойство (*)
TableName= Object.TableName()	Получить свойство (**)

Синтаксис COM:

Object.get_TableName(&TableName)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IThreadPattern – методы

GetParameters – Параметры резьбы для для заданного диаметра и шага

Интерфейс...

Синтаксис Automation:

BSTR GetParameters(long Index, double * NominalDiameter, double * P, double * L, double * L1, double * L2);

Синтаксис COM:

HRESULT GetParameters(long Index, double * NominalDiameter, double * P, double * L, double * L1, double * L2, BSTR * Result);

Возвращаемое значение:

- обозначение резьбы.

Входные параметры:

Index - индекс списка значений в базе стандартов.

Выходные параметры:

NominalDiameter - номинальный диаметр,
P - шаг резьбы,
L - рабочая длина резьбы,
L1 - длина смещения основной плоскости для внешней резьбы,
L2 - длина смещения основной плоскости для внутренней резьбы.

SelectDiameters – Список номинальных диаметров резьбы (мм)

Интерфейс...

Синтаксис Automation:

VARIANT SelectDiameters();

Синтаксис COM:

HRESULT SelectDiameters(VARIANT * Result);

Возвращаемое значение:

Массив SafeAttray типа (VT_ARRAY | VT_R8)
со списком значений диаметров для текущего стандарта.

SelectP – Список шагов резьбы для данного диаметра

Интерфейс...

Синтаксис Automation:

VARIANT SelectP(double NominalDiameter);

Синтаксис COM:

HRESULT SelectP(double NominalDiameter, VARIANT * Result);

Возвращаемое значение:

Массив SafeAttray типа (VT_ARRAY | VT_R8)
со списком значений шагов для заданного диаметра и текущего стандарта.

Входные параметры:

NominalDiameter - номинальный диаметр резьбы.

SelectParameters – Параметры резьбы для для заданного диаметра и шага

Интерфейс...

Синтаксис Automation:

BSTR SelectParameters(double NominalDiameter, double P, double * L, double * L1, double * L2);

Синтаксис COM:

HRESULT SelectParameters(double NominalDiameter, double P, double * L, double * L1, double * L2, BSTR * Result);

Возвращаемое значение:

Обозначение резьбы.

Входные параметры:

NominalDiameter - номинальный диаметр,
P - шаг резьбы.

Выходные параметры:

L - рабочая длина резьбы,
L1 - длина смещения основной плоскости для внешней резьбы,
L2 - длина смещения основной плоскости для внутренней резьбы.

Интерфейс IThreads

Интерфейс коллекции условных обозначений резьбы.

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
IThreads

Интерфейс можно получить с помощью метода:

Symbols3DContainer::Threads

IThreads – свойства

Thread – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IThread

Синтаксис Automation:

Thread = Object.Thread(Index)	Получить свойство (*)
Thread = Object.GetThread(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Thread(&Thread)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IThreads – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IThread * * Result);

Возвращаемое значение:

-указатель на интерфейс условного обозначения резьбы.

Интерфейс IThreadsParameters

Интерфейс параметров резьбы.

Иерархия:

IDispatch

IThreadsParameters

Интерфейс является дополнительным для интерфейса IThread.

IThreadsParameters – свойства

ConicalThreadAngle – Угол уклона конической резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConicalThreadAngle = Object.ConicalThreadAngle	=	Получить свойство (*)
ConicalThreadAngle = Object.GetConicalThreadAngle()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ConicalThreadAngle(&ConicalThreadAngle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Diameter – Номинальный диаметр резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

Diameter = Object.Diameter	Получить свойство (*)
Object.Diameter = Diameter	Установить свойство (*)
Diameter = Object.GetDiameter()	Получить свойство (**)
Object.SetDiameter(Diameter)	Установить свойство (**)

Синтаксис COM:

Object.get_Diameter(&Diameter)	Получить свойство
Object.put_Diameter(Diameter)	Установить свойство

Internal – TRUE – внутренняя резьба, FALSE – внешняя

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Internal = Object.Internal	Получить свойство (*)
Internal = Object.GetInternal()	Получить свойство (**)

Синтаксис COM:

Object.get_Internal(&Internal)	Получить свойство

Примечание:

Свойство доступно только для чтения.

InternalDiameterCoefficient – Коэффициент для расчета внутреннего диаметра резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

InternalDiameterCoefficient = Object.InternalDiameterCoefficient	=	Получить свойство (*)
InternalDiameterCoefficient = Object.GetInternalDiameterCoefficient()	=	Получить свойство (**)

Синтаксис COM:

Object.get_InternalDiameterCoefficient(&InternalDiameterCoefficient)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Pitch – Шаг резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

Pitch = Object.Pitch	Получить свойство (*)
Object.Pitch = Pitch	Установить свойство (*)
Pitch = Object.GetPitch()	Получить свойство (**)
Object.SetPitch(Pitch)	Установить свойство (**)

Синтаксис COM:

Object.get_Pitch(&Pitch)	Получить свойство
Object.put_Pitch(Pitch)	Установить свойство

ThreadStandardFileName – Имя файла с параметрами стандартной резьбы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ThreadStandardFileName Object.ThreadStandardFileName	=	Получить свойство (*)
ThreadStandardFileName Object.GetThreadStandardFileName()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ThreadStandardFileName(&ThreadStandardFileName)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IThreadsParameters – методы

Init – Инициализация параметров стандартной резьбы

Интерфейс...

Синтаксис Automation:

BOOL Init(BSTR StandardFileName, double Diameter, double Pitch);

Синтаксис COM:

HRESULT Init(BSTR StandardFileName, double Diameter, double Pitch, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачи.

Входные параметры:

StandardFileName	- имя таблицы стандарта резьбы,
Diameter	- номинальное значение диаметра,
Pitch	- шаг.

Интерфейс IThreadDialogParam

Параметры диалога выбора стандартной резьбы.

Иерархия:

IDispatch

IKompasAPIObject

IThreadDialogParam

Примечание:

Интерфейс можно получить с помощью метода:
IApplicationDialogs::GetDialogParam, используя константу ksObjectThreadDialogParam.

Интерфейс применяется в методе IApplicationDialogs::SelectThread.

IThreadDialogParam – свойства

IsConic – Фильтрация типа резьбы. Коническая или цилиндрическая

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsConic = Object.IsConic ;	Получить свойство (*)
Object.IsConic = IsConic;	Установить свойство (*)
IsConic = Object.GetIsConic();	Получить свойство (**)
Object.SetIsConic(IsConic);	Установить свойство (**)

Синтаксис COM:

Object.get_IsConic(&IsConic);	Получить свойство
Object.put_IsConic(IsConic);	Установить свойство

NominalDiameter – D – номинальный диаметр резьбы (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

NominalDiameter = Object.NominalDiameter;	Получить свойство (*)
Object.NominalDiameter = NominalDiameter;	Установить свойство (*)

```
NominalDiameter =                Получить свойство (**)
Object.GetNominalDiameter();
Object.SetNominalDiameter( NominalDiameter );   Установить свойство (**)
```

Синтаксис COM:

```
Object.get_NominalDiameter( &NominalDiameter );   Получить свойство
Object.put_NominalDiameter(  NominalDiameter );   Установить свойство
```

P – P – шаг резьбы

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
P = Object.P;                    Получить свойство (* )
Object.P = P;                    Установить свойство (* )
P = Object.GetP();               Получить свойство (**)
Object.SetP( P );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_P( &P ) ;             Получить свойство
Object.put_P(  P );              Установить свойство
```

Standart – Стандарт резьбы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Standart = Object.Standart;      Получить свойство (* )
Object.Standart = Standart;      Установить свойство (* )
Standart =                       Получить свойство (**)
Object.GetStandart();
Object.SetStandart(              Установить свойство (**)
Standart );
```

Синтаксис COM:

Object.get_Standart(&Standart);	Получить свойство
Object.put_Standart(Standart);	Установить свойство

Интерфейс IChooseBodies7

Справка системы КОМПАС...

kompas.chm::/301_gl_33_Oblast_primenenija_operaciy.htm

Интерфейс области применения для тел документа в операции.

Иерархия:

IDispatch

IChooseBodies7

Описание:

Позволяет получить интерфейс области применения для тел документа в операции.

Примечание:

Данный интерфейс можно получить у операции выдавливание IExtrusion и интерфейса твердого тела IBody7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);

IChooseBodies7 – свойства

Bodies – Массив тел (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Bodies = iObject.Bodies;	Получить свойство (*)
iObject.Bodies = Bodies;	Установить свойство (*)
Bodies = iObject.GetBodies();	Получить свойство (**)
iObject.SetBodies(Bodies);	Установить свойство (**)

Синтаксис COM:

iObject->get_Bodies(&Bodies);	Получить свойство
iObject->put_Bodies(Bodies);	Установить свойство

Примечание:

Позволяет установить и получить массив тел (SAFEARRAY) в области применения.

ChooseBodiesType – Тип действия над телами из перечисления ksChooseBodiesType

Интерфейс...

Тип данных: из перечисления ksChooseBodiesType

Синтаксис Automation:

ChooseBodiesType = iObject.ChooseBodiesType ;	Получить свойство (*)
iObject.ChooseBodiesType = ChooseBodiesType;	Установить свойство (*)
ChooseBodiesType = iObject.GetChooseBodiesType();	Получить свойство (**)
iObject.SetChooseBodiesType(ChooseBodiesType);	Установить свойство (**)

Синтаксис COM:

iObject->get_ChooseBodiesType(&ChooseBodiesType);	Получить свойство
iObject->put_ChooseBodiesType(ChooseBodiesType);	Установить свойство

Примечание:

Позволяет установить и получить тип действия над телами.

Интерфейс IExtrusion1

Справка системы КОМПАС...

`kompas.chm::/237_25_1_elem_vydavl.htm`

Интерфейс параметров элемента выдавливания.

Иерархия:

IDispatch

IExtrusion1

Является дополнительным интерфейсом для IExtrusion.

IExtrusion1 – свойства

CutOffByPoint – Флаг «отсекать» для способа выдавливания До точки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CutOffByPoint( Normal ) = Object.CutOffByPoint          Получить свойство (* )
Object.CutOffByPoint( Normal ) = CutOffByPoint          Установить свойство (* )
CutOffByPoint = Object.GetCutOffByPoint( Normal )       Получить свойство (**)
Object.SetCutOffByPoint( Normal, CutOffByPoint )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CutOffByPoint( Normal, &CutOffByPoint )      Получить свойство
Object.put_CutOffByPoint( Normal,  CutOffByPoint )      Установить свойство
```

DirectionObject – Направляющий объект для элемента выдавливания

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
DirectionObject = Object.DirectionObject          Получить свойство (* )
Object.DirectionObject = DirectionObject          Установить свойство (* )
DirectionObject = Object.GetDirectionObject()     Получить свойство (**)
Object.SetDirectionObject( DirectionObject )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DirectionObject( &DirectionObject )    Получить свойство
Object.put_DirectionObject(  DirectionObject )    Установить свойство
```

Свойство позволяет устанавливать и получать объект, задающий направление выдавливания. Это может быть кривая 3D, ребро или эскиз.

OperationResult – Результат операции

Интерфейс...

Тип данных: Значение из перечисления ksOperationResultEnum.

Синтаксис Automation:

```
OperationResult = Object.OperationResult        Получить свойство (* )
Object.OperationResult = OperationResult        Установить свойство (* )
OperationResult                          =      Получить свойство (**)
Object.GetOperationResult()
Object.SetOperationResult( OperationResult )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_OperationResult(                     Получить свойство
&OperationResult )
Object.put_OperationResult(                     Установить свойство
OperationResult )
```

Profile – Сечение элемента выдавливания

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
Profile = Object.Profile        Получить свойство (* )
Object.Profile = Profile        Установить свойство (* )
Profile                  =      Получить свойство (**)
Object.GetProfile()
Object.SetProfile( Profile      Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_Profile(                     Получить свойство
&Profile )
Object.put_Profile(                     Установить свойство
Profile )
```

Свойство позволяет устанавливать и получать объект, определяющий форму сечения операции. Это может быть эскиз, грань, кривая 3D.

Vector3D – Параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D.

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Свойство позволяет получать интерфейс вектора, задающего направление выдавливания.

Примечание:

Свойство доступно только для чтения.

Интерфейс IThinParameters

Справка системы КОМПАС...

kompas.chm::/708_84_5_Tonkaja_stenka.htm

Интерфейс параметров тонкой стенки.

Иерархия:

IDispatch

IThinParameters

Примечание:

IThinParameters является дополнительным интерфейсом операции выдавливание. Данный интерфейс можно получить у операции выдавливание IExtrusion посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif);

IThinParameters – свойства

Thin – Признак формирования тонкой стенки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Thin = iObject.Thin;	Получить свойство (*)
iObject.Thin = Thin;	Установить свойство (*)
Thin = iObject.GetThin();	Получить свойство (**)
iObject.SetThin(Thin);	Установить свойство (**)

Синтаксис COM:

iObject->get_Thin(&Thin);	Получить свойство
iObject->put_Thin(Thin);	Установить свойство

Примечание.

Позволяет установить и получить признак формирования тонкой стенки.

ThinType – Направление формирования тонкой стенки

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

ThinType = iObject.ThinType;	Получить свойство (*)
iObject.ThinType = ThinType;	Установить свойство (*)
ThinType = iObject.GetThinType();	Получить свойство (**)
iObject.SetThinType(ThinType);	Установить свойство (**)

Синтаксис COM:

iObject->get_ThinType(&ThinType);	Получить свойство
iObject->put_ThinType(ThinType);	Установить свойство

Примечание.

Позволяет установить и получить направление формирования тонкой стенки.

Thickness – Толщина стенки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Thickness = iObject.Thickness(normal);	Получить свойство (*)
iObject.Thickness(normal) = Thickness;	Установить свойство (*)
Thickness = iObject.GetThickness(normal);	Получить свойство (**)
iObject.SetThickness(normal, Thickness);	Установить свойство (**)

Синтаксис COM:

iObject->get_Thickness(normal, &Thickness);	Получить свойство
iObject->put_Thickness(normal, Thickness);	Установить свойство

Входные параметры:

normal	- TRUE - толщина стенки в направлении наружу, - FALSE - толщина стенки в направлении внутрь.

Примечание.

Позволяет установить и получить толщину стенки.

IThinParameters – методы

GetThinParameters – Получить параметры тонкой стенки

Интерфейс...

Синтаксис Automation:

BOOL GetThinParameters(BOOL * Thin,
Direction_Type * ThinType,
double * ThicknessNormal,
double * ThicknessReverse

Синтаксис COM:

HRESULT GetThinParameters(BOOL * Thin,
Direction_Type * ThinType,
double * ThicknessNormal,
double * ThicknessReverse

Выходные параметры:

Thin	- признак формирования тонкой стенки,
ThinType	- направление формирования тонкой стенки,
ThicknessNormal	- толщина стенки в направлении наружу,
ThicknessReverse	- толщина стенки в направлении внутрь.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получать параметры тонкой стенки.

SetThinParameters – Установить параметры тонкой стенки

Интерфейс...

Синтаксис Automation:

BOOL SetThinParameters(BOOL * Thin,
Direction_Type * ThinType,
double * ThicknessNormal,
double * ThicknessReverse

Синтаксис COM:

HRESULT SetThinParameters(BOOL * Thin,
Direction_Type * ThinType,
double * ThicknessNormal,
double * ThicknessReverse

Выходные параметры:

Thin	- признак формирования тонкой стенки,
ThinType	- направление формирования тонкой стенки,
ThicknessNormal	- толщина стенки в направлении наружу,
ThicknessReverse	- толщина стенки в направлении внутрь.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить параметры тонкой стенки.

Интерфейс IHole3D

Интерфейс отверстия.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IHole3D

Примечание:

Интерфейс можно получить с помощью методов:

IHoles3D::Hole3D

IHoles3D::Add

Интерфейс отверстия имеет также дополнительный интерфейс

IHoleDisposal- Размещение

IHole3D – свойства

Axis – Создавать ось

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Axis = Object.Axis	Получить свойство (*)
Object.Axis = Axis	Установить свойство (*)
Axis = Object.GetAxis()	Получить свойство (**)
Object.SetAxis(Axis)	Установить свойство (**)

Синтаксис COM:

Object.get_Axis(&Axis)	Получить свойство
Object.put_Axis(Axis)	Установить свойство

Diameter – Диаметр отверстия

Интерфейс...

Тип данных: double

Синтаксис Automation:

Diameter = Object.Diameter	Получить свойство (*)
Object.Diameter = Diameter	Установить свойство (*)
Diameter = Object.GetDiameter()	Получить свойство (**)
Object.SetDiameter(Diameter)	Установить свойство (**)

Синтаксис COM:

Object.get_Diameter(&Diameter)	Получить свойство
Object.put_Diameter(Diameter)	Установить свойство

Depth – Глубина отверстия

Интерфейс...

Тип данных: double

Синтаксис Automation:

Depth = Object.Depth	Получить свойство (*)
Object.Depth = Depth	Установить свойство (*)
Depth = Object.GetDepth()	Получить свойство (**)
Object.SetDepth(Depth)	Установить свойство (**)

Синтаксис COM:

Object.get_Depth(&Depth)	Получить свойство
Object.put_Depth(Depth)	Установить свойство

DepthType – Способ определения глубины отверстия

Интерфейс...

Тип данных: из перечисления ksDepthTypeEnum

Синтаксис Automation:

DepthType = Object.DepthType	Получить свойство (*)
Object.DepthType = DepthType	Установить свойство (*)
DepthType = Object.GetDepthType()	Получить свойство (**)
Object.SetDepthType(DepthType)	Установить свойство (**)

Синтаксис COM:

Object.get_DepthType(&DepthType)	Получить свойство
Object.put_DepthType(DepthType)	Установить свойство

DepthVertex – Вершина или точка для определения глубины отверстия

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

DepthVertex = Object.DepthVertex	Получить свойство (*)
Object.DepthVertex = DepthVertex	Установить свойство (*)
DepthVertex = Object.GetDepthVertex()	Получить свойство (**)
Object.SetDepthVertex(DepthVertex)	Установить свойство (**)

Синтаксис COM:

Object.get_DepthVertex(&DepthVertex)	Получить свойство
Object.put_DepthVertex(DepthVertex)	Установить свойство

DepthFace – Поверхность для определения глубины отверстия

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

DepthFace = Object.DepthFace	Получить свойство (*)
Object.DepthFace = DepthFace	Установить свойство (*)

```
DepthFace = Object.GetDepthFace()                    Получить свойство (**)
Object.SetDepthFace( DepthFace )                     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DepthFace( &DepthFace )                   Получить свойство
Object.put_DepthFace(  DepthFace )                   Установить свойство
```

EndFaceType – Форма торца

Интерфейс...

Тип данных: из перечисления ksEndFaceTypeEnum

Синтаксис Automation:

```
EndFaceType = Object.EndFaceType                     Получить свойство (* )
Object.EndFaceType = EndFaceType                     Установить свойство (* )
EndFaceType = Object.GetEndFaceType()                Получить свойство (**)
Object.SetEndFaceType( EndFaceType )                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EndFaceType( &EndFaceType )               Получить свойство
Object.put_EndFaceType(  EndFaceType )               Установить свойство
```

EndFaceAngle – Угол конуса торца

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
EndFaceAngle = Object.EndFaceAngle                   Получить свойство (* )
Object.EndFaceAngle = EndFaceAngle                   Установить свойство (* )
EndFaceAngle = Object.GetEndFaceAngle()              Получить свойство (**)
Object.SetEndFaceAngle( EndFaceAngle )               Установить свойство (**)
```

Синтаксис COM:

```
Object.get_EndFaceAngle( &EndFaceAngle )             Получить свойство
Object.put_EndFaceAngle(  EndFaceAngle )             Установить свойство
```

HoleType – Тип отверстия

Интерфейс...

Тип данных: из перечисления ksHoleTypeEnum

Синтаксис Automation:

```
HoleType = Object.HoleType                  Получить свойство (* )
Object.HoleType = HoleType                  Установить свойство (* )
HoleType = Object.GetHoleType()             Получить свойство (**)
Object.SetHoleType( HoleType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HoleType( &HoleType )            Получить свойство
Object.put_HoleType(  HoleType )            Установить свойство
```

HoleParameters – Параметры отверстия

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
HoleParameters                  =       Получить свойство (* )
Object.HoleParameters
THoleParameters                 =       Получить свойство (**)
Object.GetHoleParameters()
```

Синтаксис COM:

```
Object.get_HoleParameters(              Получить свойство,
&HoleParameters )
```

Примечание:

Свойство доступно только для чтения.

В зависимости от типа отверстия возвращаются интерфейсы

- ksHTCounterbore - ISpotfacingHoleParameters
- ksHTCountersinking - ICountersinkHoleParameters
- ksHTCounterdrill - ICountersinkSpotfacingHoleParameters
- ksHTConic - IConicHoleParameters
- ksHTLfrLibrary - ILibraryHoleParameters

ShowThread – Показывать резьбу

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ShowThread = Object.ShowThread              Получить свойство (* )
Object.ShowThread = ShowThread              Установить свойство (* )
ShowThread = Object.GetShowThread()         Получить свойство (**)
```

Object.SetShowThread(ShowThread)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowThread(&ShowThread)	Получить свойство
Object.put_ShowThread(ShowThread)	Установить свойство

Thread – Условное обозначение резьбы

Интерфейс...

Тип данных: Указатель на интерфейс IThread

Синтаксис Automation:

Thread = Object.Thread	Получить свойство (*)
Thread = Object.GetThread()	Получить свойство (**)

Синтаксис COM:

Object.get_Thread(&Thread)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Интерфейс ICountersinkHoleParameters

Интерфейс отверстия с зенковкой.

Иерархия:

IDispatch

IKompasAPIObject

ICountersinkHoleParameters

Интерфейс можно получить с помощью метода:

IHole3D::HoleParameters

ICountersinkHoleParameters – свойства

CountersinkAngle – Угол конуса зенковки

Интерфейс...

Тип данных: double

Синтаксис Automation:

CountersinkAngle = Object.CountersinkAngle	Получить свойство (*)
Object.CountersinkAngle = CountersinkAngle	Установить свойство (*)

CountersinkAngle	=	Получить свойство (**)
Object.GetCountersinkAngle()		
Object.SetCountersinkAngle(CountersinkAngle)		Установить свойство (**)

Синтаксис COM:

Object.get_CountersinkAngle(&CountersinkAngle)	Получить свойство
Object.put_CountersinkAngle(CountersinkAngle)	Установить свойство

CountersinkDepth – Глубина зенковки

Интерфейс...

Тип данных: double

Синтаксис Automation:

CountersinkDepth = Object.CountersinkDepth		Получить свойство (*)
Object.CountersinkDepth = CountersinkDepth		Установить свойство (*)
CountersinkDepth	=	Получить свойство (**)
Object.GetCountersinkDepth()		
Object.SetCountersinkDepth(CountersinkDepth)		Установить свойство (**)

Синтаксис COM:

Object.get_CountersinkDepth(&CountersinkDepth)	Получить свойство
Object.put_CountersinkDepth(CountersinkDepth)	Установить свойство

CountersinkDiameter – Диаметр зенковки

Интерфейс...

Тип данных: double

Синтаксис Automation:

CountersinkDiameter	=	Получить свойство (*)
Object.CountersinkDiameter		
Object.CountersinkDiameter	=	Установить свойство (*)
CountersinkDiameter		
CountersinkDiameter	=	Получить свойство (**)
Object.GetCountersinkDiameter()		
Object.SetCountersinkDiameter(		Установить свойство (**)
CountersinkDiameter)		

Синтаксис COM:

Object.get_CountersinkDiameter(&CountersinkDiameter)	Получить свойство
Object.put_CountersinkDiameter(CountersinkDiameter)	Установить свойство

CountersinkType – Способ определения параметров зенковки

Интерфейс...

Тип данных: из перечисления ksCountersinkTypeEnum

Синтаксис Automation:

CountersinkType = Object.CountersinkType	Получить свойство (*)
Object.CountersinkType = CountersinkType	Установить свойство (*)
CountersinkType = Object.GetCountersinkType()	Получить свойство (**)
Object.SetCountersinkType(CountersinkType)	Установить свойство (**)

Синтаксис COM:

Object.get_CountersinkType(&CountersinkType)	Получить свойство
Object.put_CountersinkType(CountersinkType)	Установить свойство

Интерфейс ICountersinkSpotfacingHoleParameters

Интерфейс отверстия с зенковкой и цековкой.

Иерархия:

IDispatch

IKompasAPIObject

ICountersinkSpotfacingHoleParameters

Интерфейс можно получить с помощью метода:

IHole3D::HoleParameters

ICountersinkSpotfacingHoleParameters – свойства

CountersinkAngle – Угол конуса зенковки

Интерфейс...

Тип данных: double

Синтаксис Automation:

CountersinkAngle = Object.CountersinkAngle	Получить свойство (*)
Object.CountersinkAngle = CountersinkAngle	Установить свойство (*)
CountersinkAngle = Object.GetCountersinkAngle()	Получить свойство (**)
Object.SetCountersinkAngle(CountersinkAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_CountersinkAngle(&CountersinkAngle)	Получить свойство
Object.put_CountersinkAngle(CountersinkAngle)	Установить свойство

SpotfacingCountersinkDiameter – Диаметр цековки/зенковки

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpotfacingCountersinkDiameter = Object.SpotfacingCountersinkDiameter	Получить свойство (*)
Object.SpotfacingCountersinkDiameter = SpotfacingCountersinkDiameter	Установить свойство (*)
SpotfacingCountersinkDiameter = Object.GetSpotfacingCountersinkDiameter()	Получить свойство (**)
Object.SetSpotfacingCountersinkDiameter(SpotfacingCountersinkDiameter)	Установить свойство (**)

Синтаксис COM:

Object.get_SpotfacingCountersinkDiameter(&SpotfacingCountersinkDiameter)	Получить свойство
Object.put_SpotfacingCountersinkDiameter(SpotfacingCountersinkDiameter)	Установить свойство

SpotfacingDepth – Глубина цековки

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpotfacingDepth = Object.SpotfacingDepth	Получить свойство (*)

Object.SpotfacingDepth = SpotfacingDepth	Установить свойство (*)
SpotfacingDepth = Object.GetSpotfacingDepth()	Получить свойство (**)
Object.SetSpotfacingDepth(SpotfacingDepth)	Установить свойство (**)

Синтаксис COM:

Object.get_SpotfacingDepth(&SpotfacingDepth)	Получить свойство
Object.put_SpotfacingDepth(SpotfacingDepth)	Установить свойство

Интерфейс IConicHoleParameters

Интерфейс конического отверстия.

Иерархия:

IDispatch

IKompasAPIObject

IConicHoleParameters

Интерфейс можно получить с помощью метода:

IHole3D::HoleParameters

IConicHoleParameters – свойства

ConicAngle – Угол конуса

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConicAngle = Object.ConicAngle	Получить свойство (*)
Object.ConicAngle = ConicAngle	Установить свойство (*)
ConicAngle = Object.GetConicAngle(	Получить свойство (**)
Object.SetConicAngle(ConicAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_ConicAngle(&ConicAngle)	Получить свойство
Object.put_ConicAngle(ConicAngle)	Установить свойство

ConicDiameter – Диаметр основания конуса

Интерфейс...

Тип данных: double

Синтаксис Automation:

ConicDiameter = Object.ConicDiameter	Получить свойство (*)

Object.ConicDiameter = ConicDiameter	Установить свойство (*)
ConicDiameter = Object.GetConicDiameter()	Получить свойство (**)
Object.SetConicDiameter(ConicDiameter)	Установить свойство (**)

Синтаксис COM:

Object.get_ConicDiameter(&ConicDiameter)	Получить свойство
Object.put_ConicDiameter(ConicDiameter)	Установить свойство

ConicType – Способ определения параметров конического отверстия

Интерфейс...

Тип данных: из перечисления ksConicTypeEnum

Синтаксис Automation:

ConicType = Object.ConicType	Получить свойство (*)
Object.ConicType = ConicType	Установить свойство (*)
ConicType = Object.GetConicType()	Получить свойство (**)
Object.SetConicType(ConicType)	Установить свойство (**)

Синтаксис COM:

Object.get_ConicType(&ConicType)	Получить свойство
Object.put_ConicType(ConicType)	Установить свойство

Интерфейс ISpotfacingHoleParameters

Интерфейс отверстия с цековкой.

Иерархия:

IDispatch

IKompasAPIObject

ISpotfacingHoleParameters

Интерфейс можно получить с помощью метода:

IHole3D::HoleParameters

ISpotfacingHoleParameters – свойства

SpotfacingDiameter – Диаметр цековки

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpotfacingDiameter = Object.SpotfacingDiameter	Получить свойство (*)
Object.SpotfacingDiameter = SpotfacingDiameter	Установить свойство (*)
SpotfacingDiameter = Object.GetSpotfacingDiameter()	Получить свойство (**)
Object.SetSpotfacingDiameter(SpotfacingDiameter)	Установить свойство (**)

Синтаксис COM:

Object.get_SpotfacingDiameter(&SpotfacingDiameter)	Получить свойство
Object.put_SpotfacingDiameter(SpotfacingDiameter)	Установить свойство

SpotfacingDepth – Глубина цековки

Интерфейс...

Тип данных: double

Синтаксис Automation:

SpotfacingDepth = Object.SpotfacingDepth	Получить свойство (*)
Object.SpotfacingDepth = SpotfacingDepth	Установить свойство (*)
SpotfacingDepth = Object.GetSpotfacingDepth()	Получить свойство (**)
Object.SetSpotfacingDepth(SpotfacingDepth)	Установить свойство (**)

Синтаксис COM:

Object.get_SpotfacingDepth(&SpotfacingDepth)	Получить свойство
Object.put_SpotfacingDepth(SpotfacingDepth)	Установить свойство

Интерфейс ILibraryHoleParameters

Интерфейс параметров отверстия из библиотеки.

Иерархия:

IDispatch

IKompasAPIObject

ILibraryHoleParameters

Данный интерфейс можно получить с помощью метода IHole3D::HoleParameters

Версия Компас v18.1

ILibraryHoleParameters – свойства

FileName – Форма. Путь к библиотеке и имя фрагмента в библиотеке документов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Variable – Получить параметрическую переменную по имени или индексу

Интерфейс...

Тип данных: Указатель на интерфейс IVariable7.

Синтаксис Automation:

Variable = Object.Variable(Index)	Получить свойство (*)
Variable = Object.GetVariable(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Variable(Index, &Variable)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Variables – Получить массив параметрических переменных в виде SAFEARRAY VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Variables = Object.Variables	Получить свойство (*)
Variables = Object.GetVariables()	Получить свойство (**)

Синтаксис COM:

Object.get_Variables(&Variables)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

VariablesCount – Получить количество параметрических переменных

Интерфейс...

Тип данных: long

Синтаксис Automation:

VariablesCount = Object.VariablesCount	Получить свойство (*)
VariablesCount = Object.GetVariablesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_VariablesCount(&VariablesCount)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Интерфейс IHoleDisposal

Интерфейс размещения отверстия.

Иерархия:

IDispatch

IHoleDisposal

Интерфейс можно получить методом QueryInterface у интерфейса IHole3D.

IHoleDisposal – свойства

AssociationVertex – Точечный объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationVertex = Object.AssociationVertex	Получить свойство (*)
Object.AssociationVertex = AssociationVertex	Установить свойство (*)
AssociationVertex = Object.GetAssociationVertex()	Получить свойство (**)
Object.SetAssociationVertex(AssociationVertex)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationVertex(&AssociationVertex	Получить свойство
Object.put_AssociationVertex(AssociationVertex)	Установить свойство

BaseSurface – Поверхность, на которой требуется построить точку

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

BaseSurface = Object.BaseSurface	Получить свойство (*)
Object.BaseSurface = BaseSurface	Установить свойство (*)
BaseSurface = Object.GetBaseSurface()	Получить свойство (**)
Object.SetBaseSurface(BaseSurface)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseSurface(&BaseSurface)	Получить свойство
Object.put_BaseSurface(BaseSurface)	Установить свойство

Direction – Направление построения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)

Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

OffsetType – Тип смещения

Интерфейс...

Тип данных: из перечисления ksPoint3DSurfaceParamTypeEnum

Синтаксис Automation:

OffsetType = Object.OffsetType	Получить свойство (*)
Object.OffsetType = OffsetType	Установить свойство (*)
OffsetType = Object.GetOffsetType()	Получить свойство (**)
OffsetType = Object.GetOffsetType()	Установить свойство (**)

Синтаксис COM:

Object.get_OffsetType(&OffsetType)	Получить свойство
Object.put_OffsetType(OffsetType)	Установить свойство

Perpendicular – Перпендикулярно поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Perpendicular = Object.Perpendicular	Получить свойство (*)
Object.Perpendicular = Perpendicular	Установить свойство (*)
Perpendicular = Object.GetPerpendicular()	Получить свойство (**)
Object.SetPerpendicular(Perpendicular)	Установить свойство (**)

Синтаксис COM:

Object.get_Perpendicular(&Perpendicular)	Получить свойство
Object.put_Perpendicular(Perpendicular)	Установить свойство

Point3DParamSurface – Параметры пространственной точки на поверхности

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Point3DParamSurface = Object.Point3DParamSurface	=	Получить свойство (*)
Point3DParamSurface = Object.GetPoint3DParamSurface()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Point3DParamSurface(&Point3DParamSurface)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

ProcessCanopy – Обрабатывать навес

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProcessCanopy = Object.ProcessCanopy	Получить свойство (*)
Object.ProcessCanopy = ProcessCanopy	Установить свойство (*)
ProcessCanopy = Object.GetProcessCanopy()	Получить свойство (**)
Object.SetProcessCanopy(ProcessCanopy)	Установить свойство (**)

Синтаксис COM:

Object.get_ProcessCanopy(&ProcessCanopy)	Получить свойство
Object.put_ProcessCanopy(ProcessCanopy)	Установить свойство

Vector – Вектор

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector = Object.Vector	Получить свойство (*)

Vector = Object.GetVector() Получить свойство (**)

Синтаксис COM:

Object.get_Vector(&Vector) Получить свойство,

Примечание:

Свойство доступно только для чтения.

Интерфейс IChamfer

Интерфейс операции Фаска.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IChamfer

Интерфейс можно получить с помощью метода коллекции операций фаска IChamfers::Add или свойства IChamfers::Chamfer.

КОМПАС версия v18

IChamfer – свойства

Angle – Угол фаски

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle Получить свойство (*)
Object.Angle = Angle Установить свойство (*)
Angle = Object.GetAngle() Получить свойство (**)
Object.SetAngle(Angle) Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle) Получить свойство
Object.put_Angle(Angle) Установить свойство

BaseObjects – Массив ребер и граней, на которых строится фаска

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

BaseObjects = Object.BaseObjects	Получить свойство (*)
Object.BaseObjects = BaseObjects	Установить свойство (*)
BaseObjects = Object.GetBaseObjects()	Получить свойство (**)
Object.SetBaseObjects(BaseObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjects(&BaseObjects)	Получить свойство
Object.put_BaseObjects(BaseObjects)	Установить свойство

Примечание.

Позволяет получать и устанавливать опорный объект или массив опорных объектов, участвующих в операции фаска.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

BuildingType – Способ построения фаски

Интерфейс...

Тип данных: Значение из перечисления ksChamferBuildingTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType())	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Direction – Направление фаски

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Direction( &Direction )      Получить свойство
Object.put_Direction(  Direction )      Установить свойство
```

Значение свойства:

```
TRUE                                    - первое,
FALSE                                   - второе.
```

Distance1 – Длина первой стороны

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Distance1 = Object.Distance1            Получить свойство (* )
Object.Distance1 = Distance1            Установить свойство (* )
Distance1 = Object.GetDistance1()       Получить свойство (**)
Object.SetDistance1( Distance1 )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Distance1( &Distance1 )      Получить свойство
Object.put_Distance1(  Distance1 )      Установить свойство
```

Distance2 – Длина второй стороны

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Distance2 = Object.Distance2            Получить свойство (* )
Object.Distance2 = Distance2            Установить свойство (* )
Distance2 = Object.GetDistance2()       Получить свойство (**)
Object.SetDistance2( Distance2 )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Distance2( &Distance2 )      Получить свойство
Object.put_Distance2(  Distance2 )      Установить свойство
```

Tangent – По касательным ребрам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Tangent = Object.Tangent	Получить свойство (*)
Object.Tangent = Tangent	Установить свойство (*)
Tangent = Object.GetTangent()	Получить свойство (**)
Object.SetTangent(Tangent)	Установить свойство (**)

Синтаксис COM:

Object.get_Tangent(&Tangent)	Получить свойство
Object.put_Tangent(Tangent)	Установить свойство

Интерфейс ICoupling

Интерфейс цепочки операции Сечение.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICoupling

Данный интерфейс можно получить с помощью методов операции по сечениям ILoft::AddCoupling и ILoft::Coupling.

КОМПАС версия v18

ICoupling – свойства

Count – Количество сечений в цепочке

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Count = Object.GetCount()	Получить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Position – Величина смещения точки вдоль контура сечения в %

Интерфейс...

Тип данных: double

Синтаксис Automation:

Position = Object.Position(Index)	Получить свойство (*)
Object.Position(Index) = Position	Установить свойство (*)
Position = Object.GetPosition(Index)	Получить свойство (**)
Object.SetPosition(Index, Position)	Установить свойство (**)

Синтаксис COM:

Object.get_Position(Index, &Position)	Получить свойство
Object.put_Position(Index, Position)	Установить свойство

Входные параметры:

long Index	- индекс сечения в цепочке.

PositionObject – Фиксирующий объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

PositionObject = Object.PositionObject(Index)	Получить свойство (*)
Object.PositionObject(Index) = PositionObject	Установить свойство (*)
PositionObject = Object.GetPositionObject(Index)	Получить свойство (**)
Object.SetPositionObject(Index, PositionObject)	Установить свойство (**)

Синтаксис COM:

Object.get_PositionObject(Index, &PositionObject)	Получить свойство
Object.put_PositionObject(Index, PositionObject)	Установить свойство

Входные параметры:

long Index	- индекс сечения в цепочке.

PositionOffset – Величина смещения точки вдоль контура сечения в мм

Интерфейс...

Тип данных: double

Синтаксис Automation:

PositionOffset = Object.PositionOffset(Index)	Получить свойство (*)
Object.PositionOffset(Index) = PositionOffset	Установить свойство (*)
PositionOffset = Object.GetPositionOffset(Index)	Получить свойство (**)
Object.SetPositionOffset(Index, PositionOffset)	Установить свойство (**)

Синтаксис COM:

Object.get_PositionOffset(Index, &PositionOffset)	Получить свойство
Object.put_PositionOffset(Index, PositionOffset)	Установить свойство

Входные параметры:

long Index	- индекс сечения в цепочке.

Sketch – Сечение. Эскиз, контур, пространственная кривая, грань

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Sketch = Object.Sketch(Index)	Получить свойство (*)
Sketch = Object.GetSketch(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Sketch(Index, &Sketch)	Получить свойство

Входные параметры:

long Index	- индекс сечения в цепочке.

Примечание:

Свойство доступно только для чтения.

ICoupling – методы

Delete – Удалить цепочку

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetPoint – Получить координаты точки

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long Index, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(long Index, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

long Index	- индекс сечения в цепочке.

Выходные параметры:

X, Y, Z	- координаты точки.

SetPoint – Получить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(long Index, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(long Index, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

long Index - индекс сечения в цепочке,
X, Y, Z - координаты точки.

Интерфейс ICut

Интерфейс операции Сечение.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
ICut

КОМПАС версия v18

ICut – свойства

BuildingType – Способ сечения при сечении по эскизу

Интерфейс...

Тип данных: Значение из перечисления ksCutBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType — Получить свойство (*)
Object.BuildingType = BuildingType — Установить свойство (*)
BuildingType = Object.GetBuildingType() — Получить свойство (**)
Object.SetBuildingType(BuildingType) — Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType) — Получить свойство
Object.put_BuildingType(BuildingType) — Установить свойство

ChooseBodies – Массив тел

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
ChooseBodies = Object.ChooseBodies                  Получить свойство (* )
Object.ChooseBodies = ChooseBodies                  Установить свойство (* )
ChooseBodies = Object.GetChooseBodies()             Получить свойство (**)
Object.SetChooseBodies( ChooseBodies )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ChooseBodies( &ChooseBodies )            Получить свойство
Object.put_ChooseBodies( ChooseBodies )             Установить свойство
```

Примечание.

Позволяет получать и устанавливать тело или массив тел, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY тел LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ChooseParts – Массив компонентов (SAFEARRAY)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
ChooseParts = Object.ChooseParts                    Получить свойство (* )
Object.ChooseParts = ChooseParts                    Установить свойство (* )
ChooseParts = Object.GetChooseParts()               Получить свойство (**)
Object.SetChooseParts( ChooseParts )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ChooseParts( &ChooseParts )              Получить свойство
Object.put_ChooseParts( ChooseParts )               Установить свойство
```

Примечание.

Позволяет получать и устанавливать компонент или массив компонентов, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY компонентов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ChoosePartsType – Способ определения области применения для компонентов

Интерфейс...

Тип данных: Значение из перечисления ksChoosePartsType.

Синтаксис Automation:

ChoosePartsType	=	Получить свойство (*)
Object.ChoosePartsType		
Object.ChoosePartsType	=	Установить свойство (*)
ChoosePartsType		
ChoosePartsType	=	Получить свойство (**)
Object.GetChoosePartsType()		
Object.SetChoosePartsType(		Установить свойство (**)
ChoosePartsType)		

Синтаксис COM:

Object.get_ChoosePartsType(	Получить свойство
&ChoosePartsType)	
Object.put_ChoosePartsType(	Установить свойство
ChoosePartsType)	

ChooseType – Область применения

Интерфейс...

Тип данных: Значение из перечисления ksChooseType.

Синтаксис Automation:

ChooseType = Object.ChooseType	Получить свойство (*)
Object.ChooseType = ChooseType	Установить свойство (*)
ChooseType = Object.GetChooseType()	Получить свойство (**)
Object.SetChooseType(ChooseType)	Установить свойство (**)

Синтаксис COM:

Object.get_ChooseType(&ChooseType)	Получить свойство
Object.put_ChooseType(ChooseType)	Установить свойство

CutObject – Секущий объект (эскиз или поверхность)

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

CutObject = Object.CutObject	Получить свойство (*)
Object.CutObject = CutObject	Установить свойство (*)
CutObject = Object.GetCutObject()	Получить свойство (**)
Object.SetCutObject(CutObject)	Установить свойство (**)

Синтаксис COM:

Object.get_CutObject(&CutObject)	Получить свойство
Object.put_CutObject(CutObject)	Установить свойство

Direction – Направление отсечения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Интерфейс IEvolution

Интерфейс кинематической операции.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IEvolution

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям IEvolutions::Add или свойства IEvolutions::Evolution.

Интерфейс также используется для кинематической поверхности.

КОМПАС версия v18

IEvolution – свойства

BySurfaceNormal – Согласованно с нормалью поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BySurfaceNormal = Object.BySurfaceNormal	=	Получить свойство (*)
Object.BySurfaceNormal = BySurfaceNormal	=	Установить свойство (*)
BySurfaceNormal = Object.GetBySurfaceNormal()	=	Получить свойство (**)
Object.SetBySurfaceNormal(BySurfaceNormal)		Установить свойство (**)

Синтаксис COM:

Object.get_BySurfaceNormal(&BySurfaceNormal)	Получить свойство
Object.put_BySurfaceNormal(BySurfaceNormal)	Установить свойство

Edges – Траектория. Массив SAFEARRAY кривых, задающих траекторию движения сечения кинематического элемента

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Object.Edges = Edges	Установить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)
Object.SetEdges(Edges)	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство
Object.put_Edges(Edges)	Установить свойство

Примечание.

Позволяет получать и устанавливать массив кривых, задающих траекторию движения сечения кинематического элемента.

Массив возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

OperationResult – Результат операции

Интерфейс...

Тип данных: Значение из перечисления ksOperationResultEnum.

Синтаксис Automation:

```
OperationResult = Object.OperationResult          Получить свойство (* )
Object.OperationResult = OperationResult          Установить свойство (* )
OperationResult                          =        Получить свойство (**)
Object.GetOperationResult()
Object.SetOperationResult(                        Установить свойство (**)
OperationResult )
```

Синтаксис COM:

```
Object.get_OperationResult(                       Получить свойство
&OperationResult)
Object.put_OperationResult(                       Установить свойство
OperationResult )
```

Sketch – Сечение. Эскиз

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Sketch = Object.Sketch                            Получить свойство (* )
Object.Sketch = Sketch                            Установить свойство (* )
Sketch = Object.GetSketch()                       Получить свойство (**)
Object.SetSketch( Sketch )                        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch )                      Получить свойство
Object.put_Sketch(  Sketch )                      Установить свойство
```

SketchShiftType – Тип движения сечения по траектории

Интерфейс...

Тип данных: Значение из перечисления ksEvolutionShiftSketchTypeEnum.

Синтаксис Automation:

SketchShiftType = Object.SketchShiftType	Получить свойство (*)
Object.SketchShiftType = SketchShiftType	Установить свойство (*)
SketchShiftType = Object.GetSketchShiftType()	Получить свойство (**)
Object.SetSketchShiftType(SketchShiftType)	Установить свойство (**)

Синтаксис COM:

Object.get_SketchShiftType(&SketchShiftType)	Получить свойство
Object.put_SketchShiftType(SketchShiftType)	Установить свойство

IEvolution – методы

GetPathLength – Длина траектории

Интерфейс...

Синтаксис Automation:

double GetPathLength(ksLengthUnitsEnum Unit);

Синтаксис COM:

HRESULT GetPathLength(ksLengthUnitsEnum Unit, double * Result);

Возвращаемое значение:

- длина траектории.

Входные параметры:

Unit	- единицы измерения длины из перечисления ksLengthUnitsEnum.

Интерфейс IFillet

Интерфейс операции Скругление.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFillet

Интерфейс можно получить с помощью метода коллекции операций скругления IFillets::Add или свойства IFillets::Fillet.

IFillet – свойства

AutoSaveEdge – Автоопределение сохранения кромки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
AutoSaveEdge = Object.AutoSaveEdge                  Получить свойство (* )
Object.AutoSaveEdge = AutoSaveEdge                  Установить свойство (* )
AutoSaveEdge = Object.GetAutoSaveEdge()             Получить свойство (**)
Object.SetAutoSaveEdge( AutoSaveEdge )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AutoSaveEdge( &AutoSaveEdge )            Получить свойство
Object.put_AutoSaveEdge(  AutoSaveEdge )            Установить свойство
```

BaseObjects – Массив скругляемых объектов (граней и ребер)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
BaseObjects = Object.BaseObjects                    Получить свойство (* )
Object.BaseObjects = BaseObjects                    Установить свойство (* )
BaseObjects = Object.GetBaseObjects()               Получить свойство (**)
Object.SetBaseObjects( BaseObjects )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BaseObjects( &BaseObjects )              Получить свойство
Object.put_BaseObjects(  BaseObjects )              Установить свойство
```

BuildingType – Способ построения скругления

Интерфейс...

Тип данных: Значение из перечисления ksFilletBuildingTypeEnum.

Синтаксис Automation:

```
BuildingType = Object.BuildingType                  Получить свойство (* )
Object.BuildingType = BuildingType                  Установить свойство (* )
BuildingType = Object.GetBuildingType()             Получить свойство (**)
```

Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Примечание.

Метод позволяет получать и устанавливать опорный объект или массив опорных объектов, участвующих в операции **Скругление**.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Chord – Хорда

Интерфейс...

Тип данных: double

Синтаксис Automation:

Chord = Object.Chord	Получить свойство (*)
Object.Chord = Chord	Установить свойство (*)
Chord = Object.GetChord()	Получить свойство (**)
Object.SetChord(Chord)	Установить свойство (**)

Синтаксис COM:

Object.get_Chord(&Chord)	Получить свойство
Object.put_Chord(Chord)	Установить свойство

Coefficient – Коэффициент

Интерфейс...

Тип данных: double

Синтаксис Automation:

Coefficient = Object.Coefficient	Получить свойство (*)
Object.Coefficient = Coefficient	Установить свойство (*)
Coefficient = Object.GetCoefficient()	Получить свойство (**)
Object.SetCoefficient(Coefficient)	Установить свойство (**)

Синтаксис COM:

Object.get_Coefficient(&Coefficient)	Получить свойство
Object.put_Coefficient(Coefficient)	Установить свойство

Radius1 – Радиус скругления или полуось 1

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius1 = Object.Radius1                    Получить свойство (* )
Object.Radius1 = Radius1                    Установить свойство (* )
Radius1 = Object.GetRadius1()               Получить свойство (**)
Object.SetRadius1( Radius1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius1( &Radius1 )              Получить свойство
Object.put_Radius1(  Radius1 )              Установить свойство
```

Radius2 – Полуось 2

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Radius2 = Object.Radius2                    Получить свойство (* )
Object.Radius2 = Radius2                    Установить свойство (* )
Radius2 = Object.GetRadius2()               Получить свойство (**)
Object.SetRadius2( Radius2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Radius2( &Radius2 )              Получить свойство
Object.put_Radius2(  Radius2 )              Установить свойство
```

RoundCorners – Обход углов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
RoundCorners = Object.RoundCorners          Получить свойство (* )
Object.RoundCorners = RoundCorners          Установить свойство (* )
RoundCorners = Object.GetRoundCorners()     Получить свойство (**)
Object.SetRoundCorners( RoundCorners )      Установить свойство (**)
```

Синтаксис COM:

Object.get_RoundCorners(&RoundCorners)	Получить свойство
Object.put_RoundCorners(RoundCorners)	Установить свойство

SaveEdge – Сохранять кромку

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveEdge = Object.SaveEdge	Получить свойство (*)
Object.SaveEdge = SaveEdge	Установить свойство (*)
SaveEdge = Object.GetSaveEdge()	Получить свойство (**)
Object.SetSaveEdge(SaveEdge)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveEdge(&SaveEdge)	Получить свойство
Object.put_SaveEdge(SaveEdge)	Установить свойство

SmoothCorner – Сглаживать углы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SmoothCorner = Object.SmoothCorner	Получить свойство (*)
Object.SmoothCorner = SmoothCorner	Установить свойство (*)
SmoothCorner = Object.GetSmoothCorner()	Получить свойство (**)
Object.SetSmoothCorner(SmoothCorner)	Установить свойство (**)

Синтаксис COM:

Object.get_SmoothCorner(&SmoothCorner)	Получить свойство
Object.put_SmoothCorner(SmoothCorner)	Установить свойство

StopFilletCutByObject – Усекать по объекту (При секущей) для точки останова скругления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StopFilletCutByObject = Получить свойство (*)
Object.StopFilletCutByObject(First)
Object.StopFilletCutByObject(First) = Установить свойство (*)
StopFilletCutByObject
StopFilletCutByObject = Получить свойство (**)
Object.GetStopFilletCutByObject(First)
Object.SetStopFilletCutByObject(First, Установить свойство (**)
StopFilletCutByObject)

Синтаксис COM:

Object.get_StopFilletCutByObject(First, Получить свойство
&StopFilletCutByObject)
Object.put_StopFilletCutByObject(First, Установить свойство
StopFilletCutByObject)

Входные параметры:

BOOL First - TRUE - начальная точка останова,
FALSE - конечная точка останова.

StopFilletCutObject – Секущий объект для точки останова скругления

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

StopFilletCutObject = Object.StopFilletCutObject(Получить свойство (*)
First)
Object.StopFilletCutObject(First) = Установить свойство (*)
StopFilletCutObject
StopFilletCutObject = Получить свойство (**)
Object.GetStopFilletCutObject(First)
Object.SetStopFilletCutObject(First, Установить свойство (**)
StopFilletCutObject)

Синтаксис COM:

Object.get_StopFilletCutObject(First, Получить свойство
&StopFilletCutObject)
Object.put_StopFilletCutObject(First, Установить свойство
StopFilletCutObject)

Входные параметры:

BOOL First - TRUE FALSE	- начальная точка останова, - конечная точка останова.

StopFilletCutPointIndex – Индекс точки пересечения с секущим объектом для точки останова скругления

Интерфейс...

Тип данных: long

Синтаксис Automation:

StopFilletCutPointIndex Object.StopFilletCutPointIndex(First)	=	Получить свойство (*)
Object.StopFilletCutPointIndex(First) StopFilletCutPointIndex	=	Установить свойство (*)
StopFilletCutPointIndex Object.GetStopFilletCutPointIndex(First)	=	Получить свойство (**)
Object.SetStopFilletCutPointIndex(First, StopFilletCutPointIndex)		Установить свойство (**)

Синтаксис COM:

Object.get_StopFilletCutPointIndex(First, &StopFilletCutPointIndex)		Получить свойство
Object.put_StopFilletCutPointIndex(First, StopFilletCutPointIndex)		Установить свойство

Входные параметры:

BOOL First - TRUE FALSE	- начальная точка останова, - конечная точка останова.

StopFilletDirection – Направление усечения в точке останова

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StopFilletDirection = Object.StopFilletDirection	Получить свойство (*)
Object.StopFilletDirection = StopFilletDirection	Установить свойство (*)

StopFilletDirection = Object.GetStopFilletDirection()	Получить свойство (**)
Object.SetStopFilletDirection(StopFilletDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_StopFilletDirection(&StopFilletDirection)	Получить свойство
Object.put_StopFilletDirection(StopFilletDirection)	Установить свойство

Примечание:

false - направление от начальной точки до конечной.

StopFilletOffset – Величина смещения для точки останова скругления

Интерфейс...

Тип данных: double

Синтаксис Automation:

StopFilletOffset = Object.StopFilletOffset(First)	Получить свойство (*)
Object.StopFilletOffset(First) = StopFilletOffset	Установить свойство (*)
StopFilletOffset = Object.GetStopFilletOffset(First)	Получить свойство (**)
Object.SetStopFilletOffset(First, StopFilletOffset)	Установить свойство (**)

Синтаксис COM:

Object.get_StopFilletOffset(&StopFilletOffset)	First,	Получить свойство
Object.put_StopFilletOffset(StopFilletOffset)	First,	Установить свойство

Входные параметры:

BOOL First - TRUE	- начальная точка останова,
FALSE	- конечная точка останова.

StopFilletOffsetMode – Способ расчета смещения для точки останова скругления

Интерфейс...

Тип данных: Значение из перечисления ksFilletOffsetModeEnum.

Синтаксис Automation:

StopFilletOffsetMode = Object.StopFilletOffsetMode(First)		Получить свойство (*)
Object.StopFilletOffsetMode(First) = StopFilletOffsetMode		Установить свойство (*)
StopFilletOffsetMode = Object.GetStopFilletOffsetMode(First)		Получить свойство (**)
Object.SetStopFilletOffsetMode(First, StopFilletOffsetMode)		Установить свойство (**)

Синтаксис COM:

Object.get_StopFilletOffsetMode(First, &StopFilletOffsetMode)		Получить свойство
Object.put_StopFilletOffsetMode(First, StopFilletOffsetMode)		Установить свойство

Входные параметры:

BOOL First - TRUE	- начальная точка останова,
FALSE	- конечная точка останова.

StopFilletOn – Использовать точку останова скругления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

StopFilletOn = Object.StopFilletOn(First)	Получить свойство (*)
Object.StopFilletOn(First) = StopFilletOn	Установить свойство (*)
StopFilletOn = Object.GetStopFilletOn(First)	Получить свойство (**)
Object.SetStopFilletOn(First, StopFilletOn)	Установить свойство (**)

Синтаксис COM:

Object.get_StopFilletOn(First, &StopFilletOn)	Получить свойство
Object.put_StopFilletOn(First, StopFilletOn)	Установить свойство

Входные параметры:

BOOL First - TRUE	- начальная точка останова,

FALSE - конечная точка останова.

Tangent – По касательным ребрам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Tangent = Object.Tangent	Получить свойство (*)
Object.Tangent = Tangent	Установить свойство (*)
Tangent = Object.GetTangent()	Получить свойство (**)
Object.SetTangent(Tangent)	Установить свойство (**)

Синтаксис COM:

Object.get_Tangent(&Tangent)	Получить свойство
Object.put_Tangent(Tangent)	Установить свойство

VariableRadius – Значение переменного радиуса в точке

Интерфейс...

Тип данных: double

Синтаксис Automation:

VariableRadius = Object.VariableRadius(Index)	Получить свойство (*)
Object.VariableRadius(Index) = VariableRadius	Установить свойство (*)
VariableRadius = Object.GetVariableRadius(Index)	Получить свойство (**)
Object.SetVariableRadius(Index, VariableRadius)	Установить свойство (**)

Синтаксис COM:

Object.get_VariableRadius(&VariableRadius)	Index,	Получить свойство
Object.put_VariableRadius(VariableRadius)	Index,	Установить свойство

Входные параметры:

long Index	- индекс параметров скругления.

VariableRadiusCount – Количество вершин в настройках переменных радиусов скруглений

Интерфейс...

Тип данных: long

Синтаксис Automation:

VariableRadiusCount = Object.VariableRadiusCount	=	Получить свойство (*)
VariableRadiusCount = Object.GetVariableRadiusCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_VariableRadiusCount(&VariableRadiusCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

VariableRadiusOn – Включить настройку переменных радиусов скругления

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

VariableRadiusOn = Object.VariableRadiusOn		Получить свойство (*)
Object.VariableRadiusOn = VariableRadiusOn		Установить свойство (*)
VariableRadiusOn = Object.GetVariableRadiusOn()	=	Получить свойство (**)
Object.SetVariableRadiusOn(VariableRadiusOn)		Установить свойство (**)

Синтаксис COM:

Object.get_VariableRadiusOn(&VariableRadiusOn)	Получить свойство
Object.put_VariableRadiusOn(VariableRadiusOn)	Установить свойство

VariableRadiusPosition – Величина смещения точки настройки переменного радиуса в %

Интерфейс...

Тип данных: double

Синтаксис Automation:

VariableRadiusPosition = Object.VariableRadiusPosition(Index)	=	Получить свойство (*)
Object.VariableRadiusPosition(Index) = VariableRadiusPosition	=	Установить свойство (*)
VariableRadiusPosition = Object.GetVariableRadiusPosition(Index)	=	Получить свойство (**)
Object.SetVariableRadiusPosition(Index, VariableRadiusPosition)		Установить свойство (**)

Синтаксис COM:

Object.get_VariableRadiusPosition(Index, &VariableRadiusPosition)		Получить свойство
Object.put_VariableRadiusPosition(Index, VariableRadiusPosition)		Установить свойство

Входные параметры:

long Index	- индекс параметров скругления.

VariableRadiusPositionLenght – Величина смещения точки настройки переменного радиуса в мм

Интерфейс...

Тип данных: double

Синтаксис Automation:

VariableRadiusPositionLenght = Object.VariableRadiusPositionLenght(Index)	=	Получить свойство (*)
Object.VariableRadiusPositionLenght(Index) = VariableRadiusPositionLenght	=	Установить свойство (*)
VariableRadiusPositionLenght = Object.GetVariableRadiusPositionLenght(Index)	=	Получить свойство (**)
Object.SetVariableRadiusPositionLenght(Index, VariableRadiusPositionLenght)		Установить свойство (**)

Синтаксис COM:

Object.get_VariableRadiusPositionLenght(Index, &VariableRadiusPositionLenght)		Получить свойство
Object.put_VariableRadiusPositionLenght(Index, VariableRadiusPositionLenght)		Установить свойство

Входные параметры:

long Index	- индекс параметров скругления.

VariableRadiusEdge – Ребро, для которого задана настройка переменного радиуса

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

VariableRadiusEdge Object.VariableRadiusEdge(Index)	=	Получить свойство (*)
VariableRadiusEdge Object.GetVariableRadiusEdge(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_VariableRadiusEdge(&VariableRadiusEdge)	Index,	Получить свойство

Входные параметры:

long Index	- индекс параметров скругления.

Примечание:

Свойство доступно только для чтения.

IFillet – методы

AddVariableRadius – Добавить настройку переменного радиуса для ребра в точке, заданной смещением

Интерфейс...

Синтаксис Automation:

BOOL AddVariableRadius(IModelObject * PointObject, double Radius, double Position, BOOL PositionType);

Синтаксис COM:

HRESULT AddVariableRadius(IModelObject * PointObject, double Radius, double Position, BOOL PositionType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PointObject - фиксирующий объект,
Radius - радиус скругления,
Position - положение на кривой,
PositionType - способ определения положения на кривой:
TRUE - задано величиной,
FALSE - в процентах от длины кривой.

VariableRadiusClear – Очистить список настроек переменных радиусов скруглений

Интерфейс...

Синтаксис Automation:

BOOL VariableRadiusClear();

Синтаксис COM:

HRESULT VariableRadiusClear(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

VariableRadiusDelete – Удалить настройку переменного радиуса скруглений по индексу

Интерфейс...

Синтаксис Automation:

BOOL VariableRadiusDelete(long Index);

Синтаксис COM:

HRESULT VariableRadiusDelete(long Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Index - индекс настроек.

Интерфейс IFullFillet

Интерфейс операции Полное скругление.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFullFillet

Данный интерфейс можно получить с помощью метода коллекции операций уклон IFullFillets::Add или свойства IFullFillets::FullFillet.

Версия Компас v18.1

IFullFillet – свойства

Side1Faces – Боковые грани 1

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Side1Faces = Object.Side1Faces	Получить свойство (*)
Object.Side1Faces = Side1Faces	Установить свойство (*)
Side1Faces = Object.GetSide1Faces()	Получить свойство (**)
Object.SetSide1Faces(Side1Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Side1Faces(&Side1Faces)	Получить свойство
Object.put_Side1Faces(Side1Faces)	Установить свойство

Примечание.

1. Позволяет получать и устанавливать грань или массив граней первой стороны, участвующих в операции полного скругления.
2. Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Side2Faces – Боковые грани 2

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Side2Faces = Object.Side2Faces	Получить свойство (*)
Object.Side2Faces = Side2Faces	Установить свойство (*)
Side2Faces = Object.GetSide2Faces()	Получить свойство (**)

Object.SetSide2Faces(Side2Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Side2Faces(&Side2Faces)	Получить свойство
Object.put_Side2Faces(Side2Faces)	Установить свойство

Примечание.

1. Позволяет получать и устанавливать грань или массив граней второй стороны, участвующих в операции полного скругления.
2. Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

CenterFaces – Центральные грани

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

CenterFaces = Object.CenterFaces	Получить свойство (*)
Object.CenterFaces = CenterFaces	Установить свойство (*)
CenterFaces = Object.GetCenterFaces()	Получить свойство (**)
Object.SetCenterFaces(CenterFaces)	Установить свойство (**)

Синтаксис COM:

Object.get_CenterFaces(&CenterFaces)	Получить свойство
Object.put_CenterFaces(CenterFaces)	Установить свойство

Примечание.

1. Позволяет получать и устанавливать грань или массив центральных граней, участвующих в операции полного скругления.
2. Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Tangent – По касательным граням

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Tangent = Object.Tangent	Получить свойство (*)
Object.Tangent = Tangent	Установить свойство (*)
Tangent = Object.GetTangent()	Получить свойство (**)
Object.SetTangent(Tangent)	Установить свойство (**)

Синтаксис COM:

Object.get_Tangent(&Tangent)	Получить свойство
Object.put_Tangent(Tangent)	Установить свойство

Интерфейс IIncline

Интерфейс операции Уклон.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IIncline

Интерфейс можно получить с помощью метода коллекции операций уклон IInclines::Add или свойства IInclines::Incline.

КОМПАС версия v18

IIncline – свойства

Angle – Угол уклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Direction – Направление уклона

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значение свойства:

TRUE	- наружу,
FALSE	- внутрь.

Faces – Массив уклоняемых граней

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces(Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces)	Получить свойство
Object.put_Faces(Faces)	Установить свойство

Примечание.

Позволяет получать и устанавливать грань или массив граней, участвующих в операции уклон.

Массив граней возвращается в виде массива SAFEARRAY граней LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Plane – Основание уклона

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство
Object.put_Plane(Plane)	Установить свойство

Интерфейс ILoft

Интерфейс операции По сечениям.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ILoft

Данный интерфейс можно получить с помощью метода коллекции операций по сечениям ILofts::Add или свойства ILofts::Loft. Интерфейс используется также для поверхности по сечениям.

КОМПАС версия v18

ILoft – свойства

AxisLine – Осевая линия. Ребро, кривая или зскиз, в котором лежит кривая

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

AxisLine = Object.AxisLine	Получить свойство (*)
Object.AxisLine = AxisLine	Установить свойство (*)
AxisLine = Object.GetAxisLine()	Получить свойство (**)
Object.SetAxisLine(AxisLine)	Установить свойство (**)

Синтаксис COM:

Object.get_AxisLine(&AxisLine)	Получить свойство
Object.put_AxisLine(AxisLine)	Установить свойство

BuildingObject – Направляющий объект построения элемента у крайних сечений

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

BuildingObject = Object.BuildingObject(BeginSection)	Получить свойство (*)
Object.BuildingObject(BeginSection) = BuildingObject	Установить свойство (*)
BuildingObject = Object.GetBuildingObject(BeginSection)	Получить свойство (**)
Object.SetBuildingObject(BeginSection, BuildingObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingObject(BeginSection, &BuildingObject)	Получить свойство
Object.put_BuildingObject(BeginSection, BuildingObject)	Установить свойство

Входные параметры:

BOOL BeginSection	- TRUE у начального сечения, - FALSE - у конечного сечения.

BuildingType – Способы построения элемента у крайних сечений

Интерфейс...

Тип данных: Значение из перечисления ksLoftBuildingType.

Синтаксис Automation:

BuildingType = Object.BuildingType(BeginSection)	Получить свойство (*)
Object.BuildingType(BeginSection) = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType(BeginSection)	Получить свойство (**)
Object.SetBuildingType(BeginSection, BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(BeginSection, &BuildingType)	Получить свойство
Object.put_BuildingType(BeginSection, BuildingType)	Установить свойство

Входные параметры:

BOOL BeginSection	- TRUE у начального сечения, - FALSE - у конечного сечения.

Closed – Замкнуть траекторию

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Closed = Object.Closed	Получить свойство (*)
Object.Closed = Closed	Установить свойство (*)
Closed = Object.GetClosed()	Получить свойство (**)
Object.SetClosed(Closed)	Установить свойство (**)

Синтаксис COM:

Object.get_Closed(&Closed)	Получить свойство
Object.put_Closed(Closed)	Установить свойство

Coupling – Получить параметры цепочки по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ICoupling.

Синтаксис Automation:

Coupling = Object.Coupling(Index)	Получить свойство (*)
Coupling = Object.GetCoupling(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Coupling(Index, &Coupling)	Получить свойство

Входные параметры:

long Index	- индекс цепочки.

Примечание:

Свойство доступно только для чтения.

CouplingsCount – Количество цепочек

Интерфейс...

Тип данных: long

Синтаксис Automation:

CouplingsCount = Object.CouplingsCount	=	Получить свойство (*)
CouplingsCount = Object.GetCouplingsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_CouplingsCount(&CouplingsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DirectionalLines – Направляющие кривые

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

DirectionalLines = Object.DirectionalLines		Получить свойство (*)
Object.DirectionalLines = DirectionalLines		Установить свойство (*)
DirectionalLines = Object.GetDirectionalLines())	=	Получить свойство (**)
Object.SetDirectionalLines(DirectionalLines)		Установить свойство (**)

Синтаксис COM:

Object.get_DirectionalLines(&DirectionalLines)	Получить свойство
Object.put_DirectionalLines(DirectionalLines)	Установить свойство

Примечание.

Позволяет получать и устанавливать направляющую кривую или направляющих кривых.

Массив кривых возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

OperationResult – Результат операции

Интерфейс...

Тип данных: Значение из перечисления ksOperationResultEnum.

Синтаксис Automation:

OperationResult = Object.OperationResult	Получить свойство (*)
Object.OperationResult = OperationResult	Установить свойство (*)
OperationResult = Object.GetOperationResult()	Получить свойство (**)
Object.SetOperationResult(OperationResult)	Установить свойство (**)

Синтаксис COM:

Object.get_OperationResult(&OperationResult)	Получить свойство
Object.put_OperationResult(OperationResult)	Установить свойство

Sketchs – Сечения. Эскизы, контуры, пространственные кривые, грани

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Sketchs = Object.Sketchs	Получить свойство (*)
Object.Sketchs = Sketchs	Установить свойство (*)
Sketchs = Object.GetSketchs()	Получить свойство (**)
Object.SetSketchs(Sketchs)	Установить свойство (**)

Синтаксис COM:

Object.get_Sketchs(&Sketchs)	Получить свойство
Object.put_Sketchs(Sketchs)	Установить свойство

Примечание.

Свойство позволяет получать и устанавливать массив сечений, участвующих в операции.

Массив возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ILoft – методы

AddCoupling – Добавить цепочку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddCoupling();

Синтаксис COM:

HRESULT AddCoupling(ICoupling ** Result);

Возвращаемое значение:

- указатель на интерфейс ICoupling.

ClearCouplings – Удалить все цепочки

Интерфейс...

Синтаксис Automation:

BOOL ClearCouplings();

Синтаксис COM:

HRESULT ClearCouplings(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

DeleteCoupling – Удалить цепочку по индексу

Интерфейс...

Синтаксис Automation:

BOOL DeleteCoupling(long Index);

Синтаксис COM:

HRESULT DeleteCoupling(long Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Index - индекс цепочки.

Интерфейс IRib

Интерфейс операции Ребро жесткости.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRib

Данный интерфейс можно получить с помощью метода коллекции операций ребро жесткости IRibs::Add или свойства IRibs::Rib.

КОМПАС версия v18

IRib – свойства

Angle – Уклон граней ребра. Угол уклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                    Получить свойство (* )
Object.Angle = Angle                    Установить свойство (* )
Angle = Object.GetAngle()               Получить свойство (**)
Object.SetAngle( Angle )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )              Получить свойство
Object.put_Angle(  Angle )              Установить свойство
```

Body – Тело для операции

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
Body = Object.Body                      Получить свойство (* )
Object.Body = Body                      Установить свойство (* )
Body = Object.GetBody()                 Получить свойство (**)
Object.SetBody( Body )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Body( &Body )                Получить свойство
Object.put_Body(  Body )                Установить свойство
```

SegmentIndex – Уклон граней ребра. Угол уклона. Индекс направляющего сегмента

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
SegmentIndex = Object.SegmentIndex                  Получить свойство (* )
Object.SegmentIndex = SegmentIndex                  Установить свойство (* )
SegmentIndex = Object.GetSegmentIndex()             Получить свойство (**)
Object.SetSegmentIndex( SegmentIndex )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_SegmentIndex( &SegmentIndex )            Получить свойство
Object.put_SegmentIndex(  SegmentIndex )            Установить свойство
```

Side – Положение

Интерфейс...

Тип данных: Значение из перечисления ksRibSideEnum.

Синтаксис Automation:

```
Side = Object.Side                                  Получить свойство (* )
Object.Side = Side                                  Установить свойство (* )
Side = Object.GetSide()                             Получить свойство (**)
Object.SetSide( Side )                              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Side( &Side )                            Получить свойство
Object.put_Side(  Side )                            Установить свойство
```

Sketch – Контур ребра жесткости

Интерфейс...

Тип данных: Указатель на интерфейс ISketch.

Синтаксис Automation:

```
Sketch = Object.Sketch                              Получить свойство (* )
Object.Sketch = Sketch                              Установить свойство (* )
Sketch = Object.GetSketch()                         Получить свойство (**)
Object.SetSketch( Sketch )                          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sketch( &Sketch )                        Получить свойство
Object.put_Sketch(  Sketch )                        Установить свойство
```

Интерфейс IMultiThicknessGroupsManager

Менеджер разнотолщинных групп.

Иерархия:

IDispatch

IMultiThicknessGroupsManager

Примечание

Интерфейс является дополнительным для интерфейса IShell (Оболочка),

Интерфейс позволяет настроить разную толшину для выбранных граней.

IMultiThicknessGroupsManager – свойства

MultiThick – Признак используемости разнотолщинных групп при построении операции

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MultiThick = Object.MultiThick	Получить свойство (*)
Object.MultiThick = MultiThick	Установить свойство (*)
MultiThick = Object.GetMultiThick()	Получить свойство (**)
Object.SetMultiThick(MultiThick)	Установить свойство (**)

Синтаксис COM:

Object.get_MultiThick(&MultiThick)	Получить свойство
Object.put_MultiThick(MultiThick)	Установить свойство

MultiThicknessGroupsCount – Количество групп параметров разных толщин

Интерфейс...

Тип данных: long

Синтаксис Automation:

MultiThicknessGroupsCount = Object.MultiThicknessGroupsCount(Type)	Получить свойство (*)

MultiThicknessGroupsCount = Object.GetMultiThicknessGroups Count(Type)		Получить свойство (**)

Синтаксис COM:

Object.get_MultiThicknessGroups Count(Type, &MultiThicknessGroupsCount)	Получить свойство

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum.

Примечание:

Свойство доступно только для чтения.

MultiThicknessGroupsObjects – Объекты в группе

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

MultiThicknessGroupsObjects = Object.MultiThicknessGroupsObjects(Index)	Type,	Получить свойство (*)
Object.MultiThicknessGroupsObjects(Index) = MultiThicknessGroupsObjects	Type,	Установить свойство (*)
MultiThicknessGroupsObjects = Object.GetMultiThicknessGroupsObjects(Index)	Type,	Получить свойство (**)
Object.SetMultiThicknessGroupsObjects(Index, MultiThicknessGroupsObjects)	Type,	Установить свойство (**)

Синтаксис COM:

Object.get_MultiThicknessGroupsObjects(Index, &MultiThicknessGroupsObjects)	Type,	Получить свойство
Object.put_MultiThicknessGroupsObjects(Index, MultiThicknessGroupsObjects)	Type,	Установить свойство

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum,

long Index	- Индекс параметров в группе.

MultiThicknessGroupsThickness – Толщина, заданная для группы

Интерфейс...

Тип данных: double

Синтаксис Automation:

MultiThicknessGroupsThickness = Object.MultiThicknessGroupsThickness(Type, Index)	Получить свойство (*)
Object.MultiThicknessGroupsThickness(Type, Index) = MultiThicknessGroupsThickness	Установить свойство (*)
MultiThicknessGroupsThickness = Object.GetMultiThicknessGroupsThickness(Type, Index)	Получить свойство (**)
Object.SetMultiThicknessGroupsThickness(Type, Index, MultiThicknessGroupsThickness)	Установить свойство (**)

Синтаксис COM:

Object.get_MultiThicknessGroupsThickness(Type, Index, &MultiThicknessGroupsThickness)	Получить свойство
Object.put_MultiThicknessGroupsThickness(Type, Index, MultiThicknessGroupsThickness)	Установить свойство

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum,
long Index	- Индекс параметров в группе.

MultiThicknessGroupsManager– методы

AddMultiThicknessGroup – Добавить группу объектов с заданной толщиной

Интерфейс...

Синтаксис Automation:

BOOL AddMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, VARIANT Objects, double Thickness);

Синтаксис COM :

RESULT AddMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, VARIANT Objects, double Thickness, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum,
Objects	- Список граней - массив SafeArray типа VT_ARRAY | VT_DISPATCH для нескольких граней и VT_DISPATCH для одиночной грани,
Thickness	- толщина.

DeleteMultiThicknessGroup – Удалить группу объектов с заданной толщиной

Интерфейс...

Синтаксис Automation :

BOOL DeleteMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, long Index);

Синтаксис COM :

HRESULT DeleteMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum,
Index	- Индекс параметров в группе.

DestroyMultiThicknessGroup – Разрушить группу объектов с заданной толщиной

Интерфейс...

Синтаксис Automation:

BOOL DestroyMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, long Index);

Синтаксис COM :

HRESULT DestroyMultiThicknessGroup(ksMultiThicknessGroupTypeEnum Type, long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Type	- Тип группы параметров из перечисления ksMultiThicknessGroupTypeEnum,
Index	- Индекс параметров в группе.

ExcludeMultiThicknessGroupObjects – Исключить объекты из групп

Интерфейс...

Синтаксис Automation:

BOOL ExcludeMultiThicknessGroupObjects(VARIANT Objects);

Синтаксис COM :

HRESULT ExcludeMultiThicknessGroupObjects(VARIANT Objects, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

Objects	- Грань VT_DISPATCH или список граней массив SafeArray VT_ARRAY I VT_DISPATCH.

ClearMultiThicknessGroups – Удалить все группы толщин

Интерфейс...

Синтаксис Automation:

BOOL ClearMultiThicknessGroups();

Синтаксис COM:

HRESULT ClearMultiThicknessGroups(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Интерфейс IShell

Интерфейс операции Оболочка.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IShell

Данный интерфейс можно получить с помощью метода коллекции операций оболочка IShells::Add или свойства IShells::Shell.

КОМПАС версия v18

IShell – свойства

DeletedFaces – Массив удаляемых граней в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

DeletedFaces = Object.DeletedFaces	Получить свойство (*)
Object.DeletedFaces = DeletedFaces	Установить свойство (*)
DeletedFaces = Object.GetDeletedFaces()	Получить свойство (**)
Object.SetDeletedFaces(DeletedFaces)	Установить свойство (**)

Синтаксис COM:

Object.get_DeletedFaces(&DeletedFaces)	Получить свойство
Object.put_DeletedFaces(DeletedFaces)	Установить свойство

Примечание.

Позволяет получать и устанавливать грань или массив граней, участвующих в операции оболочка.

Массив граней возвращается в виде массива SAFEARRAY граней LPDISPATCH (VT_ARRAY | VT_DISPATCH).

ThinType – Направление формирования тонкой стенки

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum.

Синтаксис Automation:

ThinType = Object.ThinType	Получить свойство (*)
Object.ThinType = ThinType	Установить свойство (*)

ThinType = Object.GetThinType()	Получить свойство (**)
Object.SetThinType(ThinType)	Установить свойство (**)

Синтаксис COM:

Object.get_ThinType(&ThinType)	Получить свойство
Object.put_ThinType(ThinType)	Установить свойство

Thickness – Толщина стенки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Thickness = Object.Thickness	Получить свойство (*)
Object.Thickness = Thickness	Установить свойство (*)
Thickness = Object.GetThickness()	Получить свойство (**)
Object.SetThickness(Thickness)	Установить свойство (**)

Синтаксис COM:

Object.get_Thickness(&Thickness)	Получить свойство
Object.put_Thickness(Thickness)	Установить свойство

Массивы

Интерфейс IFeaturePatterns

Интерфейс коллекции массивов.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IFeaturePatterns

Интерфейс можно получить у контейнера объектов 3D с помощью метода IModelContainer::FeaturePatterns.

IFeaturePatterns – свойства

FeaturePattern – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IFeaturePattern

Синтаксис Automation:

FeaturePattern = Object.FeaturePattern(Index)	Получить свойство (*)
FeaturePattern = Object.GetFeaturePattern(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_FeaturePattern(Index, &FeaturePattern)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IFeaturePatterns – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum Type);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum Type, IFeaturePattern ** Result);

Выходные параметры:

Type	- тип объекта ksObj3dTypeEnum.

Возвращаемое значение:

- указатель на интерфейс IFeaturePattern.

В зависимости от типа объекта создается операция копирования определенного типа.

Интерфейс IFeaturePattern

Справка системы КОМПАС...

kompas.chm::/Glava_48_Obzhie_svedeniy.htm

Базовый интерфейс работы с массивами.

Иерархия:

IDispatch

IFeaturePattern

Интерфейс позволяет получать и изменять общие параметры для операций копирования.

В зависимости от типа объекта используется определенный интерфейс операции копирования.

IFeaturePattern – свойства

BasePoint – Базовая точка

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

BasePoint = Object.BasePoint	Получить свойство (*)
Object.BasePoint = BasePoint	Установить свойство (*)
BasePoint = Object.GetBasePoint()	Получить свойство (**)
Object.SetBasePoint(BasePoint)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePoint(&BasePoint)	Получить свойство
Object.put_BasePoint(BasePoint)	Установить свойство

BasePointType – Способ задания базовой точки

Интерфейс...

Тип данных: из перечисления ksPatternBasePointTypeEnum

Синтаксис Automation:

BasePointType = Object.BasePointType	Получить свойство (*)
BasePointType = Object.BasePointType	Установить свойство (*)
BasePointType = Object.GetBasePointType()	Получить свойство (**)
Object.SetBasePointType(BasePointType)	Установить свойство (**)

Синтаксис COM:

Object.get_BasePointType(&BasePointType)	Получить свойство

Object.put_BasePointType(BasePointType)	Установить свойство

Exemplar – Получить экземпляр по индексу (или индексам)

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Exemplar = Object.Exemplar(index1, index2)	Получить свойство (*)
Exemplar = Object.GetExemplar(index1, index2)	Получить свойство (**)

Синтаксис COM:

Object.get_Exemplar(index1, index2, &Exemplar)	Получить свойство

Входные параметры

index1, index2 - индекс экземпляра

Примечание:

Свойство доступно только для чтения.

Значения индексов начинаются с 1.

В массиве по кривой используется один индекс (первый).

GeometryPattern – Признак использования геометрического копирования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

GeometryPattern = Object.GeometryPattern	Получить свойство (*)
Object.GeometryPattern = GeometryPattern	Установить свойство (*)
GeometryPattern = Object.GetGeometryPattern()	Получить свойство (**)
Object.SetGeometryPattern(GeometryPattern)	Установить свойство (**)

Синтаксис COM:

Object.get_GeometryPattern(&GeometryPattern)	Получить свойство
Object.put_GeometryPattern(GeometryPattern)	Установить свойство

Значения свойства:

TRUE	- используется геометрический массив,

FALSE - не используется геометрический массив.

InitialObjects – Исходные объекты массива

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

InitialObjects = Object.InitialObjects	Получить свойство (*)
Object.InitialObjects = InitialObjects	Установить свойство (*)
InitialObjects = Object.GetInitialObjects()	Получить свойство (**)
Object.SetInitialObjects(InitialObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_InitialObjects(&InitialObjects)	Получить свойство
Object.put_InitialObjects(InitialObjects)	Установить свойство

Примечание:

Если копируется один объект, то возвращается VARIANT с типом VT_DISPATCH.

Если копируется более одного объекта, то возвращается VARIANT с типом VT_ARRAY | VT_DISPATCH.

InstanceDeletedIndexes – Список удаленных экземпляров массива

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

InstanceDeletedIndexes = Object.InstanceDeletedIndexes	=	Получить свойство (*)
Object.InstanceDeletedIndexes = InstanceDeletedIndexes	=	Установить свойство (*)
InstanceDeletedIndexes = Object.GetInstanceDeletedIndexes()	=	Получить свойство (**)
Object.SetInstanceDeletedIndexes(InstanceDeletedIndexes)		Установить свойство (**)

Синтаксис COM:

Object.get_InstanceDeletedIndexes(&InstanceDeletedIndexes)	Получить свойство
Object.put_InstanceDeletedIndexes(InstanceDeletedIndexes)	Установить свойство

Примечания:

Возвращается массив SafeArray VT_ARRAY | VT_I4.

Свойства применяется для получения информации об удаленных копиях, а также управления составом удаленных копий.

Информация об удаленной копии выдается в виде индексов копии в узоре копирования.

В общем случае это два индекса по ортогональным осям в плоскости узора.

Для копий по кривой используется один индекс (первый).

Значения индексов начинаются с 1.

IFeaturePattern – методы

AddInitialObjects – Добавить объекты в массив копируемых объектов

Интерфейс...

Синтаксис Automation:

BOOL AddInitialObjects(VARIANT Objects);

Синтаксис COM:

HRESULT AddInitialObjects(VARIANT Objects, BOOL * Result);

Входные параметры:

Object	- объект - тип VARIANT-а VT_DISPATCH или список объектов - тип VARIANT-а VT_ARRAY | VT_DISPATCH.

Возвращаемое значение:

TRUE	- в случае удачи.

Clear – Очистить список исходных объектов массива

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачи.

Destroy – Разрушить массив

Интерфейс...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

HRESULT Destroy(BOOL * Result);

Возвращаемое значение:

TR - в случае удачи.
UE

GetExemplarsCounts – Получить количество экземпляров

Интерфейс...

Синтаксис Automation:

BOOL GetExemplarsCounts(long * Count1, long * Count2);

Синтаксис COM:

HRESULT GetExemplarsCounts(long * Count1, long * Count2, BOOL * Result);

Выходные параметры:

Count1 - количество экземпляров в первом направлении копирования,
Count2 - количество экземпляров во втором направлении копирования.

Возвращаемое значение:

TRUE - в случае удачи.

Примечание:

Два направления копирования имеют массивы по сетке и по концентрической сетке.

Для массивов остальных типов Count2 равен 1.

IsInitialObject – Проверка: является ли объект исходным для массива

Интерфейс...

Синтаксис Automation:

BOOL IsInitialObject(IModelObject * Object);

Синтаксис COM:

HRESULT IsInitialObject(IModelObject * Object, BOOL * Result);

Входные параметры:

Object	- указатель на объект IModelObject.

Возвращаемое значение:

TRUE	- если объект есть в списке копируемых объектов.

IsSuitableObject – Проверка пригодности объекта для операции

Интерфейс...

Синтаксис Automation:

BOOL IsSuitableObject(IModelObject * Object);

Синтаксис COM:

HRESULT IsSuitableObject(IModelObject * Object, BOOL * Result);

Входные параметры:

Object	- указатель на объект IModelObject.

Возвращаемое значение:

TRUE	- указанный объект можно скопировать с помощью данной операции.

Интерфейс ILinearPattern

Справка системы КОМПАС...

`kompas.chm::/Glava_49_Massiv_po_setke.htm`

Интерфейс параметров операции копирования по сетке.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IFeaturePattern
      - ILinearPattern

Интерфейс используется для операций:

o3d_meshCopy	35	Массив операций по сетке
o3d_meshPartArray	39	Массив компонентов по сетке для сборки

o3d_AuxMeshCopy	504	Массив вспомогательной геометрии по сетке
o3d_BodiesMeshCopy	528	Массив тел по сетке

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

ILinearPattern – свойства

Angle1 – Угол наклона первой оси сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle1 = Object.Angle1	Получить свойство (*)
Object.Angle1 = Angle1	Установить свойство (*)
Angle1 = Object.GetAngle1()	Получить свойство (**)
Object.SetAngle1(Angle1)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle1(&Angle1)	Получить свойство
Object.put_Angle1(Angle1)	Установить свойство

Angle2 – Угол наклона второй оси сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle2 = Object.Angle2	Получить свойство (*)
Object.Angle2 = Angle2	Установить свойство (*)
Angle2 = Object.GetAngle2()	Получить свойство (**)
Object.SetAngle2(Angle2)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle2(&Angle2)	Получить свойство
Object.put_Angle2(Angle2)	Установить свойство

Axis1 – Первая ось сетки массива

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Axis1 = Object.Axis1	Получить свойство (*)
Object.Axis1 = Axis1	Установить свойство (*)
Axis1 = Object.GetAxis1()	Получить свойство (**)
Object.SetAxis1(Axis1)	Установить свойство (**)

Синтаксис COM:

Object.get_Axis1(&Axis1)	Получить свойство
Object.put_Axis1(Axis1)	Установить свойство

Axis2 – Вторая ось сетки массива

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Axis2 = Object.Axis2	Получить свойство (*)
Object.Axis2 = Axis2	Установить свойство (*)
Axis2 = Object.GetAxis2()	Получить свойство (**)
Object.SetAxis2(Axis2)	Установить свойство (**)

Синтаксис COM:

Object.get_Axis2(&Axis2)	Получить свойство
Object.put_Axis2(Axis2)	Установить свойство

BoundaryInstancesStepFactor1– Интерпретация значения шага копирования вдоль первой оси сетки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BoundaryInstancesStepFactor1	=	Получить свойство (*)
Object.BoundaryInstancesStepFactor1		
Object.BoundaryInstancesStepFactor1	=	Установить свойство (*)
BoundaryInstancesStepFactor1		
BoundaryInstancesStepFactor1	=	Получить свойство (**)
Object.GetBoundaryInstancesStepFactor1()		
Object.SetBoundaryInstancesStepFactor1(		Установить свойство (**)
BoundaryInstancesStepFactor1)		

Синтаксис COM:

Object.get_BoundaryInstancesStepFactor1(&BoundaryInstancesStepFactor1)	Получить свойство
Object.put_BoundaryInstancesStepFactor1(BoundaryInstancesStepFactor1)	Установить свойство

Значения свойства:

TRUE	- шаг – расстояние между крайними экземплярами,
FALSE	- шаг – расстояние между соседними экземплярами.

BoundaryInstancesStepFactor2 – Интерпретация значения шага копирования вдоль второй оси сетки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BoundaryInstancesStepFactor2 Object.BoundaryInstancesStepFactor2	=	Получить свойство (*)
Object.BoundaryInstancesStepFactor2 BoundaryInstancesStepFactor2	=	Установить свойство (*)
BoundaryInstancesStepFactor2 Object.GetBoundaryInstancesStepFactor2()	=	Получить свойство (**)
Object.SetBoundaryInstancesStepFactor2(BoundaryInstancesStepFactor2)		Установить свойство (**)

Синтаксис COM:

Object.get_BoundaryInstancesStepFactor2(&BoundaryInstancesStepFactor2)	Получить свойство
Object.put_BoundaryInstancesStepFactor2(BoundaryInstancesStepFactor2)	Установить свойство

Значения свойства:

TRUE - шаг – расстояние между крайними экземплярами,

FALSE - шаг – расстояние между соседними экземплярами.

BuildingType – Способ построения массива

Интерфейс...

Тип данных: из перечисления ksLinearPatternBuildingTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Count1 – Количество экземпляров по первой оси сетки

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count1 = Object.Count1	Получить свойство (*)
Object.Count1 = Count1	Установить свойство (*)
Count1 = Object.GetCount1()	Получить свойство (**)
Object.SetCount1(Count1)	Установить свойство (**)

Синтаксис COM:

Object.get_Count1(&Count1)	Получить свойство
Object.put_Count1(Count1)	Установить свойство

Count2 – Количество экземпляров по второй оси сетки

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count2 = Object.Count2	Получить свойство (*)
Object.Count2 = Count2	Установить свойство (*)
Count2 = Object.GetCount2()	Получить свойство (**)
Object.SetCount2(Count2)	Установить свойство (**)

Синтаксис COM:

Object.get_Count2(&Count2)	Получить свойство
Object.put_Count2(Count2)	Установить свойство

Direction1 – Направление копирования вдоль первой оси сетки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction1 = Object.Direction1	Получить свойство (*)
Object.Direction1 = Direction1	Установить свойство (*)
Direction1 = Object.GetDirection1()	Получить свойство (**)
Object.SetDirection1(Direction1)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction1(&Direction1)	Получить свойство
Object.put_Direction1(Direction1)	Установить свойство

Direction2 – Направление копирования вдоль второй оси сетки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction2 = Object.Direction2	Получить свойство (*)
Object.Direction2 = Direction2	Установить свойство (*)
Direction2 = Object.GetDirection2()	Получить свойство (**)
Object.SetDirection2(Direction2)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction2(&Direction2)	Получить свойство
Object.put_Direction2(Direction2)	Установить свойство

Step1 – Шаг копирования по первой оси сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step1 = Object.Step1	Получить свойство (*)
Object.Step1 = Step1	Установить свойство (*)
Step1 = Object.GetStep1()	Получить свойство (**)
Object.SetStep1(Step1)	Установить свойство (**)

Синтаксис COM:

Object.get_Step1(&Step1)	Получить свойство
Object.put_Step1(Step1)	Установить свойство

Step2 – Шаг копирования по второй оси сетки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step2 = Object.Step2	Получить свойство (*)
Object.Step2 = Step2	Установить свойство (*)
Step2 = Object.GetStep2()	Получить свойство (**)
Object.SetStep2(Step2)	Установить свойство (**)

Синтаксис COM:

Object.get_Step2(&Step2)	Получить свойство
Object.put_Step2(Step2)	Установить свойство

Vector1 – Параметры вектора, задающего направление первой оси сетки

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector1 = Object.Vector1	Получить свойство (*)
Vector1 = Object.GetVector1()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector1(&Vector1)	Получить свойство

Примечания:

Свойство доступно только для чтения.

Vector2 – Параметры вектора, задающего направление второй оси сетки

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector2 = Object.Vector2	Получить свойство (*)
Vector2 = Object.GetVector2()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector2(&Vector2)	Получить свойство

Примечания:

Свойство доступно только для чтения.

ILinearPattern – методы

GetBaseExemplarPlacement – Получить положение базового экземпляра в массиве

Интерфейс...

Синтаксис Automation:

BOOL GetBaseExemplarPlacement(long Index1, long Index2);

Синтаксис COM:

HRESULT GetBaseExemplarPlacement(long * Index1, long * Index2, BOOL * Result);

Входные параметры:

index1	- индекс экземпляра по 1-й оси,
index2	- индекс экземпляра по 2-й оси.

Возвращаемое значение:

TRUE	- в случае удачи.

SetBaseExemplarPlacement – Задать положение базового экземпляра в массиве

Интерфейс...

Синтаксис Automation:

BOOL SetBaseExemplarPlacement(long Index1, long Index2);

Синтаксис COM:

HRESULT SetBaseExemplarPlacement(long Index1, long Index2, BOOL * Result);

Входные параметры:

index1	- индекс экземпляра по 1-й оси,
index2	- индекс экземпляра по 2-й оси.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс ICircularPattern

Справка системы КОМПАС...

`kompas.chm::/Glava_50_Massiv_po_koncentr_setke.htm`

Интерфейс параметров операции копирования по концентрической сетке.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IFeaturePattern
      - ICircularPattern

Интерфейс используется для операций:

o3d_circularCopy	36	Массив по концентрической сетке
o3d_circPartArray	38	Массив компонентов по концентрической сетке для сборки
o3d_AuxCircularCopy	50	Массив вспомогательной геометрии по концентрической сетке
o3d_BodiesCircularCopy	52	Массив тел по концентрической сетке
	9	

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

ICircularPattern – свойства

Axis – Ось массива

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Axis = Object.Axis	Получить свойство (*)
Object.Axis = Axis	Установить свойство (*)
Axis = Object.GetAxis()	Получить свойство (**)
Object.SetAxis(Axis)	Установить свойство (**)

Синтаксис COM:

Object.get_Axis(&Axis)	Получить свойство
Object.put_Axis(Axis)	Установить свойство

BoundaryInstancesStepFactor1 – Интерпретация значения шага копирования в радиальном направлении

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BoundaryInstancesStepFactor1 = Object.BoundaryInstancesStepFactor1	=	Получить свойство (*)
Object.BoundaryInstancesStepFactor1 = BoundaryInstancesStepFactor1	=	Установить свойство (*)
BoundaryInstancesStepFactor1 = Object.GetBoundaryInstancesStepFactor1()	=	Получить свойство (**)
Object.SetBoundaryInstancesStepFactor1(BoundaryInstancesStepFactor1)		Установить свойство (**)

Синтаксис COM:

Object.get_BoundaryInstancesStepFactor1(&BoundaryInstancesStepFactor1)	Получить свойство
Object.put_BoundaryInstancesStepFactor1(BoundaryInstancesStepFactor1)	Установить свойство

Значения свойства:

TRUE	- шаг – расстояние между крайними экземплярами,

FALSE	- шаг – расстояние между соседними экземплярами.

BoundaryInstancesStepFactor2 – Интерпретация значения шага копирования в кольцевом направлении

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BoundaryInstancesStepFactor2 = Object.BoundaryInstancesStepFactor2	=	Получить свойство (*)
Object.BoundaryInstancesStepFactor2 = BoundaryInstancesStepFactor2	=	Установить свойство (*)
BoundaryInstancesStepFactor2 = Object.GetBoundaryInstancesStepFactor2()	=	Получить свойство (**)
Object.SetBoundaryInstancesStepFactor2(BoundaryInstancesStepFactor2)		Установить свойство (**)

Синтаксис COM:

Object.get_BoundaryInstancesStepFactor2(&BoundaryInstancesStepFactor2)	Получить свойство
Object.put_BoundaryInstancesStepFactor2(BoundaryInstancesStepFactor2)	Установить свойство

Значения свойства:

TRUE	- шаг – угол между крайними экземплярами,
FALSE	- шаг – угол между соседними экземплярами.

BuildingType – Способ построения массива

Интерфейс...

Тип данных: ksCircularPatternBuildingTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Count1 – Количество экземпляров в радиальном направлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count1 = Object.Count1	Получить свойство (*)
Object.Count1 = Count1	Установить свойство (*)
Count1 = Object.GetCount1()	Получить свойство (**)
Object.SetCount1(Count1)	Установить свойство (**)

Синтаксис COM:

Object.get_Count1(&Count1)	Получить свойство
Object.put_Count1(Count1)	Установить свойство

Count2 – Количество экземпляров в кольцевом направлении

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count2 = Object.Count2	Получить свойство (*)
Object.Count2 = Count2	Установить свойство (*)
Count2 = Object.GetCount2()	Получить свойство (**)
Object.SetCount2(Count2)	Установить свойство (**)

Синтаксис COM:

Object.get_Count2(&Count2)	Получить свойство
Object.put_Count2(Count2)	Установить свойство

ReverseDirection – Направление построения массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReverseDirection = Object.ReverseDirection	Получить свойство (*)
Object.ReverseDirection = ReverseDirection	Установить свойство (*)
ReverseDirection = Object.GetReverseDirection()	Получить свойство (**)

Object.SetReverseDirection(ReverseDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ReverseDirection(&ReverseDirection)	Получить свойство
Object.put_ReverseDirection(ReverseDirection)	Установить свойство

Значения свойства:

TRUE	- обратное направление,
FALSE	- прямое направление.

SaveInitialOrientation – Ориентация экземпляров массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveInitialOrientation	=	Получить свойство (*)
Object.SaveInitialOrientation		
Object.SaveInitialOrientation	=	Установить свойство (*)
SaveInitialOrientation		
SaveInitialOrientation	=	Получить свойство (**)
Object.GetSaveInitialOrientation()		
Object.SetSaveInitialOrientation(		Установить свойство (**)
SaveInitialOrientation)		

Синтаксис COM:

Object.get_SaveInitialOrientation(&SaveInitialOrientation)	Получить свойство
Object.put_SaveInitialOrientation(SaveInitialOrientation)	Установить свойство

Значения свойства:

TRUE	- сохранять исходную ориентацию,
FALSE	- доворачивать до радиального направления.

Step1 – Шаг копирования в радиальном направлении

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step1 = Object.Step1                    Получить свойство (* )
Object.Step1 = Step1                    Установить свойство (* )
Step1 = Object.GetStep1()               Получить свойство (**)
Object.SetStep1( Step1 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step1( &Step1 )              Получить свойство
Object.put_Step1(  Step1 )              Установить свойство
```

Step2 – Угловой шаг (градусы) – шаг копирования в кольцевом направлении

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Step2 = Object.Step2                    Получить свойство (* )
Object.Step2 = Step2                    Установить свойство (* )
Step2 = Object.GetStep2()               Получить свойство (**)
Object.SetStep2( Step2 )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Step2( &Step2 )              Получить свойство
Object.put_Step2(  Step2 )              Установить свойство
```

StepByAxis – Шаг вдоль оси

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
StepByAxis = Object.StepByAxis          Получить свойство (* )
Object.StepByAxis = StepByAxis          Установить свойство (* )
StepByAxis = Object.GetStepByAxis()     Получить свойство (**)
Object.SetStepByAxis( StepByAxis )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_StepByAxis( &StepByAxis )    Получить свойство
Object.put_StepByAxis(  StepByAxis )    Установить свойство
```

ICircularPattern – методы

GetBaseExemplarPlacement – Получить положение базового экземпляра в массиве

Интерфейс...

Синтаксис Automation:

BOOL GetBaseExemplarPlacement(long Index1, long Index2);

Синтаксис COM:

HRESULT GetBaseExemplarPlacement(long * Index1, long * Index2, BOOL * Result);

Входные параметры:

index1	- индекс экземпляра по радиальному направлению,
index2	- индекс экземпляра по кольцевому направлению.

Возвращаемое значение:

TRUE	- в случае удачи.

SetBaseExemplarPlacement – Задать положение базового экземпляра в массиве

Интерфейс...

Синтаксис Automation:

BOOL SetBaseExemplarPlacement(long Index1, long Index2);

Синтаксис COM:

HRESULT SetBaseExemplarPlacement(long Index1, long Index2, BOOL * Result);

Входные параметры:

index1	- индекс экземпляра по радиальному направлению,
index2	- индекс экземпляра по кольцевому направлению.

Возвращаемое значение:

TRUE	- в случае удачи.

Интерфейс IPathPattern

Справка системы КОМПАС...

kompas.chm::/Glava_51_Massiv_vdol_krivoy.htm

Интерфейс параметров операции копирования вдоль кривой.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFeaturePattern

IPathPattern

Интерфейс используется для операций:

o3d_curveCopy	37	Массив операций по кривой
o3d_curvePartArray	40	Массив компонентов по кривой для сборки
o3d_AuxCurveCopy	506	Массив вспомогательной геометрии по кривой
o3d_BodiesCurveCopy	530	Массив тел по кривой

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

IPathPattern – свойства

BoundaryInstancesStepFactor – Интерпретация значения шага

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BoundaryInstancesStepFactor = Object.BoundaryInstancesStepFactor	=	Получить свойство (*)
Object.BoundaryInstancesStepFactor = BoundaryInstancesStepFactor	=	Установить свойство (*)
BoundaryInstancesStepFactor = Object.GetBoundaryInstancesStepFactor()	=	Получить свойство (**)
Object.SetBoundaryInstancesStepFactor(BoundaryInstancesStepFactor)		Установить свойство (**)

Синтаксис COM:

Object.get_BoundaryInstancesStepFactor(&BoundaryInstancesStepFactor)	Получить свойство

Object.put_BoundaryInstancesStepFactor(BoundaryInstancesStepFactor)	Установить свойство

Значения свойства:

TRUE	- шаг – расстояние между крайними экземплярами,
FALSE	- шаг – расстояние между соседними экземплярами.

ByStep – Способ построения массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ByStep = Object.ByStep	Получить свойство (*)
Object.ByStep = ByStep	Установить свойство (*)
ByStep = Object.GetByStep()	Получить свойство (**)
Object.SetByStep(ByStep)	Установить свойство (**)

Синтаксис COM:

Object.get_ByStep(&ByStep)	Получить свойство
Object.put_ByStep(ByStep)	Установить свойство

Значения свойства:

TRUE	- по шагу,
FALSE	- вдоль всей направляющей.

Count – Количество экземпляров

Интерфейс...

Тип данных: long

Синтаксис Automation:

Count = Object.Count	Получить свойство (*)
Object.Count = Count	Установить свойство (*)
Count = Object.GetCount()	Получить свойство (**)
Object.SetCount(Count)	Установить свойство (**)

Синтаксис COM:

Object.get_Count(&Count)	Получить свойство
Object.put_Count(Count)	Установить свойство

Curves – Массив кривых – траектория копирования

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Curves = Object.Curves	Получить свойство (*)
Object.Curves = Curves	Установить свойство (*)
Curves = Object.GetCurves()	Получить свойство (**)
Object.SetCurves(Curves)	Установить свойство (**)

Синтаксис COM:

Object.get_Curves(&Curves)	Получить свойство
Object.get_Curves(&Curves)	Установить свойство

Примечание:

Если используется одна кривая, то тип VARIANT-а VT_DISPATCH.

Если используется несколько кривых, то тип VARIANT-а VT_ARRAY | VT_DISPATCH.

ReverseDirection – Направление построения массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReverseDirection = Object.ReverseDirection	Получить свойство (*)
Object.ReverseDirection = ReverseDirection	Установить свойство (*)
ReverseDirection = Object.GetReverseDirection()	Получить свойство (**)
Object.SetReverseDirection(ReverseDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_ReverseDirection(&ReverseDirection)	Получить свойство
Object.put_ReverseDirection(ReverseDirection)	Установить свойство

Значения свойства:

TRUE	- обратное направление,
FALSE	- прямое направление.

SaveInitialOrientation – Ориентация экземпляров массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveInitialOrientation = Object.SaveInitialOrientation	=	Получить свойство (*)
Object.SaveInitialOrientation = SaveInitialOrientation	=	Установить свойство (*)
SaveInitialOrientation = Object.GetSaveInitialOrientation()	=	Получить свойство (**)
Object.SetSaveInitialOrientation(SaveInitialOrientation)		Установить свойство (**)

Синтаксис COM:

Object.get_SaveInitialOrientation(&SaveInitialOrientation)	Получить свойство
Object.put_SaveInitialOrientation(SaveInitialOrientation)	Установить свойство

Значения свойства:

TRUE	- сохранять исходную ориентацию,
FALSE	- доворачивать до нормали к траектории.

StartingPoint – Начальная точка для замкнутой траектории

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

StartingPoint = Object.StartingPoint	Получить свойство (*)
Object.StartingPoint = StartingPoint	Установить свойство (*)
StartingPoint = Object.GetStartingPoint()	Получить свойство (**)
Object.SetStartingPoint(StartingPoint)	Установить свойство (**)

Синтаксис COM:

Object.get_StartingPoint(&StartingPoint)	Получить свойство
Object.put_StartingPoint(StartingPoint)	Установить свойство

Step – Значение шага (измеряется вдоль траектории)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Step = Object.Step	Получить свойство (*)
Object.Step = Step	Установить свойство (*)
Step = Object.GetStep()	Получить свойство (**)
Object.SetStep(Step)	Установить свойство (**)

Синтаксис COM:

Object.get_Step(&Step)	Получить свойство
Object.put_Step(Step)	Установить свойство

Интерфейс IDerivedPattern

Справка системы КОМПАС...

kompas.chm::/Glava_56_Massiv_po_obrazcu.htm

Интерфейс параметров массива компонентов по образцу.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IFeaturePattern
      - IDerivedPattern

Интерфейс используется для операций:

o3d_derivPartArray	41	Массив компонентов по образцу для сборки

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

IDerivedPattern – свойства

AllowDeleted – Учитывать удаленные

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AllowDeleted = Object.AllowDeleted	Получить свойство (*)
Object.AllowDeleted = AllowDeleted	Установить свойство (*)
AllowDeleted = Object.GetAllowDeleted()	Получить свойство (**)
Object.SetAllowDeleted(AllowDeleted)	Установить свойство (**)

Синтаксис COM:

Object.get_AllowDeleted(&AllowDeleted)	Получить свойство
Object.put_AllowDeleted(AllowDeleted)	Установить свойство

AllowNesting – Учитывать вложенность

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AllowNesting = Object.AllowNesting	Получить свойство (*)
Object.AllowNesting = AllowNesting	Установить свойство (*)
AllowNesting = Object.GetAllowNesting()	Получить свойство (**)
Object.SetAllowNesting(AllowNesting)	Установить свойство (**)

Синтаксис COM:

Object.get_AllowNesting(&AllowNesting)	Получить свойство
Object.put_AllowNesting(AllowNesting)	Установить свойство

MasterPattern – Массив – образец

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

MasterPattern = Object.MasterPattern	Получить свойство (*)
Object.MasterPattern = MasterPattern	Установить свойство (*)
MasterPattern = Object.GetMasterPattern()	Получить свойство (**)
Object.SetMasterPattern(MasterPattern)	Установить свойство (**)

Синтаксис COM:

Object.get_MasterPattern(&MasterPattern)	Получить свойство
Object.put_MasterPattern(MasterPattern)	Установить свойство

OrientBySample – Ориентировать по образцу

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

OrientBySample = Object.OrientBySample	Получить свойство (*)
Object.OrientBySample = OrientBySample	Установить свойство (*)
OrientBySample = Object.GetOrientBySample()	Получить свойство (**)
Object.SetOrientBySample(OrientBySample)	Установить свойство (**)

Синтаксис COM:

Object.get_OrientBySample(&OrientBySample)	Получить свойство
Object.put_OrientBySample(OrientBySample)	Установить свойство

SampleExemplar – Экземпляр массива-образца, относительно которого будет копироваться положение

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

SampleExemplar = Object.SampleExemplar	Получить свойство (*)
Object.SampleExemplar = SampleExemplar	Установить свойство (*)
OrientBySample = Object.GetSampleExemplar()	Получить свойство (**)
Object.SetSampleExemplar(SampleExemplar)	Установить свойство (**)

Синтаксис COM:

Object.get_SampleExemplar(&SampleExemplar)	Получить свойство
Object.put_SampleExemplar(SampleExemplar)	Установить свойство

Интерфейс IMirrorPattern

Справка системы КОМПАС...

`kompas.chm::/Glava_55_Zerkalniy_massiv.htm`

Интерфейс параметров операции зеркального копирования.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IFeaturePattern
      - IMirrorPattern

Интерфейс используется для операций:

o3d_mirrorOperation	48	Зеркальный массив операций

o3d_mirrorAllOperation	49	Операция «зеркально отразить тело или поверхность»; дополнительно имеет интерфейс выбора тел IChooseBodies7
o3d_AuxMirrorOperation	509	Зеркальный массив вспомогательной геометрии

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

IMirrorPattern – свойства

Plane – Плоскость симметрии

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство
Object.put_Plane(Plane)	Установить свойство

SaveInitialObjects – Признак того, что исходные объекты остаются (не удаляются)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveInitialObjects = Object.SaveInitialObjects	Получить свойство (*)
Object.SaveInitialObjects = SaveInitialObjects	Установить свойство (*)
SaveInitialObjects = Object.GetSaveInitialObjects()	Получить свойство (**)
Object.SetSaveInitialObjects(SaveInitialObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveInitialObjects(&SaveInitialObjects)	Получить свойство
Object.put_SaveInitialObjects(SaveInitialObjects)	Установить свойство

Значение свойства:

T - оставлять исходные объекты.
R
U
E

Примечание:

1. Свойство работает только для o3d_mirrorAllOperation.
2. У других операций зеркального копирования возможность скрыть экземпляры отсутствует.

Интерфейс IPointDrivenPattern

Справка системы КОМПАС...

kompas.chm::/Glava_52_Massiv_po_tockam.htm

Интерфейс параметров операции копирования по точкам.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IFeaturePattern
IPointDrivenPattern

Позволяет управлять параметрами операций копирования по точкам.

IPointDrivenPattern – свойства

DrivenObjects –Управляющие объекты для копирования

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

DrivenObjects = Object.DrivenObjects		Получить свойство (*)
Object.DrivenObjects = DrivenObjects		Установить свойство (*)
DrivenObjects	=	Получить свойство (**)
Object.GetDrivenObjects()		
Object.SetDrivenObjects(DrivenObjects)		Установить свойство (**)

Синтаксис COM:

Object.get_DrivenObjects(&DrivenObjects)	Получить управляющие объекты
Object.put_DrivenObjects(DrivenObjects)	Установить управляющие объекты

Позволяет добавлять в список управляющих объектов эскизы, группы точек, массивы точек, точки, а также получать список добавленных объектов.

OrientationObject – Ориентирующий объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

OrientationObject = Object.OrientationObject		Получить свойство (*)
Object.OrientationObject = OrientationObject		Установить свойство (*)
OrientationObject = Object.GetOrientationObject()		Получить свойство (**)
Object.SetOrientationObject(OrientationObject)		Установить свойство (**)

Синтаксис COM:

Object.get_OrientationObject(&OrientationObject)		Получить признак проецирования
Object.put_OrientationObject( )	OrientationObject	Установить признак проецирования

OrientationType – Способ ориентации экземпляров массива

Интерфейс...

Тип данных: из перечисления ksPatternExemplarsOrientationTypeEnum

Синтаксис Automation:

OrientationType = Object.OrientationType		Получить свойство (*)
Object.OrientationType = OrientationType		Установить свойство (*)
OrientationType = Object.GetOrientationType()		Получить свойство (**)
Object.SetOrientationType(OrientationType)		Установить свойство (**)

Синтаксис COM:

Object.get_OrientationType(&OrientationType)	Получить признак проецирования
Object.put_OrientationType(OrientationType)	Установить признак проецирования

ProjectionPoints – Признак проецирования управляющих точек эскиза

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ProjectionPoints Object.ProjectionPoints	=	Получить свойство (*)
Object.ProjectionPoints ProjectionPoints	=	Установить свойство (*)
ProjectionPoints Object.GetProjectionPoints()	=	Получить свойство (**)
Object.SetProjectionPoints(ProjectionPoints)		Установить свойство (**)

Синтаксис COM:

Object.get_ProjectionPoints(&ProjectionPoints)	Получить признак проецирования
Object.put_ProjectionPoints(ProjectionPoints)	Установить признак проецирования

Значения свойства:

TRUE	- проецировать точки,
FALSE	- не проецировать точки.

Позволяет получать признак проецирования управляющих точек эскиза на плоскость, параллельную плоскости эскиза и проходящую через базовую точку.

SaveInitialOrientation – Ориентация экземпляров массива (TRUE – Сохранять исходную ориентацию, иначе – доворачивать до нормали кривой или поверхности)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveInitialOrientation Object.SaveInitialOrientation	=	Получить свойство (*)
Object.SaveInitialOrientation SaveInitialOrientation	=	Установить свойство (*)
SaveInitialOrientation Object.GetSaveInitialOrientation()	=	Получить свойство (**)
Object.SetSaveInitialOrientation(SaveInitialOrientation)		Установить свойство (**)

Синтаксис COM:

Object.get_SaveInitialOrientation(&SaveInitialOrientation)	Получить свойство
Object.put_SaveInitialOrientation(SaveInitialOrientation)	Установить свойство

Свойство позволяет устанавливать и получать ориентацию экземпляров массива:

IPointDrivenPattern – методы

ClearDrivenObjects – Очистить список управляющих объектов

Интерфейс...

Синтаксис Automation:

BOOL ClearDrivenObjects();

Синтаксис COM:

HRESULT ClearDrivenObjects(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод очищает список управляющих объектов.

IsSuitableDrivenObject – Проверка пригодности управляющего объекта для операции

Интерфейс...

Синтаксис Automation:

BOOL IsSuitableDrivenObject(IModelObject * Object);

Синтаксис COM:

HRESULT IsSuitableDrivenObject(IModelObject * Object, BOOL * Result);

Входные параметры:

Object	- управляющий объект для операции.

Возвращаемое значение:

TRUE	- можно добавлять объект Object,
FALSE	- нельзя добавлять обЪект Object.

Примечание:

Метод проверяет объект Object на возможность использования в качестве управляющего объекта операции.

Интерфейс ITablePattern

Справка системы КОМПАС...

`kompas.chm::/Glava_53_Massiv_po_tablice.htm`

Интерфейс параметров массива по таблице.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IFeaturePattern

ITablePattern

ILocalCSObject

Интерфейс используется для операций:

o3d_TablePattern	522	Массив операций по таблице из файла
o3d_PartsTablePattern	523	Массив компонентов по таблице из файла
o3d_AuxTablePattern	524	Массив вспомогательной геометрии по таблице из файла
o3d_BodiesTablePattern	525	Массив тел по таблице из файла

Интерфейс можно получить в коллекции массивов IFeaturePatterns.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

ITablePattern - свойства

FileName - Файл-источник координат точек

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

FileName = Object.FileName	Получить свойство (*)
Object.FileName = FileName	Установить свойство (*)
FileName = Object.GetFileName()	Получить свойство (**)
Object.SetFileName(FileName)	Установить свойство (**)

Синтаксис COM:

Object.get_FileName(&FileName)	Получить свойство
Object.put_FileName(FileName)	Установить свойство

Свойство позволяет устанавливать и получать имя файла с таблицей координат.

OrientationObject – Ориентирующий объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

OrientationObject = Object.OrientationObject	Получить свойство (*)
Object.OrientationObject = OrientationObject	Установить свойство (*)
OrientationObject = Object.GetOrientationObject()	Получить свойство (**)
Object.SetOrientationObject(OrientationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_OrientationObject(&OrientationObject)	Получить свойство
Object.put_OrientationObject(OrientationObject)	Установить свойство

OrientationType – Способ ориентации экземпляров массива

Интерфейс...

Тип данных: из перечисления ksPatternExemplarsOrientationTypeEnum

Синтаксис Automation:

OrientationType = Object.OrientationType	Получить свойство (*)
Object.OrientationType = OrientationType	Установить свойство (*)
OrientationType = Object.GetOrientationType()	Получить свойство (**)
Object.SetOrientationType(OrientationType)	Установить свойство (**)

Синтаксис COM:

Object.get_OrientationType(&OrientationType)	Получить свойство
Object.put_OrientationType(OrientationType)	Установить свойство

PointsType – Тип координат в файле-источнике

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointsType = Object.PointsType	Получить свойство (*)
Object.PointsType = PointsType	Установить свойство (*)
PointsType = Object.GetPointsType()	Получить свойство (**)
Object.SetPointsType(PointsType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointsType(&PointsType)	Получить свойство
Object.put_PointsType(PointsType)	Установить свойство

SaveInitialOrientation – Ориентация экземпляров массива

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SaveInitialOrientation = Object.SaveInitialOrientation	Получить свойство (*)
Object.SaveInitialOrientation = SaveInitialOrientation	Установить свойство (*)
SaveInitialOrientation = Object.GetSaveInitialOrientation()	Получить свойство (**)
Object.SetSaveInitialOrientation(SaveInitialOrientation)	Установить свойство (**)

Синтаксис COM:

Object.get_SaveInitialOrientation(&SaveInitialOrientation)	Получить свойство
Object.put_SaveInitialOrientation(SaveInitialOrientation)	Установить свойство

Значения свойства:

TR UE	- сохранять исходную ориентацию,
FAL SE	- доворачивать до нормали.

Точки

Интерфейс IPoints3D

Справка системы КОМПАС...

`kompas.chm::/866_Glava102_Prostranstvennye_k.htm`

Интерфейс коллекции элементов Пространственная точка.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IPoints3D

Описание:

Интерфейс позволяет получить коллекции элементов Пространственная точка.

Примечание:

Получить интерфейс коллекции можно используя свойство контейнера трехмерных объектов IModelContainer::Points3D.

IPoints3D – свойства

Point3D – Элемент, заданный по индексу

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/866_Glava102_Prostranstvennye_k.htm`

Тип данных: указатель на интерфейс IPoint3D

Синтаксис Automation:

Point3D = iObject.Point3D(index)	Получить свойство (*)
Point3D = iObject.GetPoint3D(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Point3D(index, &Point3D)	Получить свойство

Входные параметры:

Index	VT_BSTR - имя объекта, VT_I4 - индекс объекта.

Примечание:

Свойство доступно только для чтения.

IPoints3D – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/866_Glava102_Prostranstvennye_k.htm

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IPoint3D** Result);

Возвращаемое значение:

Указатель на интерфейс пространственной точки IPoint3D.

Примечание:

Метод позволяет создать новый интерфейс пространственной точки. После получения нового интерфейса нужно задать параметры элемента и вызвать метод IModelObject::Update.

Интерфейс IPoint3D

Справка системы КОМПАС...

kompas.chm::/866_Glava102_Prostranstvennye_k.htm

Интерфейс пространственной точки.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPoint3D

ILocalCSObject

Примечание:

Интерфейс можно получить с помощью метода коллекции пространственных точек IPoints3D::Add или свойства IPoints3D::Point3D.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IPoint3D – свойства

AssociationObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

AssociationObject = iObject.AssociationObject; Получить свойство (*)

AssociationObject = iObject.GetAssociationObject();	Получить свойство (**)

Синтаксис COM:

iObject->get_AssociationObject(&AssociationObject);	Получить свойство

Примечание.

1. Установить значение свойства можно только при типе параметров "по координатам" IPoint3D::ParameterType.
2. Для установки объекта необходимо вызвать функцию IPoint3D::SetAssociationObject.

Parameters – Интерфейс параметров точки

Интерфейс...

Тип данных: указатель на интерфейс объекта IKompasAPIObject

Синтаксис Automation:

Parameters = iObject.Parameters;	Получить свойство (*)
Parameters = iObject.GetParameters();	Получить свойство (**)

Синтаксис COM:

iObject->get_Parameters(&Parameters);	Получить свойство

Примечание.

В зависимости от типа IPoint3D::ParameterType, интерфейс параметров должен приводиться к одному из следующих вариантов:

- ksPDisplace - IPoint3DParamDisplace Интерфейс параметров пространственной точки, заданной по смещению от опорного объекта,
- ksPIntersect - IPoint3DParamIntersect Интерфейс параметров пространственной точки, заданной на пересечении опорных объектов,
- ksPCenter - IPoint3DParamCenter Интерфейс параметров пространственной точки, заданной в центре опорного объекта,
- ksPCurve - IPoint3DParamCurve Интерфейс параметров пространственной точки, заданной на кривой со смещением,
- ksPSurface - IPoint3DparamSurface Интерфейс параметров пространственной точки, заданной на поверхности,

▼ ksPProjection - IPoint3DParamProjection Интерфейс параметров пространственной точки, заданной проецированием.

ParameterType – Тип параметров построения точки

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

ParameterType = iObject.ParameterType;	Получить свойство (*)
iObject.ParameterType = ParameterType;	Установить свойство (*)
ParameterType = iObject.GetParameterType( );	Получить свойство (**)
iObject.SetParameterType(ParameterType);	Установить свойство (**)

Синтаксис COM:

iObject->get_ParameterType(&ParameterType);	Получить свойство
iObject->put_ParameterType(ParameterType);	Установить свойство

Symbol – Стиль отображения

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

Symbol = iObject.Symbol;	Получить свойство (*)
iObject.Symbol = Symbol;	Установить свойство (*)
Symbol = iObject.GetSymbol();	Получить свойство (**)
iObject.SetSymbol(Symbol);	Установить свойство (**)

Синтаксис COM:

iObject->get_Symbol(&Symbol);	Получить свойство

```
iObject->put_Symbol(        Установить свойство
Symbol );
```

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
X = iObject.X;              Получить свойство (* )
iObject.X = X;              Установить свойство (* )
X = iObject.GetX();         Получить свойство (**)
iObject.SetX( X );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_X( &X );       Получить свойство
iObject->put_X( X );        Установить свойство
```

Примечание.

Установить значение свойства можно только при типе параметров "по координатам" IPoint3D::ParameterType.

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Y = iObject.Y;              Получить свойство (* )
iObject.Y = Y;              Установить свойство (* )
Y = iObject.GetY();         Получить свойство (**)
iObject.SetY( Y );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Y( &Y );       Получить свойство
iObject->put_Y( Y );        Установить свойство
```

Примечание.

Установить значение свойства можно только при типе параметров "по координатам" IPoint3D::ParameterType.

Z – Координата Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Z = iObject.Z;              Получить свойство (* )
iObject.Z = Z;              Установить свойство (* )
Z = iObject.GetZ();         Получить свойство (**)
iObject.SetZ( Z );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Z( &Z );       Получить свойство
iObject->put_Z( Z );        Установить свойство
```

Примечание.

Установить значение свойства можно только при типе параметров "по координатам" IPoint3D::ParameterType.

IPoint3D – методы

SetAssociationObject – Установить опорный объект

Интерфейс...

Синтаксис Automation:

BOOL SetAssociationObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetAssociationObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

```
obj             - указатель на опорный объект.
```

Возвращаемое значение:

```
TRUE            - в случае успешного
                  завершения,
FALSE           - в случае неудачи.
```

Примечание:

Метод позволяет установить опорный объект только для типа точки по координатам (IPoint3D::ParameterType).

Наследники – вспомогательные объекты

Интерфейс IConjunctivePoint

Справка системы КОМПАС: Команда Присоединительная точка

kompas.chm::/Postroenye_kontr_tochek.htm#connecting_point

Интерфейс присоединительной точки.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPoint3D

IConjunctivePoint

ILocalCSObject

Примечание:

1. Интерфейс можно получить, используя свойство IConjunctivePoints::ConjunctivePoint или IModelObjects::Item или метод IConjunctivePoints::Add.
2. Интерфейс наследует у интерфейса IPoint3D пространственной точки все методы и свойства для позиционирования, за исключением следующих:

   Symbol() - невозможно установить значения свойства, поскольку стиль отображения точки постоянный; получаемое значение свойства во всех случаях будет одинаковым, равным ksCirclePoint из перечисления ksAnnotationSymbolEnum.
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IConjunctivePoint – свойства

Axis – Получить ось присоединительной точки

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Axis = Object.Axis(type)	Получить свойство (*)
Axis = Object.GetAxis(type)	Получить свойство (**)

Синтаксис COM:

Object.get_ObjectX(type, &ObjectX)	Получить свойство

Входные параметры:

type	- тип запрашиваемой оси из перечисления ksObj3dTypeEnum, допустимыми значениями являются следующие: - o3d_axisOX - ось со стрелкой, - o3d_axisOY - ось без стрелки.

Возвращаемое значение:

указатель на ось	- в случае успеха,
NULL	- в случае неудачи.

Примечание.

1. Свойство позволяет получить указатель на интерфейс оси присоединительной точки в соответствии с переданным типом.
2. Свойство доступно только для чтения.

BuildingType – Способ построения присоединительной точки

Интерфейс...

Тип данных: из перечисления ksConjunctivePointTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

DirectionObject1 – Получить направляющий объект 1

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DirectionObject1 = Object.DirectionObject1	Получить свойство (*)

DirectionObject1 = Object.GetDirectionObject1 ()	Получить свойство (**)

Синтаксис COM:

Object.get_DirectionObject 1(&DirectionObject1)	Получить свойство

Примечание.

1. Свойство позволяет получить указатель на интерфейс направляющего объекта 1.
2. Свойство доступно только для чтения.

DirectionObject2 – Получить направляющий объект 2

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DirectionObject2 = Object.DirectionObject2	Получить свойство (*)
DirectionObject2 = Object.GetDirectionObject2()	Получить свойство (**)

Синтаксис COM:

Object.get_DirectionObject 2(&DirectionObject2)	Получить свойство

Примечание.

1. Свойство позволяет получить указатель на интерфейс направляющего объекта 2.
2. Свойство доступно только для чтения.

Direction1 – Направление объекта 1

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- прямое,
FALSE	- обратное.

Синтаксис Automation:

Direction1 = Object.Direction1	Получить свойство (*)
Object.Direction1 = Direction1	Установить свойство (*)

Direction1 = Object.GetDirection1( )	Получить свойство (**)
Object.SetDirection1(Direction1)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction1(&Direction1)	Получить свойство
Object.put_Direction1(Direction1)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать направление объекта 1.

Direction2 – Направление объекта 2

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- прямое,
FALSE	- обратное.

Синтаксис Automation:

Direction2 = Object.Direction2	Получить свойство (*)
Object.Direction2 = Direction2	Установить свойство (*)
Direction2 = Object.GetDirection2( )	Получить свойство (**)
Object.SetDirection2(Direction2)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction2(&Direction2)	Получить свойство
Object.put_Direction2(Direction2)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать направление объекта 2.

UseDirection2 – Использовать второе направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseDirection2 = Object.UseDirection2	Получить свойство (*)
Object.UseDirection2 = UseDirection2	Установить свойство (*)
UseDirection2 = Object.GetUseDirection2( )	Получить свойство (**)
Object.SetUseDirection2(UseDirection2)	Установить свойство (**)

Синтаксис COM:

Object.get_UseDirection2(&UseDirection2)	Получить свойство
Object.put_UseDirection2(UseDirection2)	Установить свойство

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D(First)	Получить свойство (*)
Vector3D = Object.GetVector3D(First)	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(First, &Vector3D)	Получить свойство

Входные параметры:

First	TRUE - вектор, задающий направление оси X, FALSE - вектор, задающий направление оси Y.

Примечание.

1. Свойство позволяет получить интерфейс вектора, задающего направление присоединительной точки.
2. Свойство доступно только для чтения.

IConjunctivePoint – методы

DirectionObject2 – Установить направляющий объект 2

Интерфейс...

Синтаксис Automation:

BOOL SetDirectionObject2(LPDISPATCH NewVal);

Синтаксис COM:

HRESULT SetDirectionObject2(IModelObject * NewVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetVector – Получить вектора направлений главных осей

Интерфейс...

Синтаксис Automation:

BOOL GetVector(ksObj3dTypeEnum Axis, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetVector(ksObj3dTypeEnum Axis, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Axis	- ось.

Выходные параметры:

X	- координата по X,
Y	- координата по Y,
Z	- координата по Z.

InitByMatrix3D – Установить систему координат по матрице SAFEARRAY double (VT_ARRAY | VT_R8)

Интерфейс...

Синтаксис Automation:

BOOL InitByMatrix3D(VARIANT mtr);

Синтаксис COM:

HRESULT InitByMatrix3D(VARIANT mtr, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

mtr	- массив координат SAFEARRAY double (VT_ARRAY | VT_R8).

SetDirectionObject1 – Установить направляющий объект 1

Интерфейс...

Синтаксис Automation:

BOOL SetDirectionObject1(LPDISPATCH NewVal);

Синтаксис COM:

HRESULT SetDirectionObject1(IModelObject * NewVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс IControlPoint

Справка системы КОМПАС: Команда Контрольная точка

`kompas.chm::/Postroenye_kontr_tochek.htm#kontrol_point`

Интерфейс контрольной точки.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPoint3D

IControlPoint

ILocalCSObject

Примечание:

1. Интерфейс можно получить, используя свойство IControlPoints::ControlPoint или IModelObjects::Item или метод IControlPoints::Add.
2. Интерфейс наследует у интерфейса IPoint3D пространственной точки все методы и свойства для позиционирования, за исключением следующих:

   Symbol() - невозможно установить значения свойства, поскольку стиль отображения точки постоянный; получаемое значение свойства во всех случаях будет одинаковым, равным ksCirclePoint из перечисления ksAnnotationSymbolEnum.
3. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

Интерфейс ILocalCoordinateSystem

Интерфейс локальной системы координат.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPoint3D

ILocalCoordinateSystem

ILocalCSObject

Примечание:

1. Интерфейс можно получить, используя свойство интерфейса коллекции ЛСК ILocalCoordinateSystems::LocalCoordinateSystem или метод ILocalCoordinateSystems::Add.
2. Интерфейс наследует от интерфейса 3D точки IPoint3D все методы и свойства для позиционирования ЛСК.
3. Интерфейс позволяет задавать смещение вдоль собственных осей ЛСК с помощью метода ILocalCoordinateSystem::SetDisplacementByAxis.
4. С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

ILocalCoordinateSystem – свойства

AssociationObject – Получить опорный объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject = Получить свойство (*)
Object.AssociationObject
AssociationObject = Получить свойство (**)
Object.GetAssociationObject()

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject) — Получить свойство

Примечание:

Свойство доступно только для чтения.

Current – Состояние ЛСК – текущая/не текущая

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Current = iObject.Current; — Получить свойство (*)
iObject.Current = Current; — Установить свойство (*)
Current = iObject.GetCurrent(); — Получить свойство (**)
iObject.SetCurrent(Current); — Установить свойство (**)

Синтаксис COM:

iObject->get_Current(&Current); — Получить свойство
iObject->put_Current(Current); — Установить свойство

Примечание:

По умолчанию ЛСК создаётся текущей. Чтобы ЛСК создалась не текущей, нужно установить свойство iObject.Current = FALSE.

См. также ILocalCoordinateSystems::SetCurrent.

DefaultObject – Получить примитив ЛСК (ось или плоскость)

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

DefaultObject = iObject.DefaultObject(type); — Получить свойство (*)
DefaultObject = iObject.GetDefaultObject(type); — Получить свойство (**)

Синтаксис COM:

iObject->get_DefaultObject(type, &DefaultObject); Получить свойство

Входные параметры:

type - тип объекта из перечисления ksObj3dTypeEnum;
допустимые типы:
- o3d_axisOX,
- o3d_axisOY
- o3d_axisOZ,
- o3d_planeXOY,
- o3d_planeXOZ,
- o3d_planeYOZ.

Возвращаемое значение:

Указатель на примитив ЛСК (ось или плоскость). - в случае успеха;

Примечание:

1. Свойство доступно только для чтения.
2. Позволяет получить указатель на примитив ЛСК (ось или плоскость) в виде указателя на интерфейс IModelObject.

Hidden – Состояние видимости объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Hidden = Object.Hidden;	Получить свойство (*)
Object.Hidden = Hidden;	Установить свойство (*)
Hidden = Object.GetHidden();	Получить свойство (**)
Object.SetHidden(Hidden);	Установить свойство (**)

Синтаксис COM:

Object.get_Hidden(&Hidden);	Получить свойство
Object.put_Hidden(Hidden);	Установить свойство

LocalCSParameters – Интерфейс параметров ЛСК

Интерфейс...

Тип данных: указатель на интерфейс объекта IKompasAPIObject

Синтаксис Automation:

```
LocalCSParameters = iObject.LocalCSParameters;          Получить свойство (* )
LocalCSParameters = iObject.GetLocalCSParameters();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_LocalCSParameters( &LocalCSParameters );       Получить свойство
```

Примечание:

В зависимости от типа ILocalCoordinateSystem::OrientationType интерфейс параметров должен приводиться к следующим вариантам:

- ksAxisOrientation - ILocalCSAxesDirectionParam – Интерфейс параметров ЛСК для типа ориентации "Направление осей",
- ksEulerCorners - ILocalCSEulerParam – Интерфейс параметров ЛСК для типа ориентации "Система углов Эйлера",
- ksOrientByObject - ILocalCSOrientByObjectParam – Интерфейс параметров ЛСК для типа ориентации "По объекту".

ModelObjectType – Тип 3D объекта из ksObj3dTypeEnum

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

```
ModelObjectType                 =   Получить свойство (* )
Object.ModelObjectType
ModelObjectType                 =   Получить свойство (**)
Object.GetModelObjectType
```

Синтаксис COM:

```
Object.get_ModelObjectType(                 Получить
&ModelObjectType )                          свойство
```

Примечание:

Свойство доступно только для чтения.

Name – Имя элемента трехмерной модели

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = Object.Name;                 Получить свойство (* )
Object.Name = Name;                 Установить свойство (* )
Name = Object.GetName();            Получить свойство (**)
Object.SetName( Name );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Name( &Name );     Получить свойство
Object.put_Name( Name );      Установить свойство
```

OrientationType – Тип ориентирования ЛСК

Интерфейс...

Тип данных: из перечисления ksOrientationTypeEnum

Синтаксис Automation:

```
OrientationType = iObject.OrientationType;          Получить свойство (* )
iObject.OrientationType = OrientationType;          Установить свойство (* )
OrientationType = iObject.GetOrientationType();     Получить свойство (**)
iObject.SetOrientationType( OrientationType );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_OrientationType( &OrientationType );     Получить свойство
iObject->put_OrientationType( OrientationType );      Установить свойство
```

Owner – Объект породивший этот объект

Интерфейс...

Тип данных: Указатель на интерфейс IFeature7

Синтаксис Automation:

```
Owner = Object.Owner                Получить свойство (* )
Owner = Object.GetOwner             Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Owner( &Owner )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Parameters – Получить интерфейс параметров точки

Интерфейс...

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
Parameters = Object.Parameters          Получить свойство (* )
Parameters = Object.GetParameters       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Parameters( &Parameters )    Поолучить свойство
```

Примечание:

Свойство доступно только для чтения.

ParameterType – Тип параметров построения точки

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

```
ParameterType = Object.ParameterType;          Получить свойство (* )
Object.ParameterType = ParameterType;          Установить свойство (* )
ParameterType = Object.GetParameterType();     Получить свойство (**)
Object.SetParameterType( ParameterType );      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ParameterType( &ParameterType );    Получить свойство
Object.put_ParameterType( ParameterType );     Установить свойство
```

Part – Компонент, владеющий элементом

Интерфейс...

Тип данных: Указатель на интерфейс IPart7

Синтаксис Automation:

```
Part = Object.Part                      Получить свойство (* )
Part = Object.GetPart                   Получить свойство (**)
```

Синтаксис COM:

Object.get_Part(&Part) Получить свойство

Примечание:

Свойство доступно только для чтения.

Symbol – Стиль отображения

Интерфейс...

Тип данных: из перечисления ksAnnotationSymbolEnum

Синтаксис Automation:

```
Symbol = Object.Symbol;                 Получить свойство (* )
Object.Symbol = Symbol;                 Установить свойство (* )
Symbol = Object.GetSymbol();            Получить свойство (**)
Object.SetSymbol( Symbol );             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Symbol( &Symbol );           Получить свойство
Object.put_Symbol( Symbol );            Установить свойство
```

Valid – Возвращает признак невырожденности объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Valid = Object.Valid                    Получить свойство (* )
Valid = Object.GetValid                 Получить свойство (**)
```

Синтаксис COM:

Object.get_Valid(&Valid) Получить свойство

Примечание:

Свойство доступно только для чтения.

Vector3D – Вектор, задающий направление оси

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

```
Vector3D = Object.Vector3D( Axis )      Получить свойство (* )
```

Vector3D = Object.GetVector3D(Axis) Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(Axis, &Vector3D) Получить свойство

Входные параметры:

ksObj3dTypeEnum Axis – ось.

Примечание:

Свойство доступно только для чтения.

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X; Получить свойство (*)
Object.X = X; Установить свойство (*)
X = Object.GetX(); Получить свойство (**)
Object.SetX(X); Установить свойство (**)

Синтаксис COM:

Object.get_X(&X); Получить свойство
Object.put_X(X); Установить свойство

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y; Получить свойство (*)
Object.Y = Y; Установить свойство (*)
Y = Object.GetY(); Получить свойство (**)
Object.SetY(Y); Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y); Получить свойство
Object.put_Y(Y); Установить свойство

Z – Координата Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Z = Object.Z;                    Получить свойство (* )
Object.Z = Z;                    Установить свойство (* )
Z = Object.GetZ();               Получить свойство (**)
Object.SetZ( Z );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Z( &Z );              Получить свойство
Object.put_Z( Z );               Установить свойст-
                                 во
```

ILocalCoordinateSystem – методы

GetVector – Получить вектора направлений главных осей

Интерфейс...

Синтаксис Automation:

```
BOOL GetVector( ksObj3dTypeEnum axis,
double * x,
double * y,
double * z );
```

Синтаксис COM:

```
HRESULT GetVector( [in]  ksObj3dTypeEnum   axis,
[out] double * x,
[out] double * y,
[out] double * z,
[out, retval] * Result );
```

Входные параметры:

```
axis              - тип оси из перечисления
                    ksObj3dTypeEnum;
        допустимые типы:
        - o3d_axisOX,
        - o3d_axisOY
        - o3d_axisOZ.
```

Выходные параметры:

```
x       - координата X вектора направления оси,
y       - координата Y вектора направления оси,
z       - координата Z вектора направления оси.
```

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

InitByMatrix3D – Установить систему координат по матрице

Интерфейс...

Синтаксис Automation:

BOOL InitByMatrix3D(VARIANT mtr);

Синтаксис COM:

HRESULT InitByMatrix3D(VARIANT mtr, BOOL * Result);

Входные параметры:

mtr	- массив SafeAttray типа (VT_ARRAY | VT_R8).

Примечание:

1. Элементы матрицы возвращаются в виде одномерного массива из 16 элементов.
2. Матрица имеет размер 4x4.

ReadFromFile – Прочитать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL ReadFromFile(BSTR fileName);

Синтаксис COM:

HRESULT ReadFromFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL* Result);

Входные данные:

fileName	- полное имя файла.

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

Примечание:

Позволяет установить направление осей ЛСК в соответствии со значениями в текстовом файле. Файл должен быть записан с помощью метода ILocalCoordinateSystem::WriteToFile.

SetAssociationObject – Установить опорный объект

Интерфейс...

Синтаксис Automation:

BOOL SetAssociationObject(IModelObject * NewVal);

Синтаксис COM:

HRESULT SetAssociationObject(IModelObject * NewVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

Входные параметры:

NewVal	- указатель на опорный объект.

SetDisplacementByAxis – Установить смещение по оси

Интерфейс...

Синтаксис Automation:

BOOL SetDisplacementByAxis(ksObj3dTypeEnum axis,
double NewVal);

Синтаксис COM:

HRESULT SetDisplacementByAxis(ksObj3dTypeEnum axis,
double NewVal
BOOL * Result);

Входные параметры:

axis	- тип оси из перечисления ksObj3dTypeEnum допустимые типы: - o3d_axisOX, - o3d_axisOY - o3d_axisOZ.
NewVal	- величина смещения.

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

Примечание:

1. Позволяет установить смещение точки начала координат ЛСК вдоль оси ЛСК.
2. Чтобы изменения вступили в силу, нужно вызвать метод IModelObject::Update.

SetStartingOrientation – Установить исходное положение ЛСК

Интерфейс...

Синтаксис Automation:

BOOL SetStartingOrientation();

Синтаксис COM:

HRESULT SetStartingOrientation([out, retval] VARIANT_BOOL* Result);

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

Примечание:

Позволяет установить ЛСК в положение, соответствующее матрице векторов:

Ось	X	Y	Z
Ось X	1	0	0
Ось Y	0	1	0
Ось Z	0	0	1

Update – Изменить свойства объекта

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

WriteToFile – Записать файл с данными

Интерфейс...

Синтаксис Automation:

BOOL WriteToFile(BSTR fileName);

Синтаксис COM:

HRESULT WriteToFile([in] BSTR fileName,
[out, retval] VARIANT_BOOL* Result);

Входные параметры:

fileName	- полное имя файла.

Возвращаемое значение:

TRUE	- в случае успеха;
FALSE	- в случае неудачи.

Примечание:

Позволяет записать значения векторов направлений осей ЛСК в текстовый файл.

Интерфейс IPoint3DParamCenter

Справка системы КОМПАС...

kompas.chm::/CM_CPOINT3D_CENTER.htm

Интерфейс параметров пространственной точки заданной в центре опорного объекта.

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamCenter

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamCenter – свойства

Object – Получить объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

Object = iObject.Object;	Получить свойство (*)
Object = iObject.GetObject();	Получить свойство (**)

Синтаксис COM:

iObject->get_Object(&Object);	Получить свойство

IPoint3DParamCenter – методы

SetObject – Установить объект

Интерфейс...

Синтаксис Automation:

LPDISPATCH obj);

Синтаксис COM:

HRESULT SetObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj	- объект, имеющий центр.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объект должен иметь центр.

Интерфейс IPoint3DParamCurve

Справка системы КОМПАС...

`kompas.chm::/CM_CPOINT3D_ONCURVE.htm`

Интерфейс параметров пространственной точки, заданной на кривой со смещением.

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamCurve

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamCurve – свойства

CurveObject – Получить кривую

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
CurveObject = iObject.CurveObject;      Получить свойство (* )
CurveObject = iObject.GetCurveObject(); Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_CurveObject( &CurveObject ); Получить свойство
```

Direction – Направление смещения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = iObject.Direction;          Получить свойство (* )
iObject.Direction = Direction;          Установить свойство (* )
Direction = iObject.GetDirection();     Получить свойство (**)
iObject.SetDirection( Direction );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Direction( &Direction );   Получить свойство
iObject->put_Direction( Direction );    Установить свойство
```

Значения свойства:

```
TRUE    - прямое направление,
FALSE   - обратное направление.
```

Offset – Величина смещения

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Offset = iObject.Offset;                Получить свойство (* )
iObject.Offset = Offset;                Установить свойство (* )
Offset = iObject.GetOffset();           Получить свойство (**)
iObject.SetOffset( Offset );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Offset( &Offset );          Получить свойство
iObject->put_Offset( Offset );           Установить свойство
```

Примечание:

1. Для типа смещения по параметру U (ksOffsetByU) - свойство 0..100 (величина U в %).
2. Для типа смещения по длине дуги (ksOffsetByLen) - свойство определяет длину дуги.
3. Для типа смещения по центральному углу дуги (ksOffsetByAngle) - свойство определяет угол.

OffsetType – Тип задания смещения

Интерфейс...

Тип данных: из перечисления ksPoint3DCurveParamTypeEnum

Синтаксис Automation:

```
OffsetType = iObject.OffsetType;     Получить свойство (* )
iObject.OffsetType = OffsetType;     Установить свойство (* )
OffsetType                       =   Получить свойство (**)
iObject.GetOffsetType();
iObject.SetOffsetType(               Установить свойство (**)
OffsetType );
```

Синтаксис COM:

```
iObject->get_OffsetType(             Получить свойство
&OffsetType );
iObject->put_OffsetType(             Установить свойство
OffsetType );
```

IPoint3DParamCurve – методы

SetCurveObject – Установить объект

Интерфейс...

Синтаксис Automation:

BOOL SetCurveObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetCurveObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

```
obj                  - кривая.
```

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IPoint3DParamDisplace

Справка системы КОМПАС...

kompas.chm::/CM_CPOINT3D_DISPLACE.htm

Интерфейс параметров пространственной точки, заданной по смещению от опорного объекта.

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamDisplace

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamDisplace – свойства

AssociationVertex – Интерфейс опорной точки

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

AssociationVertex = iObject.AssociationVertex;	=	Получить свойство (*)
AssociationVertex = iObject.GetAssociationVertex();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AssociationVertex(&AssociationVertex);	Получить свойство

Distance – Расстояние от опорного объекта

Интерфейс...

Тип данных: double

Distance = iObject.Distance;	Получить свойство (*)
iObject.Distance = Distance;	Установить свойство (*)

```
Distance = iObject.GetDistance();    Получить свойство (**)
iObject.SetDistance( Distance );     Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Distance( &Distance );    Получить свойство
iObject->put_Distance( Distance );     Установить свойство
```

DX – Смещение по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DX = iObject.DX;                 Получить свойство (* )
iObject.DX = DX;                 Установить свойство (* )
DX = iObject.GetDX();            Получить свойство (**)
iObject.SetDX( DX );             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DX( &DX );          Получить свойство
iObject->put_DX( DX );           Установить свойство
```

DY – Смещение по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DY = iObject.DY;                 Получить свойство (* )
iObject.DY = DY;                 Установить свойство (* )
DY = iObject.GetDY();            Получить свойство (**)
iObject.SetDY( DY );             Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DY( &DY );          Получить свойство
iObject->put_DY( DY );           Установить свойство
```

DZ – Смещение по Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
DZ = iObject.DZ;                Получить свойство (* )
iObject.DZ = DZ;                Установить свойство (* )
DZ = iObject.GetDZ();           Получить свойство (**)
iObject.SetDZ( DZ );            Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_DZ( &DZ );         Получить свойство
iObject->put_DZ( DZ );          Установить свойство
```

GuidingObject – Направляющий объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
GuidingObject = iObject.GuidingObject;      = Получить свойство (* )
GuidingObject = iObject.GetGuidingObject(); = Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_GuidingObject( &GuidingObject );   Получить свойство
```

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

```
Vector3D = Object.Vector3D          Получить свойство (* )
Vector3D = Object.GetVector3D()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Vector3D( &Vector3D )    Получить свойство
```

Свойство позволяет получить интерфейс вектора, задающего направление смещения точки.

Примечание:

Свойство доступно только для чтения.

IPoint3DParamDisplace – методы

SetAssociationVertex – Установить опорную точку

Интерфейс...

Синтаксис Automation:

BOOL SetAssociationVertex(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetAssociationVertex([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - опорная точка.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetGuidingObject – Установить направляющий объект

Интерфейс...

Синтаксис Automation:

BOOL SetGuidingObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetGuidingObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - опорная точка.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IPoint3DParamIntersect

Справка системы КОМПАС...

`kompas.chm::/CM_CPOINT3D_INTERSECT.htm`

Интерфейс параметров пространственной точки, заданной на пересечении опорных объектов.

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamIntersect

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamIntersect – свойства

CartPointArray – Массив координат точек пересечения кривой и поверхности в виде SAFEARRAY VT_R8 – VT_ARRAY | VT_R8

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
CartPointArray = Получить свойство (* )
iObject.CartPointArray;
CartPointArray = Получить свойство (**)
iObject.GetCartPointArray();
```

Синтаксис COM:

```
iObject->get_CartPointArray( Получить свойство
&CartPointArray );
```

Примечание.

Массив SafeArray типа VT_ARRAY | VT_R8 координат точек для пересечения.

CurveObject – Получить кривую

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
CurveObject = Получить свойство (* )
iObject.CurveObject;
CurveObject = Получить свойство (**)
iObject.GetCurveObject();
```

Синтаксис COM:

```
iObject->get_CurveObject(                    Получить свойство
&CurveObject );
```

Index – Индекс точки пересечения

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Index = iObject.Index;              Получить свойство (* )
iObject.Index = Index;              Установить свойство (* )
Index = iObject.GetIndex();         Получить свойство (**)
iObject.SetIndex( Index );          Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Index( &Index );       Получить свойство
iObject->put_Index( Index );        Установить свойство
```

Примечание:

Индекс нужной точки можно вычислить из IPoint3DParamIntersect::CartPointArray.

SurfaceObject – Получить поверхность

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
SurfaceObject                =  Получить свойство (* )
iObject.SurfaceObject;
SurfaceObject                =  Получить свойство (**)
iObject.GetSurfaceObject();
```

Синтаксис COM:

```
iObject->get_SurfaceObject(                  Получить свойство
&SurfaceObject );
```

IPoint3DParamIntersect – методы

SetCurveObject – Установить направляющий объект

Интерфейс...

Синтаксис Automation:

BOOL SetCurveObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetCurveObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - кривая.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetSurfaceObject – Установить поверхность

Интерфейс...

Синтаксис Automation:

BOOL SetSurfaceObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetSurfaceObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - поверхность.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IPoint3DParamProjection

Справка системы КОМПАС...

`kompas.chm::/CM_CPOINT3D_PROJECT.htm`

Интерфейс параметров пространственной точки, заданной проецированием.

Иерархия:

IDispatch
    IKompasAPIObject
        IPoint3DParamProjection

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamProjection – свойства

AssociationVertex – Получить опорную точку

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
AssociationVertex = Получить свойство (* )
iObject.AssociationVertex
AssociationVertex = Получить свойство (**)
iObject.GetAssociationVertex()
```

Синтаксис COM:

```
iObject->get_AssociationVertex( Получить свойство
&AssociationVertex )
```

GuidingObject – Получить направляющий объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
GuidingObject = Получить свойство (* )
iObject.GuidingObject;
GuidingObject = Получить свойство (**)
iObject.GetGuidingObject();
```

Синтаксис COM:

```
iObject->get_GuidingObject( Получить свойство
&GuidingObject );
```

SurfaceObject – Получить поверхность

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
SurfaceObject = Получить свойство (* )
iObject.SurfaceObject;
SurfaceObject = Получить свойство (**)
iObject.GetSurfaceObject();
```

Синтаксис COM:

iObject->get_SurfaceObject(&SurfaceObject);	Получить свойство

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс вектора, задающего направление проецирования точки.
2. Свойство доступно только для чтения.

IPoint3DParamProjection – методы

SetAssociationVertex – Установить опорную точку

Интерфейс...

Синтаксис Automation:

BOOL SetAssociationVertex(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetAssociationVertex([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj	- опорная точка.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetGuidingObject – Установить направляющий объект

Интерфейс...

Синтаксис Automation:

BOOL SetGuidingObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetGuidingObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - опорная точка.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetSurfaceObject – Установить объект1

Интерфейс...

Синтаксис Automation:

BOOL SetSurfaceObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetSurfaceObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - поверхность.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Интерфейс IPoint3DParamSurface

Справка системы КОМПАС...

`kompas.chm::/CM_CPOINT3D_ONFACE.htm`

Интерфейс параметров пространственной точки, заданной на поверхности.

Иерархия:

IDispatch
    IKompasAPIObject
        IPoint3DParamSurface

Примечание:

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamSurface – свойства

AllowBoundaries – TRUE – Учитывать границы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AllowBoundaries = Object.AllowBoundaries	=	Получить свойство (*)
Object.AllowBoundaries = AllowBoundaries	=	Установить свойство (*)
AllowBoundaries = Object.GetAllowBoundaries()	=	Получить свойство (**)
Object.SetAllowBoundaries(AllowBoundaries)		Установить свойство (**)

Синтаксис COM:

Object.get_AllowBoundaries(&AllowBoundaries)	Получить свойство
Object.put_AllowBoundaries(AllowBoundaries)	Установить свойство

Object1– Получить объект1

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

Object1 = iObject.Object1;	Получить свойство (*)
Object1 = iObject.GetObject1();	Получить свойство (**)

Синтаксис COM:

iObject->get_Object1(&Object1);	Получить свойство

Object2 – Получить объект2

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
Object2 = iObject.Object2;          Получить свойство (* )
Object2 = iObject.GetObject2();     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Object2(  &Object2     Получить свойство
);
```

OffsetType – Тип задания смещения

Интерфейс...

Тип данных: из перечисления ksPoint3DCurveParamTypeEnum

Синтаксис Automation:

```
OffsetType = iObject.OffsetType;    Получить свойство (* )
iObject.OffsetType = OffsetType;    Установить свойство (* )
OffsetType                    =     Получить свойство (**)
iObject.GetOffsetType();
iObject.SetOffsetType(              Установить свойство (**)
OffsetType );
```

Синтаксис COM:

```
iObject->get_OffsetType(            Получить свойство
&OffsetType );
iObject->put_OffsetType(            Установить свойство
OffsetType );
```

Offset1 – Величина смещения1

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Offset1 = iObject.Offset1;          Получить свойство (* )
iObject.Offset1 = Offset1;          Установить свойство (* )
Offset1 = iObject.GetOffset1();     Получить свойство (**)
iObject.SetOffset1( Offset1 );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Offset1( &Offset1 );        Получить свойство
iObject->put_Offset1( Offset1 );         Установить свойство
```

Примечание:

1. Для типа смещения по параметру U и V (ksOffsetByUV) - свойство 0..100 (величина U в %).
2. Для типа смещения по расстояниям от плоских объектов (ksOffsetByLenFromObj) - свойство определяет расстояние от плоского объекта 1.

Offset2 – Величина смещения2

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Offset2 = iObject.Offset2;          Получить свойство (* )
iObject.Offset2 = Offset2;          Установить свойство (* )
Offset2 = iObject.GetOffset2();     Получить свойство (**)
iObject.SetOffset2( Offset2 );      Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Offset2( &Offset2 );        Получить свойство
iObject->put_Offset2( Offset2 );         Установить свойство
```

Примечание:

1. Для типа смещения по параметру U и V (ksOffsetByUV) - свойство 0..100 (величина V в %).
2. Для типа смещения по расстояниям от плоских объектов (ksOffsetByLenFromObj) - свойство определяет расстояние от плоского объекта 2.

SurfaceObject – Получить поверхность

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

```
SurfaceObject = iObject.SurfaceObject;          Получить свойство (* )
SurfaceObject = iObject.GetSurfaceObject();     Получить свойство (**)
```

Синтаксис COM:

iObject->get_SurfaceObject(&SurfaceObject); — Получить свойство

IPoint3DParamSurface – методы

SetObject1 – Установить объект1

Интерфейс...

Синтаксис Automation:

BOOL SetObject1(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetObject1([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - плоский объект.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetObject2 – Установить объект2

Интерфейс...

Синтаксис Automation:

BOOL SetObject2(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetObject2([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj - плоский объект.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetSurfaceObject – Установить поверхность

Интерфейс...

Синтаксис Automation:

BOOL SetSurfaceObject(LPDISPATCH obj);

Синтаксис COM:

HRESULT SetSurfaceObject([in] IModelObject* obj, [out, retval] VARIANT_BOOL * Result);

Входные параметры:

obj	- поверхность.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IPoint3DParamByCylinder

Справка системы КОМПАС...

kompas.chm::/CM_CPOINT3D_ONFACE.htm

Интерфейс параметров пространственной точки, заданной по цилиндрическим координатам

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamByCylinder

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamByCylinder – свойства

A – Координата A (угол поворота точки)

Интерфейс...

Тип данных: double

Синтаксис Automation:

A = Object.A	Получить свойство (*)
Object.A = A	Установить свойство (*)
A = Object.GetA()	Получить свойство (**)
Object.SetA(A)	Установить свойство (**)

Синтаксис COM:

Object.get_A(&A)	Получить свойство
Object.put_A(A)	Установить свойство

Свойство позволяет устанавливать и получать угол поворота точки.

AssociationObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject	Получить свойство (*)
Object.AssociationObject = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject()	Получить свойство (**)
Object.SetAssociationObject(AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject)	Получить свойство
Object.put_AssociationObject(AssociationObject)	Установить свойство

Свойство позволяет связать точку с точечным объектом.

Примечание:

Если точка связана с точечным объектом, то координаты недоступны для изменения.

R – Координата R (радиус цилиндра)

Интерфейс...

Тип данных: double

Синтаксис Automation:

R = Object.R	Получить свойство (*)
Object.R = R	Установить свойство (*)
R = Object.GetR()	Получить свойство (**)
Object.SetR(R)	Установить свойство (**)

Синтаксис COM:

Object.get_R(&R)	Получить свойство
Object.get_R(&R)	Установить свойство

Свойство позволяет устанавливать и получать радиус цилиндра.

RadiusObject – Объект радиуса

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

RadiusObject = Object.RadiusObject	Получить свойство (*)
Object.RadiusObject = RadiusObject	Установить свойство (*)
RadiusObject = Object.GetRadiusObject()	Получить свойство (**)
Object.SetRadiusObject(RadiusObject)	Установить свойство (**)

Синтаксис COM:

Object.get_RadiusObject(&RadiusObject)	Получить свойство
Object.put_RadiusObject(RadiusObject)	Установить свойство

Свойство позволяет устанавливать и получать объект, задающий радиус. Таким объектом может являться ребро или контур в эскизе, имеющий форму окружности или дуги окружности; кривая - окружность или дуга; цилиндрическая или сферическая грань.

Примечание:

Если задан объект, определяющий радиус, координата R недоступна для изменения.

Z – Координата по оси Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

Z = Object.Z	Получить свойство (*)
Object.Z = Z	Установить свойство (*)
Object.Z = Z	Получить свойство (**)
Object.SetZ(Z)	Установить свойство (**)

Синтаксис COM:

Object.get_Z(&Z)	Получить свойство
Object.put_Z(Z)	Установить свойство

Свойство позволяет устанавливать и получать координату по оси Z.

Интерфейс IPoint3DParamBySphere

Справка системы КОМПАС...

`kompas.chm::/CM_CPOINT3D_ONFACE.htm`

Интерфейс параметров пространственной точки, заданной по сферическим координатам.

Иерархия:

IDispatch

IKompasAPIObject

IPoint3DParamBySphere

Интерфейс можно получить с помощью свойства пространственной точки IPoint3D::Parameters.

IPoint3DParamBySphere – свойства

A – Координата A (азимутальный угол)

Интерфейс...

Тип данных: double

Синтаксис Automation:

A = Object.A	Получить свойство (*)
Object.A = A	Установить свойство (*)
A = Object.GetA()	Получить свойство (**)
Object.SetA(A)	Установить свойство (**)

Синтаксис COM:

Object.get_A(&A)	Получить свойство
Object.put_A(A)	Установить свойство

Свойство позволяет устанавливать и получать азимутальный угол.

AssociationObject – Опорный объект

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject	Получить свойство (*)
Object.AssociationObject = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject()	Получить свойство (**)
Object.SetAssociationObject(AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject)	Получить свойство
Object.put_AssociationObject(AssociationObject)	Установить свойство

Свойство позволяет связать точку с точечным объектом.

Примечание:

Если точка связана с точечным объектом, то координаты недоступны для изменения.

B – Координата B (зенитный угол)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
B = Object.B                Получить свойство (* )
Object.B = B                Установить свойство (* )
B           =               Получить свойство (**)
Object.GetB()
Object.SetB(                Установить свойство (**)
B )
```

Синтаксис COM:

```
Object.get_B( &B )          Получить свойство
Object.put_B(  B )          Установить свойство
```

Свойство позволяет устанавливать и получать зенитный угол.

R – Координата R (радиус сферы)

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
R = Object.R                Получить свойство (* )
Object.R = R                Установить свойство (* )
R           =               Получить свойство (**)
Object.GetR()
Object.SetR(  R             Установить свойство (**)
)
```

Синтаксис COM:

```
Object.get_R(               Получить свойство
&R )
Object.get_R(               Установить свойство
&R )
```

Свойство позволяет устанавливать и получать радиус сферы.

RadiusObject – Объект радиуса

Интерфейс...

Тип данных: указатель на интерфейс объекта IModelObject

Синтаксис Automation:

RadiusObject = Object.RadiusObject	Получить свойство (*)
Object.RadiusObject = RadiusObject	Установить свойство (*)
RadiusObject = Object.GetRadiusObject()	Получить свойство (**)
Object.SetRadiusObject(RadiusObject)	Установить свойство (**)

Синтаксис COM:

Object.get_RadiusObject(&RadiusObject)	Получить свойство
Object.put_RadiusObject(RadiusObject)	Установить свойство

Свойство позволяет устанавливать и получать объект, задающий радиус. Таким объектом может являться ребро или контур в эскизе, имеющий форму окружности или дуги окружности; кривая - окружность или дуга; цилиндрическая или сферическая грань.

Примечание:

Если задан объект, определяющий радиус, координата R недоступна для изменения.

Эскиз

Интерфейс ISketchs

Справка системы КОМПАС...

`kompas.chm::/CM_MAKE_SKETCH.htm`

Интерфейс коллекции элементов "Эскиз".

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ISketchs

Примечание.

Получить интерфейс можно, используя свойство контейнера объектов 3D IModelContainer::Sketchs.

ISketchs- свойства

Sketch – Элемент "Эскиз", заданный по индексу или ссылке

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/CM_MAKE_SKETCH.htm

Тип данных: указатель на интерфейс ISketch.

Синтаксис Automation:

ISketch = iObject.Isketch(index)	Получить свойство (*)
ISketch = iObject.Isketch(index)	Получить свойство (**)

Синтаксис COM:

iObject->get_ISketch(&ISketch)	index,	Получить свойство

Входные параметры:

VARIANT index	- индекс операции.

Примечание:

В качестве индекса может использоваться:

- индекс объекта в коллекции,
- ссылка на объект (reference).

ISketchs – методы

Add – Создать объект "Эскиз"

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/CM_MAKE_SKETCH.htm

Синтаксис Automation:

LPDISPATH Add();

Синтаксис COM:

HRESULT Add(ISketch** Result);

Возвращаемое значение:

Указатель на интерфейс эскиза ISketch	- в случае удачного завершения.

Примечания:

1. Метод позволяет создать новый объект "Эскиз".
2. После получения нового интерфейса нужно задать параметры листового тела и вызвать метод IModelObject::Update.

Интерфейс ISketch

Справка системы КОМПАС...

KOMPAS.chm::/CM_MAKE_SKETCH.htm

Интерфейс элемента Эскиз.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ISketch

ILocalCSObject

Примечание:

Интерфейс можно получить с помощью метода коллекции эскизов ISketchs::Add или свойства ISketchs::Sketch.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

ISketch – свойства

AssociationObject – Точка привязки

Интерфейс

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
AssociationObject = Получить свойство (* )
Object.AssociationObject;
Object.AssociationObject = Установить свойство (* )
AssociationObject;
AssociationObject = Получить свойство (**)
Object.GetAssociationObject()
);
```

Object.SetAssociationObject(AssociationObject);	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject);	Получить свойство
Object.put_AssociationObject(AssociationObject);	Установить свойство

Angle – Угол поворота эскиза относительно проекции системы координат модели на плоскость эскиза (в градусах)

Интерфейс

Тип данных: double

Синтаксис Automation:

Angle = iObject.Angle;	Получить свойство (*)
iObject.Angle = Angle;	Установить свойство (*)
Angle = iObject.GetAngle();	Получить свойство (**)
iObject.SetAngle(Angle);	Установить свойство (**)

Синтаксис COM:

iObject->get_Angle(&Angle);	Получить свойство
iObject->put_Angle(Angle);	Установить свойство

CoordinateSystem – система координат эскиза

Интерфейс

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

CoordinateSystem = Object.CoordinateSystem;	Получить свойство (*)
Object.CoordinateSystem = CoordinateSystem;	Установить свойство (*)
CoordinateSystem = Object.GetCoordinateSystem();	Получить свойство (**)
Object.SetCoordinateSystem(CoordinateSystem);	Установить свойство (**)

Синтаксис COM:

Object.get_CoordinateSystem(&CoordinateSystem);	Получить свойство
Object.put_CoordinateSystem(CoordinateSystem);	Установить свойство

DirectingObject – Направляющий объект для оси

Интерфейс

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

DirectingObject = Object.DirectingObject(Axis);	Получить свойство (*)
Object.DirectingObject(Axis) = DirectingObject;	Установить свойство (*)
DirectingObject = Object.GetDirectingObject(Axis);	Получить свойство (**)
Object.SetDirectingObject(Axis, DirectingObject);	Установить свойство (**)

Синтаксис COM:

Object.get_DirectingObject(Axis, &DirectingObject);	Получить свойство
Object.put_DirectingObject(Axis, DirectingObject);	Установить свойство

Входные параметры:

Axis	- тип оси: o3d_axisOX - ось X, o3d_axisOY - ось Y

Примечание:

1. При указании базовой плоскости можно задать направляющий объект для оси X или оси Y.
2. Направление второй оси будет определяться как векторное произведение оси Z, направленной по нормали к плоскости, и заданным направлением.
3. Направление осей можно задать также параметрами вектора.

Fixed – Фиксация

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

```
Fixed = Object.Fixed;                    Получить свойство (* )
Object.Fixed = Fixed;                    Установить свойство (* )
Fixed = Object.GetFixed();               Получить свойство (**)
Object.SetFixed( Fixed );                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Fixed( &Fixed );      Получить свойство
Object.put_Fixed(  Fixed );      Установить свойство
```

LeftHandedCS – Признак левосторонней системы координат

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

```
LeftHandedCS = Object.LeftHandedCS;      Получить свойство (* )
Object.LeftHandedCS = LeftHandedCS;      Установить свойство (* )
LeftHandedCS =                           Получить свойство (**)
Object.GetLeftHandedCS();
Object.SetLeftHandedCS( LeftHandedCS     Установить свойство (**)
);
```

Синтаксис COM:

```
Object.get_LeftHanded          Получить
CS( &LeftHandedCS );           свойство
Object.put_LeftHanded          Установить
CS(  LeftHandedCS );           свойство
```

Plane – Опорная плоскость

Интерфейс

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

```
Plane = iObject.Plane;       Получить свойство (* )
iObject.Plane = Plane;       Установить свойство (* )
Plane = iObject.GetPlane();  Получить свойство (**)
iObject.SetPlane( Plane );   Установить свойство (**)
```

Синтаксис COM:

iObject->get_Plane(&Plane);	Получить свойство
iObject->put_Plane(Plane);	Установить свойство

Vector3D – Вектор, задающий направление оси

Интерфейс

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D(Axis)	Получить свойство(*)
Vector3D = Object.GetVector3D(Axis)	Получить свойство(**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Axis,	Получить свойство.

Входные параметры:

Axis	- тип оси: o3d_axisOX - ось X, o3d_axisOY - ось Y

Примечание:

1. При указании базовой плоскости можно задать направляющий объект для оси X или оси Y.
2. Направление второй оси будет определяться как векторное произведение оси Z, направленной по нормали к плоскости, и заданным направлением.
3. Свойство доступно только для чтения.

ISketch – методы

AddSketch – Создать эскиз из уже имеющегося в заданной плоскости

Интерфейс

Синтаксис Automation:

BOOL AddSketch(IModelObject * Sketch, IModelObject * Plane);

Синтаксис COM:

HRESULT AddSketch(IModelObject * Sketch, IModelObject * Plane, BOOL * Result);

Входные параметры:

Sketch	- указатель на эскиз,
Plane	- указатель на плоскость.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

BeginEdit – Войти в режим редактирования эскиза

Интерфейс

Синтаксис Automation:

BeginEdit();

Возвращаемое значение:

SketchDoc	- указатель на интерфейс IFragmentDocument.

Синтаксис COM:

HRESULT BeginEdit([out, retval] IFragmentDocument ** SketchDoc);

Возвращаемое значение:

SketchDoc	- указатель на интерфейс IFragmentDocument.

Примечание:

После редактирования необходимо вызвать функцию ISketch::EndEdit.

BeginEditEx – Войти в режим редактирования эскиза

Интерфейс

Синтаксис Automation:

LPDISPATCH BeginEditEx(BOOL ReadOnly);

Синтаксис COM:

HRESULT BeginEditEx(BOOL ReadOnly, IFragmentDocument * * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ReadOnly	- = TRUE - для чтения.

DeleteWrongProjection – Удалить ошибочные проекции

Интерфейс

Синтаксис Automation:

BOOL DeleteWrongProjection();

Синтаксис COM:

HRESULT DeleteWrongProjection(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

EndEdit – Выйти из режима редактирования эскиза

Интерфейс

Синтаксис Automation:

BOOL EndEdit();

Синтаксис COM:

HRESULT EndEdit([out, retval] VARIANT_BOOL * res);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetLocation – Получить смещение системы координат эскиза относительно проекции системы координат модели на плоскость эскиза

Интерфейс

Синтаксис Automation:

BOOL GetLocation(double X, double Y);

Синтаксис COM:

HRESULT GetLocation([out] double * X, [out] double * Y, [out, retval] VARIANT_BOOL * PVal);

Выходные параметры:

X	- смещение системы координат эскиза по оси X,
Y	- смещение системы координат эскиза по оси Y.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetLoftPoint – Получить координаты точки в плоскости эскиза

Интерфейс

Синтаксис Automation:

BOOL GetLoftPoint(double X, double Y);

Синтаксис COM:

HRESULT GetLoftPoint([out] double * X, [out] double * Y, [out, retval] VARIANT_BOOL * PVal);

Выходные параметры:

X	- значение координаты точки по оси X,
Y	- значение координаты точки по оси Y.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

RotateAxis – Сменить направление оси на противоположное

Интерфейс

Синтаксис Automation:

BOOL RotateAxis(ksObj3dTypeEnum axis);

Синтаксис COM:

HRESULT RotateAxis(ksObj3dTypeEnum axis, BOOL * Result);

Входной параметр:

axis	- тип оси из перечисления ksObj3dTypeEnum (должны передаваться типы : o3d_axisOX o3d_axisOY).

Возвращаемое значение :

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет менять направления осей на противоположные.
2. Для осей можно выбирать направляющие объекты (см. метод ISketch::DirectingObject ,ISketch::Vector3D).
3. Можно выбрать направление для оси X или оси Y, третья ось ориентируется, исходя из положения оси Z и выбранной.
4. Менять направление на противоположное можно только у выбранных осей.

SetLocation – Установить смещение системы координат эскиза относительно проекции системы координат модели на плоскость эскиза

Интерфейс

Синтаксис Automation:

BOOL SetLocation(double X, double Y);

Синтаксис COM:

HRESULT SetLocation([in] double X, [in] double Y, [out, retval] VARIANT_BOOL * PVal);

Входные параметры:

X	- смещение системы координат эскиза по оси X,
Y	- смещение системы координат эскиза по оси Y.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetLoftPoint – Установить координаты точки в плоскости эскиза

Интерфейс

Синтаксис Automation:

BOOL SetLoftPoint(double X, double Y);

Синтаксис COM:

HRESULT SetLoftPoint([in] double X, [in] double Y, [out, retval] VARIANT_BOOL * PVal);

Входные параметры:

X	- значение координаты точки по оси X,
Y	- значение координаты точки по оси Y.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

WriteToFragment – Сохранить эскиз во фрагмент

Интерфейс

Синтаксис Automation:

BOOL AddSketch(IModelObject * Sketch, IModelObject * Plane);

Синтаксис COM:

HRESULT AddSketch(IModelObject * Sketch, IModelObject * Plane, BOOL * Result);

Входные параметры:

FileName	- имя файла фрагмента.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Поверхности

Интерфейс IMathSurface3D

Интерфейс математической поверхности в трехмерном пространстве.

Иерархия:

IDispatch

IKompasAPIObject

IMathSurface3D

IMathSurface3D – свойства

BoundaryCount – Получить количество границ

Интерфейс...

Тип данных: long

Синтаксис Automation:

BoundaryCount Object.BoundaryCount	=	Получить свойство (*)
BoundaryCount Object.GetBoundaryCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_BoundaryCount(&BoundaryCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ClosedU – Получить замкнутость кривой по U

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedU = Object.ClosedU		Получить свойство (*)
ClosedU	=	Получить свойство (**)
Object.GetClosedU()		

Синтаксис COM:

Object.get_ClosedU(	Получить
&ClosedU)	свойство

Примечание:

Свойство доступно только для чтения.

ClosedV – Получить замкнутость кривой по V

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedV= Object.ClosedV		Получить свойство (*)
ClosedV	=	Получить
Object.GetClosedV()		свойство (**)

Синтаксис COM:

Object.get_ClosedV(	Получить
&ClosedV)	свойство

Примечание:

Свойство доступно только для чтения.

ParamUMax – Получить значение параметра U конечное

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamUMax	=	Получить
Object.ParamUMax		свойство (*)
ParamUMax	=	Получить
Object.GetParamUMax()		свойство (**)

Синтаксис COM:

Object.get_ParamUMax(&ParamUMax)		Получить свойство

Примечание:

Свойство доступно только для чтения.

ParamVMax – Получить значение параметра V конечное

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamVMax Object.ParamVMax	=	Получить свойство (*)
ParamVMax Object.GetParamVMax()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ParamVMax(&ParamVMax)		Получить свойство

Примечание:

Свойство доступно только для чтения.

ParamUMin – Получить значение параметра U начальное

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamUMin Object.ParamUMin	=	Получить свойство (*)
ParamUMin Object.GetParamUMin()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ParamUMin(&ParamUMin)		Получить свойство

Примечание:

Свойство доступно только для чтения.

ParamVMin – Получить значение параметра V начальное

Интерфейс...

Тип данных: double

Синтаксис Automation:

ParamVMin Object.ParamVMin	=	Получить свойство (*)
ParamVMin Object.GetParamVMin()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ParamVMin(&ParamVMin)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Surface3DType – Тип поверхности

Интерфейс...

Тип данных: из перечисления ksMathSurface3DTypeEnum

Синтаксис Automation:

Surface3DType Object.Surface3DType	=	Получить свойство (*)
Surface3DType Object.GetSurface3DType()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Surface3DType(&Surface3DType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IMathSurface3D – методы

GetArea – Получить площадь грани (ST_MIX_MM..ST_MIX_M единицы измерения)

Интерфейс...

Синтаксис Automation:

double GetArea(ksLengthUnitsEnum BitVector);

Синтаксис COM:

HRESULT GetArea(ksLengthUnitsEnum BitVector, double * Result) ;

Возвращаемое значение:

- площадь поверхности.

Входные параметры:

BitVector	- единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

GetBoundaryUVNurbs – Получить параметры границы поверхности в UV NURBS-представлении

Интерфейс...

Синтаксис Automation:

BOOL GetBoundaryUVNurbs(BOOL UV, BOOL Closed, long LoopIndex, long EdgeIndex, long * Degree, VARIANT * Points, VARIANT * Weights, VARIANT * Knots, double * TMin, double * TMax);

Синтаксис COM:

HRESULT GetBoundaryUVNurbs(BOOL UV, BOOL Closed, long LoopIndex, long EdgeIndex, long * Degree, VARIANT * Points, VARIANT * Weights, VARIANT * Knots, double * TMin, double * TMax, BOOL * Result) ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

UV	- TRUE - получить параметры границы поверхности в UV- параметрическом представлении исходной поверхности, - FALSE - получить параметры границы поверхности в 3D координатах,
Closed	- TRUE - сомкнуть, если граница разомкнутая, - FALSE - разомкнуть, если граница замкнутая,
LoopIndex	- индекс цикла,
EdgeIndex	- индекс ребра в цикле (совпадает с индексом ребра в коллекции ориентированных ребер).

Выходные параметры:

Degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
Points	- массив параметров UV или координат вершин - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
Weights	- веса точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
Knots	- узлы точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
TMin, TMax	- минимальный и максимальный параметры.

Примечание:

1. Если признак UV равен TRUE, то в массиве points возвращаются параметры UV для Nurbs-представления границы. Параметры в массиве лежат в следующей последовательности:

   u0, v0, u1, v1, ... ui, vi.

2. Если признак UV равен FALSE, то в массиве points возвращаются координаты вершин Nurbs-представления границы. Координаты точек в полученном массиве лежат в следующей последовательности:

   x0, y0, z0, x1, y1, z1, ...xi, yi, zi.

3. Если индекс ребра равен -1, формируется Nurbs-представление контура границы.
   Если индекс >= 0 то формируется Nurbs-представление ребра, входящего в цикл, параметр unclamped при этом игнорируется.

GetDerivativeU – Получить первую производную по U

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeU(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeU(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result) ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- первая производная по U.

GetDerivativeV – Получить вторую производную по V

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- первая производная по V.

GetDerivativeUU – Получить вторую производную по UU

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeUU(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeUU(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- вторая производная по UU.

GetDerivativeUUU – Получить третью производную по UUU

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeUUU(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeUUU(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- третья производная по UUU.

GetDerivativeUUV – Получить третью производную по UUV

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeUUV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeUUV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- третья производная по UUV.

GetDerivativeUV – Получить вторую производную по UV

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeUV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeUV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- вторая производная по UV.

GetDerivativeUVV – Получить третью производную по UVV

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeUVV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeUVV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- третья производная по UVV.

GetDerivativeVV – Получить вторую производную по VV

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeVV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeVV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- вторая производная по VV.

GetDerivativeVVV – Получить третью производную по VVV

Интерфейс...

Синтаксис Automation:

BOOL GetDerivativeVVV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetDerivativeVVV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y,Z	- третья производная по VVV.

GetEdgesCount – Получить количество ребер в границе

Интерфейс...

Синтаксис Automation:

long GetEdgesCount(long LoopIndex);

Синтаксис COM:

HRESULT GetEdgesCount(long LoopIndex, long * Result);

Возвращаемое значение:

- количество ребер в границе.

Выходные параметры:

LoopIndex	- индекс границы.

GetGabarit – Получить габарит

Интерфейс...

Синтаксис Automation:

BOOL GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2);

Синтаксис COM:

HRESULT GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

X1, Y1, Z1, X2, Y2, Z2	- координаты вершин габаритного параллелепипеда.

Примечание:

Координаты вершин параллелепипеда возвращаются в системе координат компонента.

GetMetricLength – Метрическая длина кривой

Интерфейс...

Синтаксис Automation:

double GetMetricLength(double StartParam, double EndParam);

Синтаксис COM:

HRESULT GetMetricLength(double StartParam, double EndParam, double * Result ;

Возвращаемое значение:

- длина кривой.

Входные параметры:

StartParam	- начальное значение параметра T в параметрическом представлении кривой,
EndParam	- конечное значение параметра T в параметрическом представлении кривой.

GetNormal – Получить нормаль

Интерфейс...

Синтаксис Automation:

BOOL GetNormal(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetNormal(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y, Z	- вектор нормали.

GetPoint – Получить точку на поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y, Z	- координаты точки на поверхности.

GetTangentVectorU – Получить касательный вектор по U

Интерфейс...

Синтаксис Automation:

BOOL GetTangentVectorU(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetTangentVectorU(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result) ;

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y, Z	- касательный вектор по U.

GetTangentVectorV – Получить касательный вектор по V

Интерфейс...

Синтаксис Automation:

BOOL GetTangentVectorV(double ParamU, double ParamV, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetTangentVectorV(double ParamU, double ParamV, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ParamU, ParamV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

X, Y, Z	- касательный вектор по V.

NearPointProjection – Получить ближайшую проекцию точки на поверхность

Интерфейс...

Синтаксис Automation:

BOOL NearPointProjection(double X, double Y, double Z, double * ParamU, double * ParamV, BOOL Ext);

Синтаксис COM:

HRESULT NearPointProjection(double X, double Y, double Z, double * ParamU, double * ParamV, BOOL Ext, BOOL * Result ;

Возвращаемое значение:

TRUE	- если проекция точки попала на саму поверхность или если ext равен TRUE,
FALSE	- если проекция точки не попала на саму поверхность.

Входные параметры:

X, Y, Z	- координаты исходной пространственной точки,
Ext	- если проекция точки вне поверхности; - TRUE проецировать на продолжение поверхности, - FALSE найти ближайшую граничную точку поверхности.

Выходные параметры:

ParamU, ParamV	- параметрические координаты точки на поверхности.

Интерфейс ISurfaceContainer

Интерфейс контейнера поверхностей.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObject

ISurfaceContainer

Описание:

Позволяет устанавливать и получать коллекции операций с поверхностями.

Примечание:

Дополнительный интерфейс компонента. Данный интерфейс можно получить у компонента IPart7 посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ISurfaceContainer – свойства

CloudPointsSurfaces – Все поверхности по пласту (облаку) точек, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции поверхностей по пласту (облаку) точек ICloudPointsSurfaces

Синтаксис Automation:

CloudPointsSurfaces = Object.CloudPointsSurfaces	Получить свойство (*)
CloudPointsSurfaces = Object.GetCloudPointsSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_CloudPointsSurfaces(&CloudPointsSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции поверхностей по пласту (облаку) точек, входящих в состав объекта.
2. Свойство доступно только для чтения.

EquidistantSurfaces – Все эквидистантные поверхности, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции эквидистантных поверхностей IEquidistantSurfaces

Синтаксис Automation:

EquidistantSurfaces = Object.EquidistantSurfaces	Получить свойство (*)
EquidistantSurfaces = Object.GetEquidistantSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_EquidistantSurfaces(&EquidistantSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции эквидистантных поверхностей, входящих в состав объекта.
2. Свойство доступно только для чтения.

EvolutionSurfaces – Коллекция поверхностей выдавливания

Интерфейс...

Тип данных: Указатель на интерфейс IEvolutions.

Синтаксис Automation:

EvolutionSurfaces = Object.EvolutionSurfaces	Получить свойство (*)
EvolutionSurfaces = Object.GetEvolutionSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_EvolutionSurfaces(&EvolutionSurfaces)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ExtensionSurfaces – Все операции продления поверхности, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций продления поверхности IExtensionSurfaces

Синтаксис Automation:

ExtensionSurfaces = Object.ExtensionSurfaces	Получить свойство (*)
ExtensionSurfaces = Object.GetExtensionSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_ExtensionSurfaces(&ExtensionSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции операций продления поверхности, входящих в состав объекта.
2. Свойство доступно только для чтения.

ExtrusionSurfaces – Все поверхности выдавливания, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции поверхностей выдавливания IExtrusions

Синтаксис Automation:

ExtrusionSurfaces = Object.ExtrusionSurfaces	=	Получить свойство (*)
ExtrusionSurfaces = Object.GetExtrusionSurfaces()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ExtrusionSurfaces(&ExtrusionSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции поверхностей выдавливания, входящих в состав объекта.
2. Свойство доступно только для чтения.

FaceRemovers – Все операции удаления граней, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций удаления граней IFaceRemovers.

Синтаксис Automation:

FaceRemovers = iObject.FaceRemovers( );	=	Получить свойство (*)
FaceRemovers = iObject.GetFaceRemovers();	=	Получить свойство (**)

Синтаксис COM:

iObject->get_FaceRemovers(&FaceRemovers)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить интерфейс коллекции операций удаления граней, входящих в состав объекта.
2. Свойство доступно только для чтения.

ImportedSurfaces – Все импортированные поверхности, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции импортированных поверхностей IImportedSurfaces

Синтаксис Automation:

ImportedSurfaces = Object.ImportedSurfaces	Получить свойство (*)
ImportedSurfaces = Object.GetImportedSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_ImportedSurfaces(&ImportedSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции импортированных поверхностей, входящих в состав объекта.
2. Свойство доступно только для чтения.

JointSurfaces – Коллекция поверхностей соединения

Интерфейс...

Тип данных: Указатель на интерфейс IJointSurfaces

Синтаксис Automation:

JointSurfaces = Object.JointSurfaces	Получить свойство (*)
JointSurfaces = Object.GetJointSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_JointSurfaces(&JointSurfaces)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LoftSurfaces – Коллекция поверхностей по сечениям

Интерфейс...

Тип данных: Указатель на интерфейс ILofts.

Синтаксис Automation:

LoftSurfaces = Object.LoftSurfaces	Получить свойство (*)
LoftSurfaces = Object.GetLoftSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_LoftSurfaces(&LoftSurfaces)	Получить свойство

Примечание:

Свойство доступно только для чтения.

MeshPointsSurfaces – Все поверхности по сети точек, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции поверхностей по сети точек IMeshPointsSurfaces

Синтаксис Automation:

MeshPointsSurfaces = Object.MeshPointsSurfaces	Получить свойство (*)
MeshPointsSurfaces = Object.GetMeshPointsSurfaces()	Получить свойство (**)

Синтаксис COM:

Object.get_MeshPoint sSurfaces(&MeshPointsSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции поверхностей по сети точек, входящих в состав объекта.
2. Свойство доступно только для чтения.

NurbsSurfaces – Все NURBS-поверхности, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции NURBS-поверхностей INurbsSurfaces.

Синтаксис Automation:

NurbsSurfaces = iObject.NurbsSurfaces( );	Получить свойство (*)
NurbsSurfaces = iObject.GetNurbsSurfa ces();	Получить свойство (**)

Синтаксис COM:

iObject- >get_NurbsSurfaces(&NurbsSurfaces)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить интерфейс коллекции NURBS-поверхностей, входящих в состав объекта.
2. Свойство доступно только для чтения.

NurbsSurfacesByCurvesMeshs – Коллекция поверхностей по сети кривых

Интерфейс...

Тип данных: Указатель на интерфейс INurbsSurfacesByCurvesMeshs

Синтаксис Automation:

NurbsSurfacesByCurvesMeshs = Получить свойство (*)
Object.NurbsSurfacesByCurvesMeshs
NurbsSurfacesByCurvesMeshs = Получить свойство (**)
Object.GetNurbsSurfacesByCurvesMeshs()

Синтаксис COM:

Object.get_NurbsSurfacesByCurvesMeshs(Получить свойство
&NurbsSurfacesByCurvesMeshs)

Примечание:

Свойство доступно только для чтения.

RestoredSurfaces – Коллекция операций восстановленная поверхность

Интерфейс...

Тип данных: Указатель на интерфейс IRestoredSurfaces

Синтаксис Automation:

RestoredSurfaces = Object.RestoredSurfaces Получить свойство (*)
RestoredSurfaces = Получить свойство (**)
Object.GetRestoredSurfaces()

Синтаксис COM:

Object.get_RestoredSurfaces(Получить свойство
&RestoredSurfaces)

Примечание:

Свойство доступно только для чтения.

RotatedSurfaces – Все поверхности вращения, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции поверхностей вращения IRotateds

Синтаксис Automation:

RotatedSurfaces = Получить свойство (*)
Object.RotatedSurface
s

RotatedSurfaces = Object.GetRotatedSurfaces()		Получить свойство (**)

Синтаксис COM:

Object.get_RotatedSurfaces(&RotatedSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции поверхностей вращения, входящих в состав объекта.
2. Свойство доступно только для чтения.

RuledSurfaces – Коллекция линейчатых поверхностей

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций создания линейчатой поверхности IRuledSurfaces

Синтаксис Automation:

RuledSurfaces = Object.RuledSurfaces		Получить свойство (*)
RuledSurfaces = Object.GetRuledSurfaces()		Получить свойство (**)

Синтаксис COM:

Object.get_RuledSurfaces(&RuledSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию всех линейчатых поверхностей.
2. Свойство доступно только для чтения.

SurfacePatches – Все заплатки, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции заплаток ISurfacePatches

Синтаксис Automation:

SurfacePatches = iObject.SurfacePatches ();	Получить свойство (*)
SurfacePatches = iObject.GetSurfacePatches();	Получить свойство (**)

Синтаксис COM:

iObject->get_SurfacePatches(&SurfacePatches)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить интерфейс коллекции заплаток, входящих в состав объекта.
2. Свойство доступно только для чтения.

SurfaceSewers – Все операции сшивки поверхностей, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций сшивки поверхностей ISurfaceSewers

Синтаксис Automation:

SurfaceSewers = iObject.SurfaceSewers ();	Получить свойство (*)
SurfaceSewers = iObject.GetSurfaceSewers();	Получить свойство (**)

Синтаксис COM:

iObject->get_SurfaceSewers(&SurfaceSewers)	Получить свойство (*)

Примечание:

1. Свойство позволяет получить интерфейс коллекции операций сшивки поверхностей, входящих в состав объекта.
2. Свойство доступно только для чтения.

TrimmedSurfaces – Все операции усечения поверхности, входящие в состав данного объекта

Интерфейс...

Тип данных: указатель на интерфейс коллекции операций усечения поверхности ITrimmedSurfaces

Синтаксис Automation:

TrimmedSurfaces = Object.TrimmedSurfaces	=	Получить свойство (*)
TrimmedSurfaces = Object.GetTrimmedSurfaces()	=	Получить свойство (**)

Синтаксис COM:

Object.get_TrimmedSurfaces(&TrimmedSurfaces)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс коллекции операций усечения поверхности, входящих в состав объекта.
2. Свойство доступно только для чтения.

Интерфейс ICloudPointsSurfaces

Справка системы КОМПАС...

`kompas.chm::/1120_118_8_Pov_po_plastu.htm`

Коллекция поверхностей по пласту (облаку) точек.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - ICloudPointsSurfaces

ICloudPointsSurfaces – свойства

CloudPointsSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ICloudPointsSurface

Синтаксис Automation:

CloudPointsSurface Object.CloudPointsSurface(Index)	=	Получить свойство (*)
CloudPointsSurface Object.GetCloudPointsSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_CloudPointsSurface(&CloudPointsSurface)	Index,	Получить свойство

Свойство позволяет получить поверхность по облаку точек из коллекции.

Примечание:

Свойство доступно только для чтения.

ICloudPointsSurfaces – методы

Add – Добавить новый элемент

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ICloudPointsSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс поверхности по облаку точек ICloudPointsSurface

Load – Прочитать точки поверхности из файла

Интерфейс...

Синтаксис Automation:

LPDISPATCH Load(BSTR FileName);

Синтаксис COM:

HRESULT Load(BSTR FileName, ICloudPointsSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс поверхности по облаку точек ICloudPointsSurface

Входные параметры:

FileName - имя файла

Метод позволяет создать поверхность по координатам точек, загруженным из файла.

Интерфейс IEquidistantSurfaces

Справка системы КОМПАС...

kompas.chm::/1139_119_11_Ekvidistanta_pov.htm

Коллекция эквидистант поверхности

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IEquidistantSurfaces

Описание:

Данный интерфейс можно получить у интерфейса ISurfaceContainer с помощью свойства ISurfaceContainer::EquidistantSurfaces .

IEquidistantSurfaces – свойства

EquidistantSurface – Возвращает эквидистанту поверхности, заданную по индексу

Интерфейс...

Тип данных: указатель на интерфейс IEquidistantSurface

Синтаксис Automation:

EquidistantSurface Object.EquidistantSurface(Index)	=	Получить свойство (*)
EquidistantSurface Object.GetEquidistantSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_EquidistantSurface(&EquidistantSurface)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IEquidistantSurfaces – методы

Add – Создать эквидистанту поверхности (добавить в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEquidistantSurface * * Result);

Возвращаемое значение:

Указатель на интерфейс эквидистанты поверхности IEquidistantSurface.

Интерфейс IExtensionSurfaces

Справка системы КОМПАС...

`kompas.chm::/1169_120_13_Prodlenie_pov.htm`

Коллекция операций продления поверхности.

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IExtensionSurfaces

IExtensionSurfaces – свойства

ExtensionSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IExtensionSurface

Синтаксис Automation:

ExtensionSurface Object.ExtensionSurface(Index)	=	Получить свойство (*)
ExtensionSurface Object.GetExtensionSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ExtensionSurface(&ExtensionSurface)	Index,	Получить свойство

Свойство позволяет получить указатель на операцию продления поверхности.

Примечание:

Свойство доступно только для чтения.

IExtensionSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IExtensionSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс операции продления поверхности IExtensionSurface.

Интерфейс IFaceRemovers

Интерфейс коллекции операций удаления граней.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

IFaceRemovers

Описание:

Интерфейс позволяет получать и создавать операции удаления граней.

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера поверхностей ISurfaceContainer::SurfacePatches.

IFaceRemovers – свойства

FaceRemover – Операция удаления грани, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс IFaceRemover

Синтаксис Automation:

FaceRemover = Получить свойство (*)
iObject.FaceRemover(Index);

FaceRemover = iObject.GetFaceRemover(Index);		Получить свойство (**)

Синтаксис COM:

iObject->get_FaceRemover(Index, &FaceRemover)	Получить свойство

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

IFaceRemovers – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(IFaceRemover ** Result);

Возвращаемое значение:

	- Указатель на интерфейс операции удаления грани IFaceRemover.

Интерфейс IImportedSurfaces

Справка системы КОМПАС...

`kompas.chm::/884_Glava103_Poverkhnosti.htm`

Коллекция импортированных поверхностей.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IImportedSurfaces

IImportedSurfaces – свойства

ImportedSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IImportedSurface

Синтаксис Automation:

ImportedSurface Object.ImportedSurface(Index)	=	Получить свойство (*)
ImportedSurface Object.GetImportedSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ImportedSurface(&ImportedSurface)	Index,	Получить свойство

Свойство позволяет получить импортированную поверхность из коллекции.

Примечание:

Свойство доступно только для чтения.

IImportedSurfaces – методы

Add – Добавить новый элемент

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IImportedSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс IImportedSurface

Load – Прочитать поверхность из файла

Интерфейс...

Синтаксис Automation:

VARIANT Load(BSTR FileName,
BOOL SewSurfaces);

Синтаксис COM:

HRESULT Load(BSTR FileName,
BOOL SewSurfaces,
VARIANT * Result);

Возвращаемое значение:

Массив VT_ARRAY | VT_DISPATCH поверхностей;

Входные параметры:

FileName	- имя файла,
SewSurfaces	- сшивать поверхности.

Метод позволяет загрузить поверхности из файла .igs или .sat.

Интерфейс IMeshPointsSurfaces

Справка системы КОМПАС...

kompas.chm::/1092_114_6_Pov_po_seti_toch.htm

Коллекция поверхностей по сети точек

Иерархия:

IDispatch
- IKompasAPIObject
  - IKompasCollection
    - IModelObjects
      - IMeshPointsSurfaces

IMeshPointsSurfaces – свойства

MeshPointsSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IMeshPointsSurface

Синтаксис Automation:

MeshPointsSurface Object.MeshPointsSurface(Index)	=	Получить свойство (*)
MeshPointsSurface Object.GetMeshPointsSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_MeshPointsSurface(&MeshPointsSurface)	Index,	Получить свойство

Свойство позволяет получить поверхность по сети точек из коллекции.

Примечание:

Свойство доступно только для чтения.

IMeshPointsSurfaces – методы

Add – Добавить новый элемент

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IMeshPointsSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс поверхности по сети точек IMeshPointsSurface

Load – Прочитать поверхность из файла

Интерфейс...

Синтаксис Automation:

LPDISPATCH Load(BSTR FileName);

Синтаксис COM:

HRESULT Load(BSTR FileName, IMeshPointsSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс поверхности по сети точек IMeshPointsSurface.

Входные параметры:

FileName - имя файла.

Метод позволяет создать поверхность по координатам точек, загруженным из файла.

Интерфейс INurbsSurfaces

Интерфейс коллекции NURBS-поверхностей.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

INurbsSurfaces

Описание:

Позволяет создавать и получать NURBS-поверхности.

Примечание

Данный интерфейс можно получить у контейнера поверхностей ISurfaceContainer::NurbsSurfaces.

INurbsSurfaces – свойства

NurbsSurface – Nurbs-поверхность, заданная по имени или индексу

Интерфейс..

Тип данных: указатель на интерфейс Nurbs-поверхности INurbsSurface.

Синтаксис Automation:

NurbsSurface Object.NurbsSurface(Index)	=	Получить свойство (*)
NurbsSurface Object.GetNurbsSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_NurbsSurface(Index, &NurbsSurface)	Получить свойство

Возвращает Nurbs-поверхность по имени или индексу

Примечание:

Свойство доступно только для чтения.

INurbsSurfaces – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add()

Синтаксис COM:

HRESULT Add(INurbsSurface ** Result);

Возвращаемое значение:

- указатель на интерфейс Nurbs-поверхности INurbsSurface

Интерфейс IPlanes3D

Коллекция плоскостей.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IPlanes3D

IPlanes3D – свойства

Plane3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: указатель на интерфейс IPlane3D

Синтаксис Automation:

Plane3D = Object.Plane3D(Index)	Получить свойство (*)
Plane3D = Object.GetPlane3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Plane3D(Index, &Plane3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IPlanes3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(ksObj3dTypeEnum PlaneType);

Синтаксис COM:

HRESULT Add(ksObj3dTypeEnum PlaneType, IPlane3D ** Result);

Возвращаемое значение:

- указатель на интерфейс плоскости IPlane3D.

Входные параметры:

PlaneType	- тип плоскости.

Интерфейс IRuledSurfaces

Справка системы КОМПАС...

kompas.chm::/1115_115_8_Linejch_pov.htm

Коллекция операций создания линейчатой поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IRuledSurfaces

Примечание:

Данный интерфейс можно получить у интерфейса ISurfaceContainer с помощью свойства ISurfaceContainer::RuledSurfaces.

IRuledSurfaces – свойства

RuledSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IRuledSurface

Синтаксис Automation:

RuledSurface = Object.RuledSurface(Index)	Получить свойство (*)
RuledSurface = Object.GetRuledSurface(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_RuledSurface(Index, &RuledSurface)	Получить свойство

Свойство позволяет получить указатель на интерфейс операции создания линейчатой поверхности.

Примечание:

Свойство доступно только для чтения.

IRuledSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRuledSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс создания линейчатой поверхности IRuledSurface.

Интерфейс ISplitLines

Интерфейс коллекции элементов "линия разъема".

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

ISplitLines

Описание:

Интерфейс позволяет получить элементы "линия разъема".

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера вспомогательной геометрии IAuxiliaryGeomContainer::SplitLines.

ISplitLines – свойства

SplitLine – Линия разъема, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISplitLine

Синтаксис Automation:

```
SplitLine = iObject.SplitLine( Получить свойство (* )
Index );
SplitLine = iObject.GetSplitLine( Получить свойство (**)
Index );
```

Синтаксис COM:

```
iObject->get_SplitLine( Index, Получить свойство
&SplitLine )
```

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ISplitLines – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(ISplitLine ** Result);

Возвращаемое значение:

- Указатель на интерфейс линии разъема ISplitLine.

Интерфейс ISurfacePatches

Интерфейс коллекции элементов "заплатка".

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

ISurfacePatches

Описание:

Интерфейс позволяет получить элементы "заплатка".

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера поверхностей ISurfaceContainer::SurfacePatches.

ISurfacePatches – свойства

SurfacePatch – Заплатка, заданная по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISurfacePatch

Синтаксис Automation:

SurfacePatch = Получить свойство (*)
iObject.SurfacePatch(Index);
SurfacePatch = Получить свойство (**)
iObject.GetSurfacePatch(Index);

Синтаксис COM:

iObject->get_SurfacePatch(Index, &SurfacePatch)	Получить свойство

Примечание:

1. В качестве индекса может использоваться индекс в коллекции и reference объекта.
2. Свойство доступно только для чтения.

ISurfacePatches – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add;

Синтаксис COM:

HRESULT Add(ISurfacePatch ** Result);

Возвращаемое значение:

- Указатель на интерфейс заплатки ISurfacePatch.

Интерфейс ISurfaceSewers

Интерфейс коллекции операций сшивки поверхностей.

Иерархия:

IKompasAPIObject

IKompasCollection

IModelObjects

ISurfaceSewers

Описание:

Позволяет получать и создавать объект операции сшивки поверхностей.

Примечание:

Получить интерфейс коллекции можно, используя свойство контейнера поверхностей ISurfaceContainer::SurfaceSewers.

ISurfaceSewers – свойства

SurfaceSewer – Указатель на элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISurfaceSewer

Синтаксис Automation:

SurfaceSewer = iObject.SurfaceSewer(Index);	=	Получить свойство (*)
SurfaceSewer = iObject.GetSurfaceSewer(Index);	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SurfaceSewer(Index, &SurfaceSewer)	Получить свойство

Входные параметры:

Index	- индекс объекта.

Примечание:

1. Свойство доступно только для чтения.
2. В качестве индекса может использоваться индекс в коллекции и reference объекта.

ISurfaceSewers – методы

Add – Создать новый элемент и добавить его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ISurfaceSewer ** Result);

Входные параметры:

DimType	- тип линии-выноски.

Типы линий-выносок:

o3d_leader3D	86	Линия-выноска 3D
o3d_markLeader3D	87	Знак маркировки 3D
o3d_positionLeader3D	89	Обозначение позиции 3D
o3d_brandLeader3D	90	Знак клеймения 3D

Возвращаемое значение:

- Указатель на интерфейс линии выноски IBaseLeader3D.

Возвращаемое значение:

- Указатель на интерфейс присоединительной точки IConjunctivePointI.

Примечание:

1. Метод позволяет создать новый интерфейс присоединительной точки.
2. После получения нового интерфейса нужно задать параметры и вызвать метод IModelObject::Update.

Интерфейс ITrimmedSurfaces

Справка системы КОМПАС...

kompas.chm::/1131_118_11_Usech_pov.htm

Коллекция операций усечения поверхности

Иерархия:

IDispatch
IKompasAPIObject
IKompasCollection
IModelObjects
ITrimmedSurfaces

ITrimmedSurfaces – свойства

TrimmedSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ITrimmedSurface

Синтаксис Automation:

TrimmedSurface Object.TrimmedSurface(Index)	=	Получить свойство (*)
TrimmedSurface Object.GetTrimmedSurface(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_TrimmedSurface(&TrimmedSurface)	Index,	Получить свойство

Свойство позволяет получить указатель на интерфейс операции усечения поверхности.

Примечание:

Свойство доступно только для чтения.

ITrimmedSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ITrimmedSurface ** Result);

Возвращаемое значение:

Указатель на интерфейс операции усечения поверхности ITrimmedSurface

Интерфейс IRestoredSurfaces

Коллекция операций Восстановленная поверхность.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IRestoredSurfaces

Примечание

Получить интерфейс коллекции операций восстановленная поверхность можно, используя свойство контейнера трехмерных объектов ISurfaceContainer::RestoredSurfaces.

IRestoredSurfaces – свойства

RestoredSurface – Возвращает элемент, заданный по индексу

Интерфейс...

Тип данных: Указатель на интерфейс IRestoredSurface

Синтаксис Automation:

RestoredSurface = Получить свойство (*)
Object.RestoredSurface(Index)
RestoredSurface = Получить свойство (**)
Object.GetRestoredSurface(Index)

Синтаксис COM:

Object.get_RestoredSurface(Index, Получить свойство
&RestoredSurface)

Входные параметры:

VARIANT Index - индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IRestoredSurfaces – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IRestoredSurface * * Result);

Возвращаемое значение:

- Указатель на интерфейс IRestoredSurface.

Примечания:

1. Метод позволяет создать новый интерфейс операции восстановления поверхности.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс ICloudPointsSurface

Справка системы КОМПАС...

kompas.chm::/1120_118_8_Pov_po_plastu.htm

Интерфейс параметров поверхности по пласту (облаку) точек

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ICloudPointsSurface

ILocalCSObject

Интерфейс можно получить у интерфейса коллекции поверхностей по пласту (облаку точек) с помощью свойства ICloudPointsSurfaces::CloudPointsSurface или метода ICloudPointsSurfaces::CloudPointsSurface.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

ICloudPointsSurface – свойства

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject(Index)	Получить свойство (*)
Object.AssociationObject(Index) = AssociationObject	Установить свойство (*)
AssociationObject = Object.GetAssociationObject(Index)	Получить свойство (**)
Object.SetAssociationObject(Index, AssociationObject)	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(Index, &AssociationObject)	Получить свойство
Object.put_AssociationObject(Index, AssociationObject)	Установить свойство

Входные параметры:

long Index - индекс точки, для которой нужно установить опорный объект

BuildingType – Тип поверхности по пласту (облаку) точек

Интерфейс...

Тип данных: из перечисления ksCloudPointsSurfaceBuildingTypeEnum

Синтаксис Automation:

BuildingType = Object.BuildingType		Получить свойство (*)
Object.BuildingType = BuildingType		Установить свойство (*)
BuildingType = Object.GetBuildingType()	=	Получить свойство (**)
Object.SetBuildingType(BuildingType)		Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Свойство позволяет устанавливать и получать способ построения поверхности по пласту (облаку) точек.

CheckSelfIntersection – Проверка самопересечений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CheckSelfIntersection = Object.CheckSelfIntersection		Получить свойство (*)
Object.CheckSelfIntersection = CheckSelfIntersection		Установить свойство (*)
CheckSelfIntersection = Object.GetCheckSelfIntersection()	=	Получить свойство (**)
Object.SetCheckSelfIntersection(CheckSelfIntersection)		Установить свойство (**)

Синтаксис COM:

Object.get_CheckSelfIntersection(&CheckSelfIntersection)	Получить свойство
Object.put_CheckSelfIntersection(CheckSelfIntersection)	Установить свойство

Свойство позволяет включать и выключать проверку самопересечений.

CloudType – Способ распознавания сети точек

Интерфейс...

Тип данных: из перечисления ksCloudTypeEnum

Синтаксис Automation:

```
CloudType = Получить свойство (* )
Object.CloudType
Object.CloudType = Установить свойство (* )
CloudType
CloudType = Получить свойство (**)
Object.GetCloudType()
Object.SetCloudType( Установить свойство (**)
CloudType )
```

Синтаксис COM:

```
Object.get_CloudType( Получить свойство
&CloudType )
Object.put_CloudType( Установить свойство
CloudType )
```

CloudLCS – СК объекта

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

```
CloudLCS = Получить свойство (* )
Object.CloudLCS
Object.CloudLCS = Установить свойство (* )
CloudLCS
CloudLCS = Получить свойство (**)
Object.GetCloudLCS()
Object.SetCloudLCS( Установить свойство (**)
CloudLCS )
```

Синтаксис COM:

```
Object.get_CloudLCS( Получить свойство
&CloudLCS )
Object.put_CloudLCS( Установить свойство
CloudLCS )
```

Свойство позволяет устанавливать и получать локальную систему координат поверхности.

Degree – Степень сплайна

Интерфейс...

Тип данных: long

Синтаксис Automation:

Degree Object.Degree	=	Получить свойство (*)
Object.Degree Degree	=	Установить свойство (*)
Degree Object.GetDegree()	=	Получить свойство (**)
Object.SetDegree(Degree)		Установить свойство (**)

Синтаксис COM:

Object.get_Degree(&Degree)	Получить свойство
Object.put_Degree(Degree)	Установить свойство

FixedPosition – Фиксировать положение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FixedPosition Object.FixedPosition	=	Получить свойство (*)
Object.FixedPosition FixedPosition	=	Установить свойство (*)
FixedPosition Object.GetFixedPositio n()	=	Получить свойство (**)
Object.SetFixedPositio n(FixedPosition)		Установить свойство (**)

Синтаксис COM:

Object.get_FixedPositi on(&FixedPosition)	Получить свойство
Object.put_FixedPositi on(FixedPosition)	Установить свойство

Points – Массив точек

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_R8

Синтаксис Automation:

```
Points = Object.Points      Получить свойство (* )
Object.Points = Points      Установить свойство (* )
Points = Object.GetPoints() Получить свойство (**)
Object.SetPoints( Points )  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Points( &Points )      Получить свойство
Object.put_Points( Points )       Установить свойство
```

Свойство позволяет устанавливать и получать массив точек, задающих поверхность.

PointsCount – Количество точек

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PointsCount = Object.PointsCount  Получить свойство (* )
PointsCount = Object.GetPointsCount() Получить свойство (**)
```

Синтаксис COM:

```
Object.get_PointsCount( &PointsCount )  Получить свойство
```

Примечание:

Свойство доступно только для чтения.

PointParameters – Интерфейс параметров точки поверхности

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

PointParameters = Object.PointParameters(Index)	=	Получить свойство (*)
PointParameters = Object.GetPointParameters(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointParameters(Index, &PointParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PointType – Тип параметров построения точки поверхности

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointType = Object.PointType(Index)	Получить свойство (*)
Object.PointType(Index) = PointType	Установить свойство (*)
PointType = Object.GetPointType(Index)	Получить свойство (**)
Object.SetPointType(Index, PointType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointType(Index, &PointType)	Получить свойство
Object.put_PointType(Index, PointType)	Установить свойство

Входные параметры:

long Index - индекс точки.

ICloudPointsSurface – методы

AddPoint – Добавить точку

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddPoint(double X,
double Y,
double Z,

ksPoint3DTypeEnum PVal,
IModelObject * AssociationObject);

Синтаксис COM:

HRESULT AddPoint(double X,
double Y,
double Z,
ksPoint3DTypeEnum PVal,
IModelObject * AssociationObject,
IKompasAPIObject ** Result);

Возвращаемое значение:

Указатель на интерфейс параметров точки

Выходные параметры:

X, Y, Z	- координаты точки,
PVal	- тип параметров точки,
AssociationObject	- опорный объект для точки.

AddPoints – Добавить точки

Интерфейс...

Синтаксис Automation:

BOOL AddPoints(VARIANT Points);

Синтаксис COM:

HRESULT AddPoints(VARIANT Points,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Points: массив VT_ARRAY | VT_R8 координат точек

ClearPoints – Очистить список рядов точек

Интерфейс...

Синтаксис Automation:

BOOL ClearPoints();

Синтаксис COM:

HRESULT ClearPoints(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Метод позволяет удалить все точки поверхности.

DeletePoint – Удалить точку

Интерфейс...

Синтаксис Automation:

BOOL DeletePoint(long Index);

Синтаксис COM:

HRESULT DeletePoint(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index - индекс удаляемой точки.

GetPoint – Получить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long Index,
double * X,
double * Y,
double * Z);

Синтаксис COM:

HRESULT GetPoint(long Index,
double * X,
double * Y,
double * Z,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index - индекс точки

Выходные параметры:

X, Y, Z - координаты точки

Метод позволяет получить координаты точки.

SetPoint – Установить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(long Index, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(long Index, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Index	- индекс точки,
X, Y, Z	- координаты точки.

Метод позволяет установить параметры точки.

Интерфейс IEquidistantSurface

Справка системы КОМПАС...

`kompas.chm::/1139_119_11_Ekvidistanta_pov.htm`

Интерфейс операции построения эквидистанты поверхности

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IEquidistantSurface

IEquidistantSurface – свойства

BaseSurface – Базовая поверхность для эквидистантной поверхности (грань или совокупность граней)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
BaseSurface                    =   Получить свойство (* )
Object.BaseSurface
Object.BaseSurface             =   Установить свойство (* )
BaseSurface
BaseSurface                    =   Получить свойство (**)
Object.GetBaseSurface()
Object.SetBaseSurface(             Установить свойство (**)
BaseSurface )
```

Синтаксис COM:

```
Object.get_BaseSurface(                Получить свойство
&BaseSurface )
Object.put_BaseSurface(                Установить свойство
BaseSurface )
```

Примечание:

В качестве базовой поверхности могут использоваться следующие объекты:

- грань тела или поверхности;
- связная совокупность граней одного тела или поверхности.

Связная совокупность граней — множество граней, каждая из которых имеет общее ребро хотя бы еще с одной гранью этого множества, причем одно ребро одновременно принадлежит не более чем двум граням.

Direction – Направление построения эквидистантной поверхности относительно базовой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction          Получить свойство (* )
Object.Direction = Direction          Установить свойство (* )
Direction = Object.GetDirection()     Получить свойство (**)
Object.SetDirection( Direction )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction )    Получить свойство
Object.put_Direction(  Direction )    Установить свойство
```

Свойство позволяет получать и устанавливать направление эквидистантной поверхности относительно базовой поверхности.

Distance – Расстояние от базовой поверхности до эквидистантной поверхности

Интерфейс...

Тип данных: double

Синтаксис Automation:

Distance = Object.Distance	Получить свойство (*)
Object.Distance = Distance	Установить свойство (*)
Distance = Object.GetDistance()	Получить свойство (**)
Object.SetDistance(Distance)	Установить свойство (**)

Синтаксис COM:

Object.get_Distance(&Distance)	Получить свойство
Object.put_Distance(Distance)	Установить свойство

Свойство позволяет получать и устанавливать расстояние от базовой поверхности до эквидистантной.

Интерфейс IExtensionSurface

Справка системы КОМПАС...

`kompas.chm::/1169_120_13_Prodlenie_pov.htm`

Интерфейс операции продления поверхности

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IExtensionSurface

IExtensionSurface – свойства

BuildingVectorParameters – Параметры вектора

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

BuildingVectorParameters = Object.BuildingVectorParameters	Получить свойство (*)
BuildingVectorParameters = Object.GetBuildingVectorParameters()	Получить свойство (**)

Синтаксис COM:

Object.get_BuildingVectorParameters(&BuildingVectorParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DirObject – Объект, определяющий направление для типа построения По направлению

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

DirObject= Object.DirObject	Получить свойство (*)
Object.DirObject= DirObject	Установить свойство (*)
DirObject = Object.GetDirObject()	Получить свойство (**)
Object.SetDirObject(DirObject)	Установить свойство (**)

Синтаксис COM:

Object.get_DirObject(&DirObject)	Получить свойство
Object.put_DirObject(DirObject)	Установить свойство

Edges – Массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Object.Edges = Edges	Установить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)
Object.SetEdges(Edges)	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство
Object.put_Edges(Edges)	Установить свойство

ExtensionLimitType – Способ ограничения

Интерфейс...

Тип данных: из перечисления ksExtensionLimitTypeEnum

Синтаксис Automation:

ExtensionLimitType = Object.ExtensionLimitType	=	Получить свойство (*)
Object.ExtensionLimitType = ExtensionLimitType	=	Установить свойство (*)
ExtensionLimitType = Object.GetExtensionLimitType()	=	Получить свойство (**)
Object.SetExtensionLimitType(ExtensionLimitType)		Установить свойство (**)

Синтаксис COM:

Object.get_ExtensionLimitType(&ExtensionLimitType)	Получить свойство
Object.put_ExtensionLimitType(ExtensionLimitType)	Установить свойство

ExtensionType – Тип продления поверхности

Интерфейс...

Тип данных: из перечисления ksExtensionTypeEnum

Синтаксис Automation:

ExtensionType = Object.ExtensionType	=	Получить свойство (*)
Object.ExtensionType = ExtensionType	=	Установить свойство (*)
ExtensionType = Object.GetExtensionType()	=	Получить свойство (**)
Object.SetExtensionType(ExtensionType)		Установить свойство (**)

Синтаксис COM:

Object.get_ExtensionType(&ExtensionType)	Получить свойство
Object.put_ExtensionType(ExtensionType)	Установить свойство

Length – Длина (расстояние удлинения поверхности)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Length = Object.Length	Получить свойство (*)
Object.Length = Length	Установить свойство (*)
Length = Object.GetLength()	Получить свойство (**)
Object.SetLength(Length)	Установить свойство (**)

Синтаксис COM:

Object.get_Length(&Length)	Получить свойство
Object.put_Length(Length)	Установить свойство

Sense – Сменить направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Sense = Object.Sense	Получить свойство (*)
Object.Sense = Sense	Установить свойство (*)
Sense = Object.GetSense()	Получить свойство (**)
Object.SetSense(Sense)	Установить свойство (**)

Синтаксис COM:

Object.get_Sense(&Sense)	Получить свойство
Object.put_Sense(Sense)	Установить свойство

SideEdges – Боковые ребра

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SideEdges = Object.SideEdges	Получить свойство (*)
Object.SideEdges = SideEdges	Установить свойство (*)
SideEdges = Object.GetSideEdges()	Получить свойство (**)
Object.SetSideEdges(SideEdges)	Установить свойство (**)

Синтаксис COM:

Object.get_SideEdges(&SideEdges)		Получить свойство
Object.put_SideEdges(SideEdges)		Установить свойство

Значения свойства:

TRUE	- как продление исходных боковых ребер,
FALSE	- по нормали к указанным кромкам.

TargetObject – Объект, до которого производится удлинение при способе построения До вершины

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

TargetObject Object.TargetObject	=	Получить свойство (*)
Object.TargetObject TargetObject	=	Установить свойство (**)
TargetObject Object.GetTargetObject()	=	Получить свойство (**)
Object.SetTargetObject(TargetObject)		Установить свойство (**)

Синтаксис COM:

Object.get_TargetObject(&TargetObject)	Получить свойство
Object.put_TargetObject(TargetObject)	Установить свойство

Интерфейс IFaceRemover

Интерфейс операции удаления граней.

Иерархия:

IKompasAPIObject

IModelObject

IFaceRemover

Примечание:

1. Интерфейс можно получить у коллекции операций удаления граней, используя свойство IFaceRemovers::FaceRemover или метод IFaceRemovers::Add.
2. После задания параметров заплатки требуется вызвать метод IModelObject::Update.

IFaceRemover – свойства

Faces – Массив граней в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces(Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces)	Получить свойство
Object.put_Faces(Faces)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать грани операции. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Интерфейс IImportedSurface

Справка системы КОМПАС...

kompas.chm::/884_Glava103_Poverkhnosti.htm

Интерфейс параметров импортированной поверхности

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IImportedSurface

ILocalCSObject

Интерфейс можно получить у интерфейса коллекции импортированных поверхностей с помощью свойства IImportedSurfaces::ImportedSurface или метода IImportedSurfaces::Add.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IImportedSurface – свойства

PointsVCount – Количество точек по V

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PointsVCount = Object.PointsVCount          Получить свойство (* )
PointsVCount = Object.GetPointsVCount()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_PointsVCou                       Получить свойство
nt( &PointsVCount )
```

Примечание:

Свойство доступно только для чтения.

PointsUCount – Количество точек по U

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PointsUCount = Object.PointsUCount          Получить свойство (* )
PointsUCount = Object.GetPointsUCount()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_PointsUCo                        Получить свойство
unt( &PointsUCount )
```

Примечание:

Свойство доступно только для чтения.

IImportedSurface – методы

AddPoint – Добавить точку в создаваемый ряд

Интерфейс...

Синтаксис Automation:

```
BOOL AddPoint( double X,
double Y,
double Z );
```

Синтаксис COM:

HRESULT AddPoint(double X,
double Y,
double Z,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

X, Y, Z	- координаты точки.

Примечание:

Метод должен вызываться между вызовами методов IImportedSurface::BeginPointsSeries и IImportedSurface::EndPointsSeries.

AddPointsSeries – Добавить ряд точек

Интерфейс...

Синтаксис Automation:

BOOL AddPointsSeries(BOOL UV,
long IndexAt,
VARIANT arr);

Синтаксис COM:

HRESULT AddPointsSeries(BOOL UV,
long IndexAt,
VARIANT arr,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Возвращаемое значение:

UV	- TRUE - добавлять по U, FALSE - по V,
IndexAt	- индекс, перед которым добавлять,
arr	- массив VT_ARRAY | VT_R8 координат точек.

BeginPointsSeries – Начать добавление нового ряда точек

Интерфейс...

Синтаксис Automation:

BOOL BeginPointsSeries(BOOL UV,
long IndexAt);

Синтаксис COM:

HRESULT BeginPointsSeries(BOOL UV,
long IndexAt,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

UV	- TRUE - добавлять по U, - FALSE - по V,
IndexAt	- индекс ряда, перед которым добавлять новый ряд.

ClearPointsSeries – Очистить список рядов точек

Интерфейс...

Синтаксис Automation:

BOOL ClearPointsSeries();

Синтаксис COM:

HRESULT ClearPointsSeries(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Метод позволяет удалить все точки поверхности.

DeletePointsSeries – Удалить ряд точек

Интерфейс...

Синтаксис Automation:

BOOL DeletePointsSeries(BOOL UV,
long Index);

Синтаксис COM:

HRESULT DeletePointsSeries(BOOL UV,
long Index,
BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

UV - TRUE - удалять по U,
- FALSE - по V,
Index - индекс удаляемого ряда.

EndPointsSeries – Завершить создание нового ряда точек

Интерфейс...

Синтаксис Automation:

BOOL EndPointsSeries();

Синтаксис COM:

HRESULT EndPointsSeries(BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

GetPoint – Получить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL GetPoint(long UIndex,
long VIndex,
double * X,
double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(long UIndex,
long VIndex,
double * X,
double * Y,
double * Z,
BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

UIndex - индекс точки по U,
VIndex - индекс точки по V.

Выходные параметры:

X, Y,Z - координаты точки.

Метод позволяет получить параметры точки.

GetPoints – Получить массив всех точек

Интерфейс...

Синтаксис Automation:

VARIANT GetPoints(long * UPCount, long * VPCount);

Синтаксис COM:

HRESULT GetPoints(long * UPCount, long * VPCount, VARIANT * Result);

Возвращаемое значение:

Массив VT_ARRAY | VT_R8 точек поверхности

Выходные параметры:

UPCount - количество точек по U,
VPCount - количество точек по V.

SetPoint – Установить параметры точки

Интерфейс...

Синтаксис Automation:

BOOL SetPoint(long UIndex,
long VIndex,
double X,
double Y,
double Z);

Синтаксис COM:

HRESULT SetPoint(long UIndex,
long VIndex,
double X,
double Y,
double Z,
BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,

FALSE - в случае неудачи.

Входные параметры:

UIndex - индекс точки по U,
VIndex - индекс точки по V,
X, Y, Z - координаты точки.

Метод позволяет установить параметры точки.

SetPoints – Установить массив всех точек

Интерфейс...

Синтаксис Automation:

BOOL SetPoints(long UPCount,
long VPCount, VARIANT Points);

Синтаксис COM:

HRESULT SetPoints(long UPCount, long VPCount, VARIANT Points, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

UPCount - количество точек по U,
VPCount - количество точек по V,
Points - массив VT_ARRAY | VT_R8 координат точек.

Метод позволяет установить массив точек поверхности.

Интерфейс IMeshPointsSurface

Справка системы КОМПАС...

`kompas.chm::/1092_114_6_Pov_po_seti_toch.htm`

Интерфейс параметров поверхности по сети точек.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IMeshPointsSurface
ILocalCSObject

Интерфейс можно получить у интерфейса коллекции поверхностей по сети точек с помощью свойства IMeshPointsSurfaces::MeshPointsSurface или метода IMeshPointsSurfaces::Add.

С помощью метода IUnknown::QueryInterface у интерфейса можно получить интерфейс подчиненного объекта ЛСК ILocalCSObject.

IMeshPointsSurface – свойства

AssociationObject – Установить опорный объект для вершины

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject = Object.AssociationObject(VIndex)	UIndex,	Получить свойство (*)
Object.AssociationObject(VIndex) = AssociationObject	UIndex,	Установить свойство (*)
AssociationObject = Object.GetAssociationObject(VIndex)	UIndex,	Получить свойство (**)
Object.SetAssociationObject(VIndex, AssociationObject)	UIndex,	Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(UIndex, VIndex, &AssociationObject)	Получить свойство
Object.put_AssociationObject(UIndex, VIndex, AssociationObject)	Установить свойство

Входные параметры:

long UIndex	- индекс точки по U,
long VIndex	- индекс точки по V.

BuildingType – Тип поверхности по сети точек

Интерфейс...

Тип данных: из перечисления ksMeshPointsSurfaceBuildingTypeEnum

Синтаксис Automation:

```
BuildingType = Object.BuildingType                Получить свойство (* )
Object.BuildingType = BuildingType                Установить свойство (* )
BuildingType                              =       Получить свойство (**)
Object.GetBuildingType()
Object.SetBuildingType( BuildingType )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BuildingType(                          Получить свойство
&BuildingType )
Object.put_BuildingType(  BuildingType            Установить свойство
)
```

Свойство позволяет устанавливать и получать способ построения поверхности по сети точек.

CheckSelfIntersection – Проверка самопересечений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CheckSelfIntersection             =       Получить свойство (* )
Object.CheckSelfIntersection
Object.CheckSelfIntersection      =       Установить свойство (* )
CheckSelfIntersection
CheckSelfIntersection             =       Получить свойство (**)
Object.GetCheckSelfIntersection()
Object.SetCheckSelfIntersection(          Установить свойство (**)
CheckSelfIntersection )
```

Синтаксис COM:

```
Object.get_CheckSelfIntersection(         Получить свойство
&CheckSelfIntersection )
Object.put_CheckSelfIntersection          Установить свойство
(  CheckSelfIntersection )
```

Свойство позволяет включать и выключать проверку самопересечений.

ClosedU – Признак замкнутости поверхности по направлению U

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedU = Object.ClosedU	Получить свойство (*)
Object.ClosedU = ClosedU	Установить свойство (*)
ClosedU = Object.GetClosedU()	Получить свойство (**)
Object.SetClosedU(ClosedU)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedU(&ClosedU)	Получить свойство
Object.put_ClosedU(ClosedU)	Установить свойство

Свойство позволяет замкнуть поверхность в направлении U.

Примечание:

Замыкание по тому или иному направлению возможно, если количество точек в этом направлении - три или больше.

ClosedV – Признак замкнутости поверхности по направлению V

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedV = Object.ClosedV	Получить свойство (*)
Object.ClosedV = ClosedV	Установить свойство (*)
ClosedV = Object.GetClosedV()	Получить свойство (**)
Object.SetClosedV(ClosedV)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedV(&ClosedV)	Получить свойство
Object.put_ClosedV(ClosedV)	Установить свойство

Свойств позволяет замкнуть поверхность в направлении V.

Примечание:

Замыкание по тому или иному направлению возможно, если количество точек в этом направлении - три или больше.

DegreeU – Порядок сплайна по направлению U

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
DegreeU = Object.DegreeU          Получить свойство (* )
Object.DegreeU = DegreeU          Установить свойство (* )
DegreeU                      =    Получить свойство (**)
Object.GetDegreeU()
Object.SetDegreeU( DegreeU )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DegreeU(                    Получить свойство
&DegreeU )
Object.put_DegreeU(                    Установить свойство
DegreeU )
```

Свойство позволяет устанавливать и получать порядок поверхности в направлении U.

DegreeV – Порядок сплайна по направлению V

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
DegreeV = Object.DegreeV               Получить свойство (* )
Object.DegreeV = DegreeV               Установить свойство (* )
DegreeV                      =         Получить свойство (**)
Object.GetDegreeV()
Object.SetDegreeV( DegreeV )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_DegreeV(  &DegreeV          Получить свойство
)
Object.put_DegreeV(  DegreeV )         Установить свойство
```

Свойство позволяет устанавливать и получать порядок поверхности в направлении V.

RowsCount – Количество рядов по направлению V

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
RowsCount = Object.RowsCount           Получить свойство (* )
RowsCount = Object.GetRowsCount()      Получить свойство (**)
```

Синтаксис COM:

```
Object.get_RowsCount( &RowsVCount )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

ColumnsCount – Количество точек в одном ряду по направлению U

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnsCount = Object.ColumnsCount		Получить свойство (*)
ColumnsCount Object.GetColumnsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnsCount(&ColumnsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PointParameters – Интерфейс параметров точки поверхности

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

PointParameters Object.PointParameters(UIndex, VIndex)	=	Получить свойство (*)
PointParameters Object.GetPointParameters(UIndex, VIndex)	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointParameters(UIndex, VIndex, &PointParameters)	Получить свойство

Входные параметры:

long UIndex	- индекс точки по U,

long VIndex	- индекс точки по V.

Примечание:

Свойство доступно только для чтения.

PointType – Тип параметров построения точки поверхности

Интерфейс...

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointType = Object.PointType(UIndex, VIndex)	Получить свойство (*)
Object.PointType(UIndex, VIndex) = PointType	Установить свойство (*)
PointType = Object.GetPointType(UIndex, VIndex)	Получить свойство (**)
Object.SetPointType(UIndex, VIndex, PointType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointType(UIndex, VIndex, &PointType)	Получить свойство
Object.put_PointType(UIndex, VIndex, PointType)	Установить свойство

Входные параметры:

long UIndex	- индекс точки по U,
long VIndex	- индекс точки по V.

IMeshPointsSurface – методы

AddPoint – Добавить точку в создаваемый ряд точек

Интерфейс..

Синтаксис Automation:

LPDISPATCH AddPoint(double X,
double Y,
double Z,
double Weight,
ksPoint3DTypeEnum PointType,
IModelObject * AssociationObject);

Синтаксис COM:

HRESULT AddPoint(double X,
double Y,
double Z,
double Weight,

ksPoint3DTypeEnum PointType,
IModelObject * AssociationObject,
IKompasAPIObject ** Result);

Возвращаемое значение:

Указатель на интерфейс параметров точки

Входные параметры:

X, Y, Z	- координаты точки,
Weight	- вес точки,
PointType	- тип параметров точки,
AssociationObject	- опорный объект.

Примечание:

Метод должен вызываться между вызовами методов IMeshPointsSurface::BeginPointsSeries и IMeshPointsSurface::EndPointsSeries.

AddPointsSeries – Добавить ряд точек

Интерфейс..

Синтаксис Automation:

BOOL AddPointsSeries(BOOL AddNewRow,
long IndexAt,
VARIANT Points,
VARIANT Weights);

Синтаксис COM:

HRESULT AddPointsSeries(BOOL AddNewRow,
long IndexAt,
VARIANT Points,
VARIANT Weights,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

AddNewRow	- TRUE - добавить ряд,
	- FALSE - добавить колонку,
IndexAt	- индекс, перед которым нужно добавить ряд,
Points	- массив VT_ARRAY | VT_R8 координат точек,
Weights	- массив VT_ARRAY | VT_R8 весов точек.

BeginPointsSeries – Начать добавление нового ряда точек

Интерфейс...

Синтаксис Automation:

BOOL BeginPointsSeries(BOOL AddNewRow,
long IndexAt);

Синтаксис COM:

HRESULT BeginPointsSeries(BOOLAddNewRow,
long IndexAt,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

AddNewRow	TRUE - добавлять ряд, FALSE - добавлять колонку,
IndexAt	- индекс ряда, перед которым нужно добавить новый ряд.

ClearPointsSeries – Очистить список рядов точек

Интерфейс..

Синтаксис Automation:

BOOL ClearPointsSeries();

Синтаксис COM:

HRESULT ClearPointsSeries(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Метод позволяет удалить все точки поверхности.

DeletePointsSeries – Удалить ряд точек

Интерфейс..

Синтаксис Automation:

BOOL DeletePointsSeries(BOOL DeleteRow,
long Index);

Синтаксис COM:

HRESULT DeletePointsSeries(BOOL DeleteRow,
long Index,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

DeleteRow	- TRUE - удаляется ряд,
	- FALSE - удаляется колонка,
Index	- индекс удаляемого ряда.

EndPointsSeries – Завершить создание ряда точек

Интерфейс..

Синтаксис Automation:

BOOL EndPointsSeries();

Синтаксис COM:

HRESULT EndPointsSeries(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

GetParams – Получить параметры поверхности

Интерфейс..

Синтаксис Automation:

BOOL GetParams(BOOL * ClosedV,
BOOL * ClosedU,
long * DegreeV,
long * DegreeU,
long * PointsVCount,
long * PointsUCount,
VARIANT * Points,
VARIANT * Weights);

Синтаксис COM:

HRESULT GetParams(BOOL * ClosedV,
BOOL * ClosedU,
long * DegreeV,
long * DegreeU,
long * PointsVCount,
long * PointsUCount,

VARIANT * Points,
VARIANT * Weights,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

ClosedV	признак замкнутости поверхности по V,
ClosedU	признак замкнутости поверхности по U,
DegreeV	порядок сплайна по V,
DegreeU	порядок сплайна по U,
PointsVC ount	количество точек по V,
PointsUC ount	количество точек по U,
Points	массив VT_ARRAY | VT_R8 координат точек,
Weights	массив VT_ARRAY | VT_R8 весов точек.

GetPoint – Параметры точки

Интерфейс..

Синтаксис Automation:

BOOL GetPoint(long UIndex,
long VIndex,
double * X,
double * Y,
double * Z,
double * Weight);

Синтаксис COM:

HRESULT GetPoint(long UIndex,
long VIndex,
double * X,
double * Y,
double * Z,
double * Weight,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

UIndex - индекс точки по U,
VIndex - индекс точки по V.

Выходные параметры:

X, Y, Z - координаты точки,
Weight - вес точки.

Метод позволяет получить параметры точки.

InitParamByFace – Создать по объекту

Интерфейс..

Синтаксис Automation:

BOOL InitParamByFace(IModelObject * Face);

Синтаксис COM:

HRESULT InitParamByFace(IModelObject * Face, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры:

Face - исходная поверхность.

Метод позволяет создать поверхность по сети точек с помощью сплайновой аппроксимации поверхности.

SetParams – Установить параметры поверхности

Интерфейс..

Синтаксис Automation:

BOOL SetParams(BOOL ClosedV,
BOOL ClosedU,
long DegreeV,
long DegreeU,
long PointsVCount,
long PointsUCount,
VARIANT Points,
VARIANT Weights);

Синтаксис COM:

HRESULT SetParams(BOOL ClosedV,
BOOL ClosedU,
long DegreeV,

long DegreeU,
long PointsVCount,
long PointsUCount,
VARIANT Points,
VARIANT Weights,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

ClosedV	- признак замкнутости поверхности по V,
ClosedU	- признак замкнутости поверхности по U,
DegreeV	- порядок сплайна по V,
DegreeU	- порядок сплайна по U,
PointsVCount	- количество точек по V,
PointsUCount	- количество точек по U,
Points	- массив VT_ARRAY | VT_R8 координат точек,
Weights	- массив VT_ARRAY | VT_R8 весов точек.

SetPoint – Параметры точки

Интерфейс..

Синтаксис Automation:

BOOL SetPoint(long UIndex,
long VIndex,
double X,
double Y,
double Z,
double Weight);

Синтаксис COM:

HRESULT SetPoint(long UIndex,
long VIndex,
double X,
double Y,
double Z,
double Weight,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

UIndex	- индекс точки по U,
VIndex	- индекс точки по V,
X, Y, Z	- координаты точки,
Weight	- вес точки.

Метод позволяет установить параметры точки.

Интерфейс INurbsSurface

Интерфейс Nurbs-поверхности.

Иерархия:

IKompasAPIObject

IModelObject

INurbsSurface

Описание:

Интерфейс позволяет создать NURBS-поверхность по заданной поверхности или по параметрам NURBS-поверхности. Есть возможность ограничить поверхность набором контуров.

Примечание:

Интерфейс можно получить с помощью свойства INurbsSurfaces::NurbsSurface и метода INurbsSurfaces::Add.

INurbsSurface – свойства

BoundaryCount – Количество циклов

Интерфейс...

Тип данных:long

Синтаксис Automation:

BoundaryCount = Object.BoundaryCount	=	Получить свойство (*)
BoundaryCount = Object.GetBoundaryCount( )	=	Получить свойство (**)

Синтаксис COM:

Object.get_BoundaryCount	Получить свойство

Свойство позволяет устанавливать и получать количество циклов.

ClosedU – Признак замкнутости поверхности по U

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedU = Object.ClosedU	Получить свойство (*)
Object.ClosedU = ClosedU	Установить свойство (*)
ClosedU = Object.GetClosedU()	Получить свойство (**)
Object.SetClosedU(ClosedU)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedU(&ClosedU)	Получить свойство
Object.put_ClosedU(ClosedU)	Установить свойство

Примечание:

Свойство позволяет получить признак замкнутости поверхности по U.

Свойство позволяет также преобразовать представление поверхности из замкнутого по U представления в разомкнутое по U и наоборот, из разомкнутого представления в замкнутое. См также INurbsSurface::ClosedV.

ClosedV – Признак замкнутости поверхности по V

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ClosedV = Object.ClosedV	Получить свойство (*)
Object.ClosedV = ClosedV	Установить свойство (*)
ClosedV = Object.GetClosedV()	Получить свойство (**)
Object.SetClosedV(ClosedV)	Установить свойство (**)

Синтаксис COM:

Object.get_ClosedV(&ClosedV)	Получить свойство
Object.put_ClosedV(ClosedV)	Установить свойство

Примечание:

Свойство позволяет получить признак замкнутости поверхности по V.

Свойство позволяет также преобразовать представление поверхности из замкнутого по V представления в разомкнутое по V и наоборот, из разомкнутого представления в замкнутое. См. также INurbsSurface::ClosedU.

INurbsSurface – методы

AddBoundary – Добавить границу для NURBS-поверхности

Интерфейс...

Синтаксис Automation:

BOOL AddBoundary(BOOL UV, BOOL Closed, long Degree, VARIANT Points, VARIANT Weights, VARIANT Knots, double TMin, double TMax);

Синтаксис COM:

HRESULT AddBoundary(BOOL UV, BOOL Closed, long Degree, VARIANT Points, VARIANT Weights, VARIANT Knots, double TMin, double TMax, BOOL * Res);

Выходные параметры:

UV	- TRUE параметры кривой заданы в UV координатах поверхности (2D-Nurbs), - FALSE параметры заданы 3D координатами (3D-Nurbs). При создании границы выполняется проецирование 3D кривой на поверхность,
Closed	- TRUE замкнутое представление - FALSE - разомкнутое представление,
Degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
Points	- массив SafeArray точек VT_ARRAY | VT_R8,
Weights	- веса,
Knots	- массив SafeArray узлов точек VT_ARRAY | VT_R8,
TMin, TMax	- минимальный и максимальный параметры кривой.

Возвращаемое значение:

TRUE	- в случае удачи.

Примечание:

Позволяет добавить границу для поверхности.

Граница не должна пересекаться с другими границами данной поверхности.

DeleteBoundary – Удалить границу

Интерфейс...

Синтаксис Automation:

BOOL DeleteBoundary(long BoundaryIndex)

Синтаксис COM:

HRESULT DeleteBoundary(long BoundaryIndex, BOOL * Res);

Выходные параметры:

BoundaryIndex	- индекс цикла (индекс контура).

Возвращаемое значение:

TRUE - в случае удачи.

GetBoundary – Получить параметры NURBS–представления границы

Интерфейс...

Синтаксис Automation:

BOOL GetBoundary(BOOL UV, BOOL Closed, long BoundaryIndex, long * Degree, VARIANT * Points, VARIANT * Weights, VARIANT * Knots, double * TMin, double * TMax);

Синтаксис COM:

HRESULT GetBoundary(BOOL UV, BOOL Closed, long BoundaryIndex, long * Degree, VARIANT * Points, VARIANT * Weights, VARIANT * Knots, double * TMin, double * TMax, BOOL * Res);

Выходные параметры:

UV	- TRUE параметры кривой заданы в UV координатах поверхности (2D-Nurbs), - FALSE параметры заданы 3D координатами (3D-Nurbs). При создании границы выполняется проецирование 3D кривой на поверхность.
Closed	- TRUE замкнутое представление, - FALSE - разомкнутое представление,
BoundaryIndex	- индекс цикла (индекс контура).

Входные параметры:

Degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
Points	- массив SafeArray точек VT_ARRAY | VT_R8,
Weights	- веса,
Knots	- массив SafeArray узлов точек VT_ARRAY | VT_R8,
TMin, TMax	- минимальный и максимальный параметры кривой.

Возвращаемое значение:

TRUE - в случае удачи.

GetNurbsParams – Получить параметры NURBS-поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetNurbsParams(BOOL ClosedV, BOOL ClosedU,
long * DegreeV, long * DegreeU, long * NPV, long * NPU,
VARIANT * Points, VARIANT * Weights, VARIANT * KnotsV, VARIANT * KnotsU);

Синтаксис COM:

HRESULT GetNurbsParams(BOOL ClosedV, BOOL ClosedU,
long * DegreeV, long * DegreeU, long * NPV, long * NPU,
VARIANT * Points, VARIANT * Weights, VARIANT * KnotsV, VARIANT * KnotsU, BOOL * Res);

Выходные параметры:

ClosedV	- TRUE - получение параметров для замкнутого по V представления, - FALSE - получение параметров для разомкнутого по V представления,
ClosedU	- TRUE- получение параметров для замкнутого по U представления, - FALSE - получение параметров для разомкнутого по U представления.

Входные параметры:

DegreeV	- порядок NURBS по V (степень полинома + 1), от 3 до 10,
DegreeU	- порядок NURBS по U (степень полинома + 1), от 3 до 10,
NPV	- количество точек по V,
NPU	- количество точек по U,
Points	- массив SafeArray точек VT_ARRAY | VT_R8,
Weights	- веса,
KnotsV	- массив SafeArray узлов точек по V VT_ARRAY | VT_R8,
KnotsU	- массив SafeArray узлов точек по U VT_ARRAY | VT_R8.

Возвращаемое значение:

TRUE	- в случае удачи.

InitParamByFace – Создать по объекту

Интерфейс...

Синтаксис Automation:

BOOL InitParamByFace(LPDISPATCH Face);

Синтаксис COM:

HRESULT InitParamByFace(IModelObject * Face, BOOL * Res);

Выходные параметры:

Face - указатель на грань IModelObject.

Возвращаемое значение:

TRUE - в случае удачи.

SetNurbsParams – Установить параметры NURBS-поверхности

Интерфейс...

Синтаксис Automation:

BOOL SetNurbsParams(BOOL ClosedV, BOOL ClosedU,
long DegreeV, long DegreeU, long NPV, long NPU,
VARIANT Points, VARIANT Weights, VARIANT KnotsV, VARIANT KnotsU);

Синтаксис COM:

HRESULT SetNurbsParams(BOOL ClosedV, BOOL ClosedU,
long DegreeV, long DegreeU, long NPV, long NPU,
VARIANT Points, VARIANT Weights, VARIANT KnotsV, VARIANT KnotsU, BOOL * Res);

Выходные параметры:

ClosedV	- TRUE- получение параметров для замкнутого по V представления, - FALSE - получение параметров для разомкнутого по V представления,
ClosedU	- TRUE- получение параметров для замкнутого по U представления, - FALSE - получение параметров для разомкнутого по U представления,
DegreeV	- порядок NURBS по V (степень полинома + 1), от 3 до 10,
DegreeU	- порядок NURBS по U (степень полинома + 1), от 3 до 10,
NPV	- количество точек по V,
NPU	- Количество точек по U,
Points	- массив SafeArray точек VT_ARRAY I VT_R8,
Weights	- веса,
KnotsV	- массив SafeArray узлов точек по V VT_ARRAY I VT_R8,
KnotsU	- массив SafeArray узлов точек по U VT_ARRAY I VT_R8.

Возвращаемое значение:

TRUE - в случае удачи.

Интерфейс IPlane3D

Интерфейс плоскости 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPlane3D

IPlane3D – свойства

Surface – Получить интерфейс математической поверхности

Интерфейс...

Тип данных: указатель на интерфейс IMathSurface3D

Синтаксис Automation:

Surface = Object.Surface	Получить свойство (*)
Surface = Object.GetSurface()	Получить свойство (**)

Синтаксис COM:

Object.get_Surface(&Surface)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

Интерфейс IPlane3DByPlaneCurve

Плоскость через плоскую кривую.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPlane3D

IPlane3DByPlaneCurve

IPlane3DByPlaneCurve – свойства

Curve – Интерфейс плоской кривой

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Curve = Object.Curve                    Получить свойство (* )
Object.Curve = Curve                    Установить свойство (* )
Curve = Object.GetCurve()               Получить свойство (**)
Object.SetCurve( Curve )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Curve( &Curve )              Получить свойство
Object.put_Curve(  Curve )              Установить свойство
```

Интерфейс IPlane3DTangentToFaceInPoint

Плоскость касательная к грани в точке 3D.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IPlane3D
IPlane3DTangentToFaceInPoint

IPlane3DTangentToFaceInPoint – свойства

Face – Грань для построения плоскости

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Face = Object.Face                      Получить свойство (* )
Object.Face = Face                      Установить свойство (* )
Face = Object.GetFace()                 Получить свойство (**)
Object.SetFace( Face )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Face( &Face )                Получить свойство
Object.put_Face(  Face )                Установить свойство
```

Point – Точка для построения плоскости, касательной к грани

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Point = Object.Point                    Получить свойство (* )
Object.Point = Point                    Установить свойство (* )
Point = Object.GetPoint()               Получить свойство (**)
Object.SetPoint( Point )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Point( &Point )              Получить свойство
Object.put_Point(  Point )              Установить свойство
```

ParamU – Параметр U

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ParamU = Object.ParamU                  Получить свойство (* )
Object.ParamU = ParamU                  Установить свойство (* )
ParamU = Object.GetParamU()             Получить свойство (**)
Object.SetParamU( ParamU )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ParamU( &ParamU )            Получить свойство
Object.put_ParamU(  ParamU )            Установить свойство
```

ParamV – Параметр V

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ParamV = Object.ParamV                  Получить свойство (* )
Object.ParamV = ParamV                  Установить свойство (* )
ParamV = Object.GetParamV()             Получить свойство (**)
Object.SetParamV( ParamV )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ParamV( &ParamV )            Получить свойство
Object.put_ParamV(  ParamV )            Установить свойство
```

Интерфейс IPlane3DByOffset

Смещенная плоскость.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPlane3D

IPlane3DByOffset

IPlane3DByOffset – свойства

BasePlane – Интерфейс базовой плоскости или плоской грани

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
BasePlane = Object.BasePlane                Получить свойство (* )
Object.BasePlane = BasePlane                Установить свойство (* )
BasePlane = Object.GetBasePlane()           Получить свойство (**)
Object.SetBasePlane( BasePlane )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BasePlane( &BasePlane )          Получить свойство
Object.put_BasePlane(  BasePlane )          Установить свойство
```

Direction – Направление смещения от базовой плоскости. TRUE – прямое направление, FALSE – обратное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction                Получить свойство (* )
Object.Direction = Direction                Установить свойство (* )
Direction = Object.GetDirection()           Получить свойство (**)
Object.SetDirection( Direction )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction )          Получить свойство
Object.put_Direction(  Direction )          Установить свойство
```

Offset – Смещение (расстояние) от базовой плоскости

Интерфейс...

Тип данных: double

Синтаксис Automation:

Offset = Object.Offset	Получить свойство (*)
Object.Offset = Offset	Установить свойство (*)
Offset = Object.GetOffset()	Получить свойство (**)
Object.SetOffset(Offset)	Установить свойство (**)

Синтаксис COM:

Object.get_Offset(&Offset)	Получить свойство
Object.put_Offset(Offset)	Установить свойство

Интерфейс IPlane3DBy3Points

Плоскость, проходящая через три вершины.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DBy3Points

IPlane3DBy3Points – свойства

Point1 – Интерфейс первой точки

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point1 = Object.Point1	Получить свойство (*)
Object.Point1 = Point1	Установить свойство (*)
Point1 = Object.GetPoint1()	Получить свойство (**)
Object.SetPoint1(Point1)	Установить свойство (**)

Синтаксис COM:

Object.get_Point1(&Point1)	Получить свойство
Object.put_Point1(Point1)	Установить свойство

Point2 – Интерфейс второй точки

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point2 = Object.Point2	Получить свойство (*)
Object.Point2 = Point2	Установить свойство (*)
Point2 = Object.GetPoint2()	Получить свойство (**)
Object.SetPoint2(Point2)	Установить свойство (**)

Синтаксис COM:

Object.get_Point2(&Point2)	Получить свойство
Object.put_Point2(Point2)	Установить свойство

Point3 – Интерфейс третьей точки

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point3 = Object.Point3	Получить свойство (*)
Object.Point3 = Point3	Установить свойство (*)
Point3 = Object.GetPoint3()	Получить свойство (**)
Object.SetPoint3(Point3)	Установить свойство (**)

Синтаксис COM:

Object.get_Point3(&Point3)	Получить свойство
Object.put_Point3(Point3)	Установить свойство

Интерфейс IPlane3DByAngle

Плоскость под углом к другой плоскости.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DByAngle

IPlane3DByAngle – свойства

Angle – Угол наклона плоскости относительно базовой плоскости

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                    Получить свойство (* )
Object.Angle = Angle                    Установить свойство (* )
Angle = Object.GetAngle()               Получить свойство (**)
Object.SetAngle( Angle )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )              Получить свойство
Object.put_Angle(  Angle )              Установить свойство
```

BaseLine – Интерфейс базовой прямой (оси или ребра)

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
 BaseLine = Object.BaseLine             Получить свойство (* )
Object.BaseLine = BaseLine              Установить свойство (* )
BaseLine = Object.GetBaseLine()         Получить свойство (**)
Object.SetBaseLine( BaseLine )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BaseLine( &BaseLine )        Получить свойство
Object.put_BaseLine(  BaseLine )        Установить свойство
```

BasePlane – Интерфейс базовой плоскости

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
BasePlane = Object.BasePlane            Получить свойство (* )
Object.BasePlane = BasePlane            Установить свойство (* )
BasePlane = Object.GetBasePlane()       Получить свойство (**)
Object.SetBasePlane( BasePlane )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_BasePlane( &BasePlane )      Получить свойство
Object.put_BasePlane(  BasePlane )      Установить свойство
```

Direction – Направление смещения от базовой плоскости. TRUE – прямое направление, FALSE – обратное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Интерфейс IPlane3DByEdgeAndPoint

Плоскость через ребро и вершину.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DByEdgeAndPoint

IPlane3DByEdgeAndPoint – свойства

Edge – Интерфейс ребра, через которое требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Edge = Object.Edge	Получить свойство (*)
Object.Edge = Edge	Установить свойство (*)
Edge = Object.GetEdge()	Получить свойство (**)
Object.SetEdge(Edge)	Установить свойство (**)

Синтаксис COM:

Object.get_Edge(&Edge)	Получить свойство
Object.put_Edge(Edge)	Установить свойство

Point – Интерфейс вершины, через которую требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point = Object.Point	Получить свойство (*)
Object.Point = Point	Установить свойство (*)
Point = Object.GetPoint()	Получить свойство (**)
Object.SetPoint(Point)	Установить свойство (**)

Синтаксис COM:

Object.get_Point(&Point)	Получить свойство
Object.put_Point(Point)	Установить свойство

Интерфейс IPlane3DBy2Edge

Плоскость через ребро параллельно /перпендикулярно другому ребру 3D.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DBy2Edge

IPlane3DBy2Edge – свойства

Edge1 – Первое ребро, через которое будет проходить новая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Edge1 = Object.Edge1	Получить свойство (*)
Object.Edge1 = Edge1	Установить свойство (*)
Edge1 = Object.GetEdge1()	Получить свойство (**)
Object.SetEdge1(Edge1)	Установить свойство (**)

Синтаксис COM:

```
Object.get_Edge1( &Edge1 )                Получить свойство
Object.put_Edge1(  Edge1 )                Установить свойство
```

Edge2 – Второе ребро, через которое будет проходить новая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Edge2 = Object.Edge2                      Получить свойство (* )
Object.Edge2 = Edge2                      Установить свойство (* )
Edge2 = Object.GetEdge2()                 Получить свойство (**)
Object.SetEdge2( Edge2 )                  Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Edge2( &Edge2 )                Получить свойство
Object.put_Edge2(  Edge2 )                Установить свойство
```

Parallel – Признак положения плоскости. TRUE – Параллельно ребру, FALSE – Перпендикулярно ребру

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Parallel = Object.Parallel                Получить свойство (* )
Object.Parallel = Parallel                Установить свойство (* )
Parallel = Object.GetParallel()           Получить свойство (**)
Object.SetParallel( Parallel )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Parallel( &Parallel )          Получить свойство
Object.put_Parallel(  Parallel )          Установить свойство
```

Интерфейс IPlane3DParallelByPoint

Плоскость через вершину параллельно другой плоскости 3D.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPlane3D

IPlane3DParallelByPoint

IPlane3DParallelByPoint – свойства

Plane – Интерфейс плоскости или плоской грани, параллельно которой требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство
Object.put_Plane(Plane)	Установить свойство

Point – Интерфейс вершины, через которую требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point = Object.Point	Получить свойство (*)
Object.Point = Point	Установить свойство (*)
Point = Object.GetPoint()	Получить свойство (**)
Object.SetPoint(Point)	Установить свойство (**)

Синтаксис COM:

Object.get_Point(&Point)	Получить свойство
Object.put_Point(Point)	Установить свойство

Интерфейс IPlane3DPerpendicularByEdge

Плоскость, проходящая через вершину перпендикулярно ребру.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IPlane3D

IPlane3DPerpendicularByEdge

IPlane3DPerpendicularByEdge – свойства

Edge – Интерфейс ребра, перпендикулярно которому будет проходить новая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Edge = Object.Edge	Получить свойство (*)
Object.Edge = Edge	Установить свойство (*)
Edge = Object.GetEdge()	Получить свойство (**)
Object.SetEdge(Edge)	Установить свойство (**)

Синтаксис COM:

Object.get_Edge(&Edge)	Получить свойство
Object.put_Edge(Edge)	Установить свойство

Point – Интерфейс вершины, через которую требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Point = Object.Point	Получить свойство (*)
Object.Point = Point	Установить свойство (*)
Point = Object.GetPoint()	Получить свойство (**)
Object.SetPoint(Point)	Установить свойство (**)

Синтаксис COM:

Object.get_Point(&Point)	Получить свойство
Object.put_Point(Point)	Установить свойство

Vector3D – Получить параметры вектора

Интерфейс...

Тип данных: Указатель на интерфейс IVector3D

Синтаксис Automation:

Vector3D = Object.Vector3D	Получить свойство (*)
Vector3D = Object.GetVector3D()	Получить свойство (**)

Синтаксис COM:

Object.get_Vector3D(&Vector3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IPlane3DNormalToSurface

Нормальная плоскость.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DNormalToSurface

IPlane3DNormalToSurface – свойства

Angle – Угол наклона плоскости относительно плоского объекта

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

AngleDirection – Направление угла.TRUE – прямое направление, FALSE – обратное направление

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AngleDirection = Object.AngleDirection	Получить свойство (*)
Object.AngleDirection = AngleDirection	Установить свойство (*)
AngleDirection = Object.GetAngleDirection()	Получить свойство (**)
Object.SetAngleDirection(AngleDirection)	Установить свойство (**)

Синтаксис COM:

Object.get_AngleDirection(&AngleDirection)	Получить свойство
Object.put_AngleDirection(AngleDirection)	Установить свойство

Face – Интерфейс цилиндрической или конической грани, к которой требуется построить нормальную плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Face = Object.Face	Получить свойство (*)
Object.Face = Face	Установить свойство (*)
Face = Object.GetFace()	Получить свойство (**)
Object.SetFace(Face)	Установить свойство (**)

Синтаксис COM:

Object.get_Face(&Face)	Получить свойство
Object.put_Face(Face)	Установить свойство

Plane – Интерфейс координатной плоскости или плоской грани, параллельно которой требуется построить плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство
Object.put_Plane(Plane)	Установить свойство

Интерфейс IPlane3DMiddle

Средняя плоскость 3D.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IPlane3D
IPlane3DMiddle

IPlane3DMiddle – свойства

Object1 – Прямолинейное ребро, плоская грань или конструктивная плоскость 1

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Object1 = Object.Object1	Получить свойство (*)
Object.Object1 = Object1	Установить свойство (*)
Object1 = Object.GetObject1()	Получить свойство (**)
Object.SetObject1(Object1)	Установить свойство (**)

Синтаксис COM:

Object.get_Object1(&Object1)	Получить свойство
Object.put_Object1(Object1)	Установить свойство

Object2 – Прямолинейное ребро, плоская грань или конструктивная плоскость 2

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Object2 = Object.Object2                Получить свойство (* )
Object.Object2 = Object2                Установить свойство (* )
Object2 = Object.GetObject2()           Получить свойство (**)
Object.SetObject2( Object2 )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Object2( &Object2 )          Получить свойство
Object.put_Object2(  Object2 )          Установить свойство
```

Orientation – Положение плоскости. TRUE – Положение 1, FALSE – Положение 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Orientation = Object.Orientation        Получить свойство (* )
Object.Orientation = Orientation        Установить свойство (* )
Orientation = Object.GetOrientation()   Получить свойство (**)
Object.SetOrientation( Orientation )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Orientation( &Orientation )  Получить свойство
Object.put_Orientation(  Orientation )  Установить свойство
```

Интерфейс IPlane3DByEdgeAndPlane

Плоскость через ребро параллельно / перпендикулярно грани 3D.

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IPlane3D
      - IPlane3DByEdgeAndPlane

IPlane3DByEdgeAndPlane – свойства

Edge – Ребро, через которое будет проходить новая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Edge = Object.Edge	Получить свойство (*)
Object.Edge = Edge	Установить свойство (*)
Edge = Object.GetEdge()	Получить свойство (**)
Object.SetEdge(Edge)	Установить свойство (**)

Синтаксис COM:

Object.get_Edge(&Edge)	Получить свойство
Object.put_Edge(Edge)	Установить свойство

Parallel – Признак положения плоскости. TRUE – Параллельно грани, FALSE – Перпендикулярно грани

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Object.Parallel = Object.Object.Parallel	Получить свойство (*)
Object.Object.Parallel = Object.Parallel	Установить свойство (*)
Object.Parallel = Object.GetObject.Parallel()	Получить свойство (**)
Object.SetObject.Parallel(Object.Parallel)	Установить свойство (**)

Синтаксис COM:

Object.get_Object.Parallel(&Object.Parallel)	Получить свойство
Object.put_Object.Parallel(Object.Parallel)	Установить свойство

Plane – Объект, параллельно или перпендикулярно которому будет проходить новая плоскость

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

```
Plane = Object.Plane                Получить свойство (* )
Object.Plane = Plane                Установить свойство (* )
Plane = Object.GetPlane()           Получить свойство (**)
Object.SetPlane( Plane )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Plane( &Plane )          Получить свойство
Object.put_Plane(  Plane )          Установить свойство
```

Интерфейс IPlane3DTangentToFace

Плоскость, касательная к грани 3D.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IPlane3D
IPlane3DTangentToFace

IPlane3DTangentToFace – свойства

Angle – Угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Angle = Object.Angle                Получить свойство (* )
Object.Angle = Angle                Установить свойство (* )
Angle = Object.GetAngle()           Получить свойство (**)
Object.SetAngle( Angle )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Angle( &Angle )          Получить свойство
Object.put_Angle(  Angle )          Установить свойство
```

Face – Цилиндрическая или коническая поверхность для построения касательной плоскости

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Face = Object.Face	Получить свойство (*)
Object.Face = Face	Установить свойство (*)
Face = Object.GetFace()	Получить свойство (**)
Object.SetFace(Face)	Установить свойство (**)

Синтаксис COM:

Object.get_Face(&Face)	Получить свойство
Object.put_Face(Face)	Установить свойство

Orientation – Положение плоскости. TRUE – Положение 1, FALSE – Положение 2

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Orientation = Object.Orientation	Получить свойство (*)
Object.Orientation = Orientation	Установить свойство (*)
Orientation = Object.GetOrientation()	Получить свойство (**)
Object.SetOrientation(Orientation)	Установить свойство (**)

Синтаксис COM:

Object.get_Orientation(&Orientation)	Получить свойство
Object.put_Orientation(Orientation)	Установить свойство

Plane – Плоскость, проходящая через ось грани

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Object.Plane = Plane	Установить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)
Object.SetPlane(Plane)	Установить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство
Object.put_Plane(Plane)	Установить свойство

Интерфейс IRuledSurface

Справка системы КОМПАС...

kompas.chm::/1115_115_8_Linejch_pov.htm

Интерфейс операции создания линейчатой поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRuledSurface

IRuledSurface – свойства

AutoSegmentation – Автоматическое разбиение линейчатой поверхности на грани

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoSegmentation = Object.AutoSegmentation	Получить свойство (*)
Object.AutoSegmentation = AutoSegmentation	Установить свойство (*)
AutoSegmentation = Object.GetAutoSegmentation()	Получить свойство (**)
Object.SetAutoSegmentation(AutoSegmentation)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoSegmentation(&AutoSegmentation)	Получить свойство
Object.put_AutoSegmentation(AutoSegmentation)	Установить свойство

Свойство позволяет включать и выключать опцию автоматического разбиения на грани.

CheckSelfIntersection – Проверка самопересечения поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CheckSelfIntersection = Object.CheckSelfIntersection	Получить свойство (*)

```
Object.CheckSelfIntersection = CheckSelfIntersection          Установить свойство (* )
CheckSelfIntersection = Object.GetCheckSelfIntersection()     Получить свойство (**)
Object.SetCheckSelfIntersection( CheckSelfIntersection )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CheckSelfIntersection( &CheckSelfIntersection )    Получить свойство
Object.put_CheckSelfIntersection(  CheckSelfIntersection )    Установить свойство
```

Свойство позволяет проверить поверхность на наличие самопересечений.

ConsiderComplianceVertices – Учет соответствия вершин направляющих при построении поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ConsiderComplianceVertices = Object.ConsiderComplianceVertices     Получить свойство (* )
Object.ConsiderComplianceVertices = ConsiderComplianceVertices     Установить свойство (* )
ConsiderComplianceVertices = Object.GetConsiderComplianceVertices()     Получить свойство (**)
Object.SetConsiderComplianceVertices( ConsiderComplianceVertices )     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ConsiderComplianceVertices( &ConsiderComplianceVertices )     Получить свойство
Object.put_ConsiderComplianceVertices( ConsiderComplianceVertices )     Установить свойство
```

Curves1 – Первая цепочка кривых (первая направляющая линейчатой поверхности) в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

```
Curves1 = Object.Curves1                  Получить свойство (* )
Object.Curves1 = Curves1                  Установить свойство (* )
Curves1 = Object.GetCurves1()             Получить свойство (**)
Object.SetCurves1( Curves1 )              Установить свойство (**)
```

Синтаксис COM:

Object.get_Curves1(&Curves1)	Получить свойство
Object.put_Curves1(Curves1)	Установить свойство

Свойство позволяет устанавливать и получать первую направляющую.

Примечание:

Направляющей может являться:

- пространственная кривая;
- цепочка последовательно соединяющихся пространственных кривых;
- точечный объект.

Curves2 – Вторая цепочка кривых (вторая направляющая линейчатой поверхности) SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Curves2 = Object.Curves2	Получить свойство (*)
Object.Curves2 = Curves2	Установить свойство (*)
Curves2 = Object.GetCurves2()	Получить свойство (**)
Object.SetCurves2(Curves2)	Установить свойство (**)

Синтаксис COM:

Object.get_Curves2(&Curves2)	Получить свойство
Object.put_Curves2(Curves2)	Установить свойство

Свойство позволяет устанавливать и получать первую направляющую.

Примечание:

Направляющей может являться:

- пространственная кривая;
- цепочка последовательно соединяющихся пространственных кривых;
- точечный объект.

EdgesCount – Количество ребер

Интерфейс...

Тип данных: long

Синтаксис Automation:

EdgesCount = Object.EdgesCount	Получить свойство (*)
EdgesCount = Object.GetEdgesCount()	Получить свойство (**)

Синтаксис COM:

Object.get_EdgesCount(&EdgesCount)	Получить свойство

Примечание

Свойство доступно только для чтения.

IRuledSurface – методы

AddNewEdge – Добавление ребра линейчатой поверхности после указанного индексом

Интерфейс...

Синтаксис Automation:

BOOL AddNewEdge(long IndexAt);

Синтаксис COM:

HRESULT AddNewEdge(long IndexAt, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

IndexAt - индекс ребра, после которого нужно вставить новое ребро.

Метод позволяет вставить новое ребро перед заданным. Новое ребро будет иметь следующий по порядку номер.

DeleteEdge – Удаление ребра линейчатой поверхности

Интерфейс...

Синтаксис Automation:

BOOL DeleteEdge(long Index)

Синтаксис COM:

HRESULT DeleteEdge(long Index, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

Index - индекс удаляемого ребра.

Метод позволяет удалить ребро, заданное по индексу.

Примечания:

1. Оставшиеся ребра нумеруются заново.
2. Удаление ребер невозможно, если в списке остались два ребра.

GetEdgePointParam – Получение параметров точки ребра линейчатой поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetEdgePointParam(long EdgeIndex,
BOOL StartPoint,
double * X,
double * Y,
double * Z,
double * T,
IModelObject ** Segment,
IModelObject ** AssociateVertex);

Синтаксис COM:

HRESULT GetEdgePointParam(long EdgeIndex,
BOOL StartPoint,
double * X,
double * Y,
double * Z,
double * T,
IModelObject ** Segment,
IModelObject ** AssociateVertex,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

EdgeIndex	- индекс ребра,
StartPoint	- начальная или конечная вершина ребра.

Выходные параметры

X, Y, Z	- координаты точки,
T	- параметр точки на кривой,
Segment	- сегмент направляющей, на котором находится точка,

Associate Vertex	- вершина направляющей, с которой связана вершина ребра.

Метод позволяет получить параметры вершины ребра линейчатой поверхности.

GetEdgePointParams – Получение параметров ребер линейчатой поверхности

Интерфейс...

Синтаксис Automation:

BOOL GetEdgePointParams(VARIANT * Points1,
VARIANT * T1,
VARIANT * Segments1,
VARIANT * AssociateVertexes1,
VARIANT * Points2,
VARIANT * T2, V
ARIANT * Segments2,
VARIANT * AssociateVertexes2);

Синтаксис COM:

HRESULT GetEdgePointParams(VARIANT * Points1,
VARIANT * T1,
VARIANT * Segments1,
VARIANT * AssociateVertexes1,
VARIANT * Points2,
VARIANT * T2,
VARIANT * Segments2,
VARIANT * AssociateVertexes2,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

Points1	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 координат вершин ребер на первой направляющей,
T1	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 параметров вершин ребер на первой направляющей,
Segments1	- массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH сегментов первой направляющей,
AssociateVertexes1	- массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH вершин первой направляющей, связанных с вершинами,
Points2	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 координат вершин ребер на второй направляющей,
T2	- массив SAFEARRAY VT_R8 - VT_ARRAY | VT_R8 параметров вершин ребер на второй направляющей,

Segments2	- массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH сегментов второй направляющей,
AssociateVertexes2	- массив SAFEARRAY DISPATCH - VT_ARRAY | VT_DISPATCH вершин второй направляющей, связанных с вершинами.

Метод позволяет получать параметры ребер линейчатой поверхности.

SetEdgePointParam – Изменение положения точки ребра линейчатой поверхности

Интерфейс...

Синтаксис Automation:

BOOL SetEdgePointParam(long EdgeIndex,
BOOL StartPoint,
double X,
double Y,
double Z,
double * T,
IModelObject * Segment,
IModelObject * AssociateVertex);

Синтаксис COM:

HRESULT SetEdgePointParam(long EdgeIndex,
BOOL StartPoint,
double X,
double Y,
double Z,
double * T,
IModelObject * Segment,
IModelObject * AssociateVertex,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры

EdgeIndex	- индекс ребра,
StartPoint	- начальная или конечная вершина ребра,
X, Y, Z	- координаты точки,
T	- параметр точки на кривой,
Segment	- сегмент направляющей, на котором находится точка,
AssociateVertex	- вершина направляющей, с которой связана вершина ребра.

Метод позволяет задать положение вершины ребра линейчатой поверхности на направляющей.

Интерфейс ISplitLine

Интерфейс линии разъема.

Иерархия:

IKompasAPIObject

IModelObject

ISplitLine

Примечание:

1. Интерфейс можно получить у коллекции линий разъема, используя свойство ISplitLines::SplitLine или метод ISplitLines::Add.
2. После задания параметров линии разъема требуется вызвать метод IModelObject::Update.

ISplitLine – свойства

CutObjects – Секущие объекты

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
CutObjects = Object.CutObjects          Получить свойство (* )
Object.CutObjects = CutObjects          Установить свойство (* )
CutObjects = Object.GetCutObjects       Получить свойство (**)
Object.SetCutObjects( CutObjects )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CutObjects( &CutObjects )    Получить свойство
Object.put_CutObjects(  CutObjects )    Установить свойство
```

Direction – Направление формирования линии разъема

Интерфейс...

Тип данных: из перечисления ksDirectionTypeEnum

Синтаксис Automation:

```
Direction = Object.Direction            Получить свойство (* )
Object.Direction = Direction            Установить свойство (* )
Direction = Object.GetDirection()       Получить свойство (**)
Object.SetDirection( Direction )        Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction )      Получить свойство
```

```
Object.put_Direction(  Direction )        Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать направление формирования линии разъема.

Faces – Массив граней в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Faces = Object.Faces          Получить свойство (* )
Object.Faces = Faces          Установить свойство (* )
Faces = Object.GetFaces()     Получить свойство (**)
Object.SetFaces( Faces )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Faces( &Faces )        Получить свойство
Object.put_Faces(  Faces )        Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать грани операции. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.

Sketch – Эскиз

Интерфейс...

Тип данных: указатель на интерфейс эскиза ISketch

Синтаксис Automation:

```
Sketch = iObject.Sketch();    Получить свойство (* )
iObject.Sketch() = Sketch;    Установить свойство (* )
Sketch =                      Получить свойство (**)
iObject.GetSketch();
iObject.SetSketch( );         Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Sketch( &Sketch )        Получить свойство
iObject->put_Sketch( Sketch )         Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать эскиз линии разъема.

Интерфейс ISurfacePatch

Интерфейс заплатки.

Иерархия:

IKompasAPIObject

IModelObject

ISurfacePatch

Примечание:

1. Интерфейс можно получить у коллекции заплаток, используя свойство ISurfacePatches::SurfacePatch или метод ISurfacePatches::Add.
2. После задания параметров заплатки требуется вызвать метод IModelObject::Update.

ISurfacePatch – свойства

Edges – Массив ребер в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Object.Edges = Edges	Установить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)
Object.SetEdges(Edges)	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство
Object.put_Edges(Edges)	Установить свойство

Примечание.

Свойство позволяет устанавливать и получать ребра операции.

Интерфейс ISurfaceSewer

Интерфейс операции сшивки поверхностей.

Иерархия:

IKompasAPIObject

IModelObject

ISurfaceSewer

Описание:

Интерфейс позволяет создавать и редактировать операцию сшивки поверхностей.

Примечание:

1. Интерфейс можно получить у коллекции заплаток, используя свойство ISurfaceSewers::SurfaceSewer или метод ISurfaceSewers::Add.
2. После задания параметров заплатки требуется вызвать метод IModelObject::Update.

ISurfaceSewer – свойства

CreateBody – Создавать тело

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
CreateBody = Object.CreateBody              Получить свойство (* )
Object.CreateBody = CreateBody              Установить свойство (* )
CreateBody = Object.GetCreateBody()         Получить свойство (**)
Object.SetCreateBody( CreateBody )          Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CreateBody( &CreateBody )    Получить свойство
Object.put_CreateBody(  CreateBody )    Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать флаг создания тела.

Precision – Точность сшивки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Precision = Object.Precision                Получить свойство (* )
Object.Precision = Precision                Установить свойство (* )
Precision = Object.GetPrecision()           Получить свойство (**)
Object.SetPrecision( Precision )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Precision( &Precision )    Получить свойство
Object.put_Precision(  Precision )    Установить свойство
```

Примечание.

Свойство позволяет устанавливать и получать точность сшивки (максимальное расстояние между сшиваемыми ребрами).

Shells – Массив оболочек в виде SAFEARRAY DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Shells = Object.Shells          Получить свойство (* )
Object.Shells = Shells          Установить свойство (* )
Shells = Object.GetShells()     Получить свойство (**)
Object.SetShells( Shells )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Shells( &Shells )    Получить свойство
Object.put_Shells(  Shells )    Установить свойство
```

Примечание.

1. Свойство позволяет устанавливать и получать оболочки для выполнения операции.
2. Свойство возвращает оболочки в виде интерфейсов IBody7.

Интерфейс ITrimmedSurface

Справка системы КОМПАС...

kompas.chm::/1131_118_11_Usech_pov.htm

Интерфейс параметров операции усечения поверхности.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

ITrimmedSurface

ITrimmedSurface – свойства

CutObject – Секущий объект для усечения поверхности

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
CutObject = Object.CutObject                    Получить свойство (* )
Object.CutObject = CutObject                    Установить свойство (**)
CutObject = Object.GetCutObject()               Получить свойство (**)
CutObject = Object.GetCutObject()               Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CutObject( &CutObject )              Получить свойство
Object.put_CutObject(  CutObject )              Установить свойство
```

Свойство позволяет устанавливать и получать секущий объект или совокупность объектов. Секущие объекты должны однозначно определять границу отсечения, т.е. линия пересечения должна полностью пересекать усекаемую поверхность. Если объект один, используется тип VARIANT-a VT_DISPATCH, если несколько - тип VARIANT-a VT_ARRAY | VT_DISPATCH.

OperationResult – Результат операции

Интерфейс...

Тип данных: из перечисления ksOperationResultEnum

Синтаксис Automation:

```
OperationResult = Object.OperationResult        Получить свойство (* )
Object.OperationResult = OperationResult        Установить свойство (**)
OperationResult                 =               Получить свойство (**)
Object.GetOperationResult()
OperationResult                 =               Установить свойство (**)
Object.GetOperationResult()
```

Синтаксис COM:

```
Object.get_OperationResult(                     Получить свойство
&OperationResult )
Object.put_OperationResult(                     Установить свойство
OperationResult )
```

Sense – Сменить направление усечения поверхности

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Sense = Object.Sense                            Получить свойство (* )
```

```
Object.Sense = Sense                    Установить свойство (**)
Sense = Object.GetSense()               Получить свойство (**)
Object.SetSense( Sense )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Sense( &Sense )              Получить свойство
Object.put_Sense(  Sense )              Установить свойство
```

Свойство позволяет менять направление усечения, т.е. выбрать для удаления ту или иную часть усекаемой поверхности.

Surface – Усекаемая поверхность

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Surface = Object.Surface                Получить свойство (* )
Object.Surface = Surface                Установить свойство (* )
Surface = Object.GetSurface()           Получить свойство (**)
Object.SetSurface( Surface )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Surface( &Surface )          Получить свойство
Object.put_Surface(  Surface )          Установить свойство
```

Свойство позволяет устанавливать и получать усекаемую поверхность. В качестве усекаемой поверхности может использоваться связная совокупность граней одного тела или одной поверхности. Если грань одна, используется тип VARIANT-а VT_DISPATCH, если несколько - тип VARIANT-а VT_ARRAY | VT_DISPATCH.

Интерфейс IRestoredSurface

Интерфейс Восстановленная поверхность.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IRestoredSurface

Примечание:

Интерфейс можно получить с помощью метода коллекции операций восстановленная поверхность IRestoredSurfaces::Add или свойства IRestoredSurfaces::RestoredSurface.

Версия: KOMPAS v19

IRestoredSurface – свойства

Face – Грань поверхности

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

Face = Object.Face	Получить свойство (*)
Object.Face = Face	Установить свойство (*)
Face = Object.GetFace()	Получить свойство (**)
Object.SetFace(Face)	Установить свойство (**)

Синтаксис COM:

Object.get_Face(&Face)	Получить свойство,
Object.put_Face(Face)	Установить свойство.

Типы загрузки

Интерфейс ILoadCombination

Справка системы КОМПАС...

KOMPAS.chm::/1072_116_2_Tipy_zagruzki_sborki.htm

Интерфейс для работы с типами загрузки.

Иерархия:

IDispatch

ILoadCombination

Примечание:

1. Интерфейс может быть получен в качестве дополнительного для интерфейса IAssemblyDocument.
2. Интерфейс предназначен для работы с типами загрузки. Позволяет получить массив типов загрузки документа, текущий индекс типа загрузки, создать, удалить, применить тип загрузки.

   У сборки 3D есть три умолчательных типа загрузки, индексы которых соответствуют перечислению ksLoadStateEnum:

Полная	0	- полная загрузка,
Пустая	1	- все компоненты не загружены,
Упрощенная	2	- все компоненты загружены частично.

   Кроме этого могут быть еще и пользовательские типы загрузки.

ILoadCombination – свойства

CompletelyLoaded – Документ загружен полностью

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE - документ загружен полностью,
FALSE - документ загружен не полностью,

Синтаксис Automation:

```
CompletelyLoaded = iObject.CompletelyLoaded          Получить свойство (* )
CompletelyLoaded = iObject.GetCompletelyLoaded()     Получить свойство (**)
```

Синтаксис COM:

```
iObject-                                             Получить свойство (*
>get_CompletelyLoaded(&CompletelyLoaded)             )
```

Примечание:

1. Свойство позволяет узнать загружен ли документ полностью.
2. Если документ загружен полностью, можно получить массово-центровочные характеристики документа. В противном случае вернутся обнуленные данные.

CurrentIndex – Текущий индекс набора

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
CurrentIndex = iObject.CurrentIndex          Получить свойство (* )
CurrentIndex = iObject.GetCurrentIndex()     Получить свойство (**)
```

Синтаксис COM:

```
iObject-                                     Получить свойство (* )
>get_CurrentIndex(&CurrentIndex)
```

Примечание:

1. Свойство позволяет получить текущий индекс набора, по которому загружается документ по умолчанию.
2. Набор становится текущим после выполнения команды **Применить**.

LoadCombinations – Массив наборов

Интерфейс...

Тип данных: VARIANT - массив наборов в виде массива SAFEARRAY BSTR - (VT_ARRAY | VT_BSTR)

Синтаксис Automation:

LoadCombinations = Получить свойство (*)
iObject.LoadCombinations
LoadCombinations = Получить свойство (**)
iObject.GetLoadCombinations()

Синтаксис COM:

iObject->get_LoadCombinations (&LoadCombinations) Получить свойство (*)

Примечание:

1. Свойство позволяет получить массив наборов документа.
2. У сборки 3D есть три умолчательных набора, индексы которых соответствуют перечислению ksLoadStateEnum.:

Полная	0	- полная загрузка,
Пустая	1	- все компоненты не загружены,
Упрощенная	2	- все компоненты загружены частично.

Кроме этого могут быть еще и пользовательские наборы.

ProtectedFlags – Получить массив признаков защищенности типов загрузок в виде SAFEARRAY BOOL – (VT_ARRAY | VT_BOOL)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ProtectedFlags = Object.ProtectedFlags Получить свойство (*)
ProtectedFlags = Object.GetProtectedFlags() Получить свойство (**)

Синтаксис COM:

Object.get_ProtectedFlags(&ProtectedFlags) Получить свойство (*)

Примечание:

Свойство доступно только для чтения.

ILoadCombination – методы

Apply – Применить набор

Интерфейс...

Синтаксис Automation:

BOOL Apply(VARIANT loadCombinationIndex)

Синтаксис COM:

HRESULT Apply([in] VARIANT loadCombinationIndex, [out, retval] BOOL * Value);

Входные параметры:

loadCombinationIndex	- тип VARIANT - индекс набора или имя набора.

Возвращаемое значение:

TRUE	- набор успешно применен,
FALSE	- не применен.

Примечание:

1. Метод позволяет применить набор с заданным именем или индексом.
2. Набор становится текущим, что позволит открывать документ с таким набором по умолчанию.

ApplyEx – Применить тип загрузки

Интерфейс...

Синтаксис Automation:

BOOL ApplyEx(VARIANT LoadCombinationIndex, BSTR Password);

Синтаксис COM:

HRESULT ApplyEx(VARIANT LoadCombinationIndex, BSTR Password, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
Password	- пароль.

Create – Создать набор

Интерфейс...

Синтаксис Automation:

long Create(LPCTSTR combinationName)

Синтаксис COM:

HRESULT Create([in] BSTR combinationName, [out, retval] long * result);

Входные параметры:

combinationName	- имя набора.

Возвращаемое значение:

индекс набора в массиве наборов	- в случае успеха,
-1	- в случае неудачи.

Примечание:

1. Метод позволяет создать пользовательский набор с заданным именем.
2. В набор сохраняется текущее состояние дерева построения. То есть предварительно нужно отдельно пройтись по вставкам компонентов и выполнить команды **Выгрузить** или **Загрузить компонент**.
3. Набор не становится автоматически текущим.

Delete – Удалить набор

Интерфейс...

Синтаксис Automation:

BOOL Delete(VARIANT loadCombinationIndex)

Синтаксис COM:

HRESULT Delete([in] VARIANT loadCombinationIndex, [out, retval] BOOL * result);

Входные параметры:

loadCombinationIndex	- тип VARIANT - индекс набора или имя набора.

Возвращаемое значение:

TRUE	- удален,
FALSE	- не удален.

Примечание:

Метод позволяет удалить пользовательский набор с заданным именем или индексом.

DeleteEx – Удалить тип загрузки

Интерфейс...

Синтаксис Automation:

BOOL DeleteEx(VARIANT LoadCombinationIndex,
BSTR Password,
BOOL DeleteDependant);

Синтаксис COM:

HRESULT DeleteEx(VARIANT LoadCombinationIndex,
BSTR Password,
BOOL DeleteDependant,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
Password	- пароль,
DeleteDependant	- TRUE - с подчиненными типами загрузки, - FALSE - без подчиненных.

GetLoadCombinationComment – Получить комментарий

Интерфейс...

Синтаксис Automation:

BSTR GetLoadCombinationComment(VARIANT LoadCombinationIndex);

Синтаксис COM:

HRESULT GetLoadCombinationComment(VARIANT LoadCombinationIndex, BSTR * Result);

Возвращаемое значение:

строка с комментарием

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки.

GetLoadCombinationName – Получить имя

Интерфейс...

Синтаксис Automation:

BSTR GetLoadCombinationName(VARIANT LoadCombinationIndex);

Синтаксис COM:

HRESULT GetLoadCombinationName(VARIANT LoadCombinationIndex, BSTR * Result);

Возвращаемое значение:

строка с именем

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки.

SetLoadCombinationComment – Установить комментарий

Интерфейс...

Синтаксис Automation:

BOOL SetLoadCombinationComment(VARIANT LoadCombinationIndex,
BSTR NewVal,
BSTR Password);

Синтаксис COM:

HRESULT SetLoadCombinationComment(VARIANT LoadCombinationIndex,
BSTR NewVal,
BSTR Password,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
NewVal	- строка с комментарием,
Password	- пароль.

SetLoadCombinationName – Установить имя

Интерфейс...

Синтаксис Automation:

BOOL SetLoadCombinationName(VARIANT LoadCombinationIndex,
BSTR NewVal,
BSTR Password);

Синтаксис COM:

HRESULT SetLoadCombinationName(VARIANT LoadCombinationIndex,
BSTR NewVal,
BSTR Password,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
NewVal	- строка с именем,
Password	- пароль.

SetPassword – Установить пароль

Интерфейс...

Синтаксис Automation:

BOOL SetPassword(VARIANT LoadCombinationIndex,
BSTR OldPassword,
BSTR Password,
BOOL UnprotectUsers);

Синтаксис COM:

HRESULT SetPassword(VARIANT LoadCombinationIndex,
BSTR OldPassword,
BSTR Password,
BOOL UnprotectUsers,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
OldPassword	- старый пароль,
Password	- новый пароль,
UnprotectUsers	- TRUE - при удалении пароля системного типа удалять пароли пользовательских типов.

UpdateByModel – Обновить по текущему состоянию модели

Интерфейс...

Синтаксис Automation:

BOOL UpdateByModel(VARIANT LoadCombinationIndex, BSTR Password);

Синтаксис COM:

HRESULT UpdateByModel(VARIANT LoadCombinationIndex, BSTR Password, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

LoadCombinationIndex	- индекс типа загрузки,
Password	- пароль.

Интерфейс IOpenDocumentParam

Справка системы КОМПАС...

kompas.chm::/1074_116_2_1_Vybort_tipa_zagruzki_sborki.htm

Интерфейс параметров открытия документа.

Иерархия:

IDispatch

IKompasAPIObject

IOpenDocumentParam

Позволяет настраивать параметры открытия документа.

Примечание:

Интерфейс можно получить с помощью метода IDocuments::GetOpenDocumentParam или IPart7::GetOpenDocumentParam.

IOpenDocumentParam – свойства

ApplyingIndex – Индекс примененного типа загрузки

Интерфейс...

Тип данных: long

Синтаксис Automation:

ApplyingIndex = Object.ApplyingIndex	Получить свойство (*)
Object.ApplyingIndex = ApplyingIndex	Установить свойство (*)
ApplyingIndex = Object.GetApplyingIndex()	Получить свойство (**)
Object.SetApplyingIndex(ApplyingIndex)	Установить свойство (**)

Синтаксис COM:

Object.get_ApplyingIndex(&ApplyingIndex)	Получить свойство
Object.get_ApplyingIndex(&ApplyingIndex)	Установить свойство

Password – Пароль типа загрузки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Object.Password = Password	Получить свойство (*)
Object.SetPassword(Password)	Получить свойство (**)

Синтаксис COM:

Object.put_Password(Password)	Получить свойство

ReadOnly – Открытие документа с присвоением признака «Для чтения»

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ReadOnly = Object.ReadOnly	Получить свойство (*)
Object.ReadOnly = ReadOnly	Установить свойство (*)
ReadOnly = Object.GetReadOnly()	Получить свойство (**)
Object.SetReadOnly(ReadOnly)	Установить свойство (**)

Синтаксис COM:

Object.get_ReadOnly(&ReadOnly)	Получить свойство
Object.put_ReadOnly(ReadOnly)	Установить свойство

Visible – Видимость документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = Object.Visible	Получить свойство (*)
Object.Visible = Visible	Установить свойство (*)
Visible = Object.GetVisible()	Получить свойство (**)
Object.SetVisible(Visible)	Установить свойство (**)

Синтаксис COM:

Object.get_Visible(&Visible)	Получить свойство
Object.put_Visible(Visible)	Установить свойство

Интерфейс ILoadCombinationsParam

Справка системы КОМПАС...

kompas.chm::/1075_116_2_2_Vybort_tipa_zagruzki_comp.htm

Интерфейс параметров типа загрузки документа.

Иерархия:

IDispatch

IKompasAPIObject

ILoadCombinationsParam

Позволяет настраивать параметры типов загрузки документа.

Примечание:

Интерфейс можно получить с помощью метода IDocuments::GetLoadCombinationsParam.

ILoadCombinationsParam – свойства

ApplyingIndex – Получить индекс примененного типа загрузки

Интерфейс...

Тип данных: long

Синтаксис Automation:

ApplyingIndex = Object.ApplyingIndex	Получить свойство (*)
ApplyingIndex = Object.GetApplyingIndex()	Получить свойство (**)

Синтаксис COM:

Object.get_ApplyingIndex(&ApplyingIndex)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LoadCombinations – Получить массив типов загрузок в виде SAFEARRAY BSTR(VT_ARRAY | VT_BSTR)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

LoadCombinations = Object.LoadCombinations	Получить свойство (*)
LoadCombinations = Object.GetLoadCombinations()	Получить свойство (**)

Синтаксис COM:

Object.get_LoadCombinations(&LoadCombinations)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ProtectedFlags – Получить массив признаков защищенности типов загрузок в виде SAFEARRAY BOOL(VT_ARRAY | VT_BOOL)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ProtectedFlags = Object.ProtectedFlags	Получить свойство (*)
ProtectedFlags = Object.GetProtectedFlags()	Получить свойство (**)

Синтаксис COM:

Object.get_ProtectedFlags(&ProtectedFlags)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Вектор

Интерфейс IVector3D

Справка системы КОМПАС...

`kompas.chm::/1035_Glava111_Vektor.htm`

Интерфейс параметров вектора 3D.

Иерархия:

IDispatch

IKompasAPIObject

IVector3D

Интерфейс позволяет получать параметры вектора 3D.

Интерфейс можно получить:

- у интерфейса массива по сетке с помощью свойств ILinearPattern::Vector1 и ILinearPattern::Vector2;
- у интерфейса локальной системы координат с помощью свойства ILocalCoordinateSystem::Vector3D;
- у интерфейса параметров точки, заданной по смещению от опорного объекта с помощью свойства IPoint3DParamDisplace::Vector3D;
- у интерфейса присоединительной точки с помощью свойства IConjunctivePoint::Vector3D;
- у интерфейса эквидистанты кривой с помощью свойства IEquidistant3D::Vector3D;
- у интерфейса операции продления поверхности с помощью свойства IExtensionSurface::BuildingVectorParameters;
- у интерфейса линии очерка с помощью свойства ICurveOutLine::Vector3D.

IVector3D – свойства

ParametersType – Тип параметров вектора 3D

Интерфейс...

Тип данных: из перечисления ksVector3DParametersTypeEnum

Синтаксис Automation:

```
ParametersType = Object.ParametersType                  Получить свойство (* )
Object.ParametersType = ParametersType                  Установить свойство (* )
ParametersType = Object.GetParametersType()             Получить свойство (**)
Object.SetParametersType( ParametersType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ParametersType( &ParametersType )            Получить свойство
Object.put_ParametersType(  ParametersType )            Установить свойство
```

Свойство позволяет устанавливать и получать тип параметров вектора 3D.

Parameters – Получить интерфейс параметров построения вектора

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

```
Parameters = Object.Parameters          Получить свойство (* )
Parameters = Object.GetParameters()     Получить свойство (**)
```

Синтаксис COM:

Object.get_Parameters(&Parameters)	Получить свойство

Свойство позволяет получать интерфейс параметров вектора, соответствующий заданному типу параметров.

Примечание:

Свойство доступно только для чтения.

Интерфейс IVector3DAlongSurfaceNormalParameters

Справка системы КОМПАС...

kompas.chm::/Vector_postroenie.htm#perpendikulairno

Интерфейс параметров вектора, перпендикулярного грани в указанной точке.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DAlongSurfaceNormalParameters

Интерфейс позволяет устанавливать и получать параметры вектора, перпендикулярного грани в указанной точке.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DAlongSurfaceNormalParameters – свойства

BaseObject – Объект для построения вектора

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Свойство позволяет устанавливать и получать грань для построения вектора.

Direction – Направление вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Свойство позволяет устанавливать и получать направление вектора.

U – Параметр U

Интерфейс...

Тип данных: double

Синтаксис Automation:

U = Object.U	Получить свойство (*)
Object.U = U	Установить свойство (*)
U = Object.GetU()	Получить свойство (**)
Object.SetU(U)	Установить свойство (**)

Синтаксис COM:

Object.get_U(&U)	Получить свойство
Object.put_U(U)	Установить свойство

Свойство позволяет устанавливать и получать параметр U для задания точки на грани.

V – Параметр V

Интерфейс...

Тип данных: double

Синтаксис Automation:

V = Object.V	Получить свойство (*)
Object.V = V	Установить свойство (*)
V = Object.GetV()	Получить свойство (**)
Object.SetV(V)	Установить свойство (**)

Синтаксис COM:

Object.get_V(&V)	Получить свойство
Object.put_V(V)	Установить свойство

Свойство позволяет устанавливать и получать параметр V для задания точки на грани.

Интерфейс IVector3DBy2AnglesParameters

Справка системы КОМПАС...

kompas.chm::/Vector_postroenie.htm#v_spherichеskoi

Интерфейс параметров вектора по двум углам.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DBy2AnglesParameters

Интерфейс позволяет устанавливать и получать параметры вектора по двум углам.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DBy2AnglesParameters – свойства

AngleA – Азимутальный угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

AngleA = Object.AngleA	Получить свойство (*)
Object.AngleA = AngleA	Установить свойство (*)
AngleA = Object.GetAngleA()	Получить свойство (**)
Object.SetAngleA(AngleA)	Установить свойство (**)

Синтаксис COM:

Object.get_AngleA(&AngleA)	Получить свойство
Object.put_AngleA(AngleA)	Установить свойство

Свойство позволяет устанавливать и получать азимутальный угол.

AngleB – Зенитный угол

Интерфейс...

Тип данных: double

Синтаксис Automation:

AngleB = Object.AngleB	Получить свойство (*)
Object.AngleB = AngleB	Установить свойство (*)
AngleB = Object.GetAngleB()	Получить свойство (**)
Object.SetAngleB(AngleB)	Установить свойство (**)

Синтаксис COM:

Object.get_AngleB(&AngleB)	Получить свойство
Object.put_AngleB(AngleB)	Установить свойство

Свойство позволяет устанавливать и получать зенитный угол.

LocalCS – Локальная система координат

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

LocalCS = Object.LocalCS	Получить свойство (*)
Object.LocalCS = LocalCS	Установить свойство (*)
LocalCS = Object.GetLocalCS()	Получить свойство (**)
Object.SetLocalCS(LocalCS)	Установить свойство (**)

Синтаксис COM:

Object.get_LocalCS(&LocalCS)	Получить свойство
Object.put_LocalCS(LocalCS)	Установить свойство

Свойство позволяет устанавливать и получать локальную систему координат.

Примечание:

Чтобы установить глобальную СК, нужно передать NULL.

Интерфейс IVector3DBy2VertexesParameters

Справка системы КОМПАС...

kompas.chm::/Vector_postroenie.htm#dve_vershiny

Интерфейс параметров вектора 3D по двум вершинам.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DBy2VertexesParameters

Интерфейс позволяет устанавливать и получать параметры вектора по двум вершинам.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DBy2VertexesParameters – свойства

Direction – Направление вектора 3D

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Свойство позволяет устанавливать и получать направление вектора.

Vertex1 – Первая вершина вектора 3D

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Vertex1 = Object.Vertex1	Получить свойство (*)
Object.Vertex1 = Vertex1	Установить свойство (*)
Vertex1 = Object.GetVertex1()	Получить свойство (**)

Object.SetVertex1(Vertex1)	Установить свойство (**)

Синтаксис COM:

Object.get_Vertex1(&Vertex1)	Получить свойство
Object.put_Vertex1(Vertex1)	Установить свойство

Свойство позволяет устанавливать и получать первую вершину.

Vertex2 – Вторая вершина вектора 3D

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Vertex2 = Object.Vertex2	Получить свойство (*)
Object.Vertex2 = Vertex2	Установить свойство (*)
Vertex2 = Object.GetVertex2()	Получить свойство (**)
Object.SetVertex2(Vertex2)	Установить свойство (**)

Синтаксис COM:

Object.get_Vertex2(&Vertex2)	Получить свойство
Object.put_Vertex2(Vertex2)	Установить свойство

Свойство позволяет устанавливать и получать вторую вершину.

Интерфейс IVector3DByCoefficientsParameters

Справка системы КОМПАС...

`kompas.chm::/Vector_postroenie.htm#po_osi`

Интерфейс параметров вектора по коэффициентам.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DByCoefficientsParameters

Интерфейс позволяет устанавливать и получать параметры вектора по коэффициентам.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DByCoefficientsParameters – свойства

CoefficientByX – Коэффициент разложения по оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

CoefficientByX = Object.CoefficientByX	Получить свойство (*)
Object.CoefficientByX = CoefficientByX	Установить свойство (*)
CoefficientByX = Object.GetCoefficientByX()	Получить свойство (**)
Object.SetCoefficientByX(CoefficientByX)	Установить свойство (**)

Синтаксис COM:

Object.get_CoefficientByX(&CoefficientByX)	Получить свойство
Object.put_CoefficientByX(CoefficientByX)	Установить свойство

Свойство позволяет устанавливать и получать коэффициент разложения по оси X.

CoefficientByY – Коэффициент разложения по оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

CoefficientByY = Object.CoefficientByY	Получить свойство (*)
Object.CoefficientByY = CoefficientByY	Установить свойство (*)
CoefficientByY = Object.GetCoefficientByY()	Получить свойство (**)
Object.SetCoefficientByY(CoefficientByY)	Установить свойство (**)

Синтаксис COM:

Object.get_CoefficientByY(&CoefficientByY)	Получить свойство
Object.put_CoefficientByY(CoefficientByY)	Установить свойство

Свойство позволяет устанавливать и получать коэффициент разложения по оси X.

CoefficientByZ – Коэффициент разложения по оси Z

Интерфейс...

Тип данных: double

Синтаксис Automation:

CoefficientByZ = Object.CoefficientByZ	Получить свойство (*)
Object.CoefficientByZ = CoefficientByZ	Установить свойство (*)
CoefficientByZ = Object.GetCoefficientByZ()	Получить свойство (**)
Object.SetCoefficientByZ(CoefficientByZ)	Установить свойство (**)

Синтаксис COM:

Object.get_CoefficientByZ(&CoefficientByZ)	Получить свойство
Object.put_CoefficientByZ(CoefficientByZ)	Установить свойство

Свойство позволяет устанавливать и получать коэффициент разложения по оси Z.

LocalCS – Локальная система координат

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

LocalCS = Object.LocalCS	Получить свойство (*)
Object.LocalCS = LocalCS	Установить свойство (*)
LocalCS = Object.GetLocalCS()	Получить свойство (**)
Object.SetLocalCS(LocalCS)	Установить свойство (**)

Синтаксис COM:

Object.get_LocalCS(&LocalCS)	Получить свойство
Object.put_LocalCS(LocalCS)	Установить свойство

Свойство позволяет устанавливать и получать локальную систему координат.

Примечание:

Чтобы установить глобальную СК, нужно передать NULL.

Интерфейс IVector3DByCurveParameters

Справка системы КОМПАС...

`kompas.chm::/Vector_postroenie.htm#bazisnui_vektor`

Интерфейс параметров вектора по базисному вектору в точке кривой.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DByCurveParameters

Интерфейс позволяет устанавливать и получать параметры вектора по базисному вектору в точке кривой (кроме прямолинейных объектов).

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DByCurveParameters – свойства

Curve – Кривая

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

Curve = Object.Curve	Получить свойство (*)
Object.Curve = Curve	Установить свойство (*)
Curve = Object.GetCurve()	Получить свойство (**)
Object.SetCurve(Curve)	Установить свойство (**)

Синтаксис COM:

Object.get_Curve(&Curve)	Получить свойство
Object.put_Curve(Curve)	Установить свойство

Свойство позволяет устанавливать и получать кривую для построения вектора.

Direction – Направление вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- прямое направление
FALSE	- обратное направление

Свойство позволяет устанавливать и получать направление вектора.

Offset – Смещение

Интерфейс...

Тип данных: double

Синтаксис Automation:

Offset = Object.Offset	Получить свойство (*)
Object.Offset = Offset	Установить свойство (*)
Offset = Object.GetOffset()	Получить свойство (**)
Object.SetOffset(Offset)	Установить свойство (**)

Синтаксис COM:

Object.get_Offset(&Offset)	Получить свойство
Object.put_Offset(Offset)	Установить свойство

Свойство позволяет устанавливать и получать смещение вектора.

VectorType – Тип базисного вектора

Интерфейс...

Тип данных: из перечисления ksBasisVectorTypeEnum

Синтаксис Automation:

VectorType = Object.VectorType	Получить свойство (*)
Object.VectorType = VectorType	Установить свойство (*)
VectorType = Object.GetVectorType()	Получить свойство (**)
Object.SetVectorType(VectorType)	Установить свойство (**)

Синтаксис COM:

Object.get_VectorType(&VectorType)	Получить свойство
Object.put_VectorType(VectorType)	Установить свойство

Свойство позволяет устанавливать и получать тип базисного вектора.

Интерфейс IVector3DByLocalCSParameters

Справка системы КОМПАС:
Построение вектора по углу в плоскости СК

kompas.chm::/Vector_postroenie.htm#po_uglu_v_ploscosti

Интерфейс параметров вектора по углу в плоскости СК и по оси СК.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DByLocalCSParameters

Интерфейс позволяет устанавливать и получать параметры вектора по углу в плоскости СК (тип параметров ksVector3DAngle) и по оси СК (тип параметров ksVector3DAxis).

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DByLocalCSParameters – свойства

Angle – Угол наклона вектора

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle		Получить свойство (*)
Object.Angle = Angle		Установить свойство (*)
Angle	=	Получить свойство (**)
Object.GetAngle()		
Object.SetAngle(Angle)		Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

Свойство позволяет устанавливать и получать угол наклона вектора.

Примечание:

Свойство доступно для типа параметров вектора ksVector3DAngle.

AxisType – Тип оси

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

```
AxisType = Object.AxisType                  Получить свойство (* )
Object.AxisType = AxisType                  Установить свойство (* )
AxisType = Object.GetAxisType()             Получить свойство (**)
Object.SetAxisType( AxisType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_AxisType( &AxisType )            Получить свойство
Object.put_AxisType(  AxisType )            Установить свойство
```

Свойство позволяет устанавливать и получать тип оси (X, Y, Z).

Примечание:

Тип оси также определяет плоскость размещения вектора.

Direction – Направление вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Direction = Object.Direction                Получить свойство (* )
Object.Direction = Direction                Установить свойство (* )
Direction = Object.GetDirection()           Получить свойство (**)
Object.SetDirection( Direction )            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Direction( &Direction )          Получить свойство
Object.put_Direction(  Direction )          Установить свойство
```

Значения свойства:

```
TRUE                    - прямое направление,
FALSE                   - обратное направление.
```

Свойство позволяет устанавливать и получать направление вектора.

Примечание:

Свойство доступно для типа параметров вектора ksVector3DAxis.

LocalCS – Локальная система координат

Интерфейс...

Тип данных: указатель на интерфейс ILocalCoordinateSystem

Синтаксис Automation:

LocalCS = Object.LocalCS	Получить свойство (*)
Object.LocalCS = LocalCS	Установить свойство (*)
LocalCS = Object.GetLocalCS()	Получить свойство (**)
Object.SetLocalCS(LocalCS)	Установить свойство (**)

Синтаксис COM:

Object.get_LocalCS(&LocalCS)	Получить свойство
Object.put_LocalCS(LocalCS)	Установить свойство

Свойство позволяет устанавливать и получать локальную систему координат.

Примечание:

Чтобы установить глобальную СК, нужно передать NULL.

Интерфейс IVector3DByObjectParameters

Справка системы КОМПАС: Построение вектора по прямолинейному ребру, оси или перпендикулярно плоскости кривой

`kompas.chm::/Vector_postroenie.htm#po_rebru`

Интерфейс параметров вектора по ребру или плоскости.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DByObjectParameters

Интерфейс позволяет устанавливать и получать параметры вектора по ребру или плоскости.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DByObjectParameters – свойства

BaseObject – Объект для построения вектора

Интерфейс...

Тип данных: указатель на интерфейс IModelObject

Синтаксис Automation:

BaseObject = Object.BaseObject	Получить свойство (*)
Object.BaseObject = BaseObject	Установить свойство (*)
BaseObject = Object.GetBaseObject()	Получить свойство (**)
Object.SetBaseObject(BaseObject)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObject(&BaseObject)	Получить свойство
Object.put_BaseObject(BaseObject)	Установить свойство

Свойство позволяет устанавливать и получать объект, задающий направление вектора (ребро или плоскость).

Direction – Направление вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Свойство позволяет устанавливать и получать направление вектора.

Интерфейс IVector3DByScreenNormalParameters

Справка системы КОМПАС...

`kompas.chm::/Vector_postroenie.htm#ploskosti_perpend`

Интерфейс параметров вектора, перпендикулярного плоскости экрана.

Иерархия:

IDispatch

IKompasAPIObject

IVector3DByScreenNormalParameters

Интерфейс позволяет устанавливать и получать параметры вектора, перпендикулярного плоскости экрана.

Интерфейс можно получить у интерфейса вектора с помощью свойства IVector3D::Parameters.

IVector3DByScreenNormalParameters – свойства

Direction – Направление вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Direction = Object.Direction	Получить свойство (*)
Object.Direction = Direction	Установить свойство (*)
Direction = Object.GetDirection()	Получить свойство (**)
Object.SetDirection(Direction)	Установить свойство (**)

Синтаксис COM:

Object.get_Direction(&Direction)	Получить свойство
Object.put_Direction(Direction)	Установить свойство

Значения свойства:

TRUE	Прямое направление,
FALSE	Обратное направление.

Свойство позволяет устанавливать и получать направление вектора.

Fix – Фиксация вектора

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Fix = Object.Fix	Получить свойство (*)
Object.Fix = Fix	Установить свойство (*)
Fix = Object.GetFix()	Получить свойство (**)
Object.SetFix(Fix)	Установить свойство (**)

Синтаксис COM:

Object.get_Fix(&Fix)	Получить свойство
Object.put_Fix(Fix)	Установить свойство

Свойство позволяет устанавливать и получать признак фиксации вектора.

Твердое тело

Интерфейс IBody7

Интерфейс твердого тела.

Иерархия:

IDispatch

IKompasAPIObject

IBody7

IMassInertiaParam7

IPropertyKeeper

Описание:

Позволяет получить интерфейс твердого тела. Тело является результатом выполнения операций (например, операции выдавливания). Представляет собой область, ограниченную замкнутой поверхностью.

Примечание:

1. Интерфейс можно получить у интерфейса дерева с помощью метода IFeature7::ResultBodies.
2. Интерфейс можно получить у интерфейса области применения для тел документа в операции с помощью метода IChooseBodies7::Bodies.
3. Имеет дополнительные интерфейсы IMassInertiaParam7, IColorParam7, IFeature7, IPropertyKeeper, которые можно получить с помощью метода IUnknown::QueryInterface.

IBody7 – свойства

BeginBodyId – Идентификатор начального тела

Интерфейс...

Тип данных: long

Синтаксис Automation:

BeginBodyId = Object. BeginBodyId	Получить свойство (*)
BeginBodyId = Object.GetBeginBodyId()	Получить свойство (**)

Синтаксис COM:

Object.get_BeginBodyId(&BeginBodyId)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BodyId – Идентификатор тела

Интерфейс...

Тип данных: long

Синтаксис Automation:

BodyId = Object. BodyId	Получить свойство (*)
BodyId = Object.GetBodyId()	Получить свойство (**)

Синтаксис COM:

Object.get_BodyId(&BodyId)	Получить свойство

Примечание:

Свойство доступно только для чтения.

CreateSpcObjects – Создавать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CreateSpcObjects = Object.CreateSpcObjects	=	Получить свойство (*)
Object.CreateSpcObjects = CreateSpcObjects	=	Установить свойство (*)
CreateSpcObjects = Object.GetCreateSpcObjects()	=	Получить свойство (**)
Object.SetCreateSpcObjects(CreateSpcObjects)		Установить свойство (**)

Синтаксис COM:

Object.get_CreateSpcObjects(&CreateSpcObjects)	Получить свойство
Object.put_CreateSpcObjects(CreateSpcObjects)	Установить свойство

Editable – Признак редактирования

Интерфейс...

Тип данных: из перечисления ksEditableStateEnum

Синтаксис Automation:

Editable = Object.Editable	Получить свойство (*)
Object.Editable = Editable	Установить свойство (*)
Editable = Object.GetEditable()	Получить свойство (**)
Object.SetEditable(Editable)	Установить свойство (**)

Синтаксис COM:

Object.get_Editable(&Editable)	Получить свойство
Object.put_Editable(Editable)	Установить свойство

FinalBodyId – Идентификатор конечного тела

Интерфейс...

Тип данных: long

Синтаксис Automation:

FinalBodyId = Object. FinalBodyId	Получить свойство (*)
FinalBodyId = Object.GetFinalBodyId()	Получить свойство (**)

Синтаксис COM:

Object.get_FinalBodyId(&FinalBodyId)	Получить свойство

Примечание:

Свойство доступно только для чтения.

HatchParam – Параметры штриховки

Интерфейс...

Тип данных: Указатель на интерфейс IHatchParam

Синтаксис Automation:

HatchParam = Object.HatchParam		Получить свойство (*)
HatchParam Object.GetHatchParam()	=	Получить свойство (**)

Синтаксис COM:

Object.get_HatchParam(&HatchParam)	Получить свойство

Свойство позволяет получать интерфейс параметров штриховки тела.

Примечание:

Свойство доступно только для чтения.

Hidden – Состояние видимости объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Hidden = Object.Hidden          Получить свойство (* )
Object.Hidden = Hidden          Установить свойство (* )
Hidden = Object.GetHidden()     Получить свойство (**)
Object.SetHidden( Hidden )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Hidden( &Hidden )    Получить свойство
Object.put_Hidden(  Hidden )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать обозначение компонента.

HiddenEx – Состояние видимости объекта

Интерфейс...

Тип данных: из перечисления ksVisibleStateEnum

Синтаксис Automation:

```
HiddenEx = Object.HiddenEx          Получить свойство (* )
Object.HiddenEx = HiddenEx          Установить свойство (* )
HiddenEx                       =    Получить свойство (**)
Object.GetHiddenEx()
Object.SetHiddenEx( HiddenEx )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_HiddenEx( &HiddenEx )    Получить свойство
Object.put_HiddenEx(  HiddenEx )    Установить свойство
```

LayerNumber – Номер слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

LayerNumber = Object.LayerNumber	=	Получить свойство (*)
Object.LayerNumber = LayerNumber	=	Установить свойство (*)
LayerNumber = Object.GetLayerNumber()	=	Получить свойство (**)
Object.SetLayerNumber(LayerNumber)		Установить свойство (**)

Синтаксис COM:

Object.get_LayerNumber(&LayerNumber)	Получить свойство
Object.put_LayerNumber(LayerNumber)	Установить свойство

Marking – Обозначение компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

Marking = iObject.Marking	Получить свойство (*)
iObject.Marking = Marking	Установить свойство (*)
Marking = iObject.GetMarking()	Получить свойство (**)
iObject.SetMarking (Marking)	Установить свойство (**)

Синтаксис COM:

iObject->get_Marking (&Marking)	Получить свойство
iObject->put_Marking (Marking)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать обозначение компонента.

Name – Имя тела

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

iObject.SetName (Name) Установить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name) Получить свойство
iObject->put_Name (Name) Установить свойство

Примечание:

Свойство позволяет устанавливать и получать имя трехмерной модели.

OwnerBodyId – Идентификатор тела, которое поглотило это тело

Интерфейс...

Тип данных: long

Синтаксис Automation:

OwnerBodyId = Object. OwnerBodyId Получить свойство (*)
OwnerBodyId = Object.GetOwnerBodyId() Получить свойство (**)

Синтаксис COM:

Object.get_OwnerBodyId(&OwnerBodyId) Получить свойство

Примечание:

Свойство доступно только для чтения.

Projected – Признак проецирования

Интерфейс...

Тип данных: из перечисления ksProjectionOptionEnum

Синтаксис Automation:

Projected = Object.Projected Получить свойство (*)
Object.Projected = Projected Установить свойство (*)
Projected = Получить свойство (**)
Object.GetProjected()
Object.SetProjected(Projected) Установить свойство (**)

Синтаксис COM:

Object.get_Projected(&Projected) Получить свойство
Object.put_Projected(Projected) Установить свойство

Threads – Коллекция условных обозначений резьбы

Интерфейс...

Тип данных: Указатель на интерфейс IThreads

Синтаксис Automation:

Threads = Object.Threads	Получить свойство(*)
Threads = Object.GetThreads()	Получить свойство (**)

Синтаксис COM:

Object.get_Threads(&Threads)	Получить свойство

UserParameters – Получить интерфейс параметров

Интерфейс...

Тип данных: Указатель на интерфейс IUnknown

Синтаксис Automation:

UserParameters = Object.UserParameters(Val)	Получить свойство(*)
UserParameters = Object.GetUserParameters(Val)	Получить свойство (**)

Синтаксис COM:

Object.get_UserParameters(Val, &UserParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IBody7 – методы

GetGabarit – Получить габарит

Интерфейс...

Синтаксис Automation:

```
BOOL GetGabarit( double * X1,
double * Y1,
double * Z1,
double * X2,
double * Y2,
double * Z2 );
```

Синтаксис COM:

HRESULT GetGabarit(double * X1,
double * Y1,
double * Z1,
double * X2,
double * Y2,
double * Z2
BOOL * Result);

Входные параметры:

X1, Y1, Z1, X2, Y2, Z2 - координаты габаритного прямоугольника.

Возвращаемое значение:

TRUE - успешное завершение,
FALSE - в случае неудачи.

Примечание:

Метод позволяет установить габаритный прямоугольник Панели свойств. Точки задаются в относительных координатах окна КОМПАС.

Update – Изменить свойства объекта

Интерфейс...

Синтаксис Automation:

BOOL Update( );

Синтаксис COM:

HRESULT Update (VARIANT_BOOL* PVal);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

После вызова метода вступают в силу измененные свойства тела: имя, обозначение, свойства массово-центровочных характеристик.

Свойства цвета объекта

Интерфейс IColorParam7

Свойства цвета объекта.

Иерархия:

IDispatch

IColorParam7

Примечание:

1. Интерфейс является дополнительным для IModelObject, IBody7 и IEmbodiment.
2. Оптические свойства используются для верхнего компонента, вставки, тела, операций, поверхностей, грани. Для остальных объектов используется только цвет и свойство UseColor.
3. Оптические свойства задаются в относительных единицах 0...1.

IColorParam7 – свойства

Ambient – Общий свет

Интерфейс...

Тип данных: double

Значения свойства:

- Общий свет (от 0 до 1).

Синтаксис Automation:

Ambient = Object.Ambient	Получить свойство (*)
Object.Ambient = Ambient	Установить свойство (*)
Ambient = Object.GetAmbient()	Получить свойство (**)
Object.SetAmbient (Ambient)	Установить свойство (**)

Синтаксис COM:

Object.get_Ambient(&Ambient)	Получить свойство
Object.put_Ambient(Ambient)	Установить свойство

Color – Цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

Color = Object.Color	Получить свойство (*)
Object.Color = Color	Установить свойство (*)
Color = Object.GetColor()	Получить свойство (**)
Object.SetColor(Color)	Установить свойство (**)

Синтаксис COM:

Object.get_Color(&Color)	Получить свойство

Object.put_Color(Color)	Установить свойство

Diffuse – Диффузия

Интерфейс...

Тип данных: double

Значения свойства:

- Диффузия (от 0 до 1).

Синтаксис Automation:

Diffuse = Object.diffuse	Получить свойство (*)
Object.diffuse = diffuse	Установить свойство (*)
Diffuse = Object.GetDiffuse()	Получить свойство (**)
Object.SetDiffuse (diffuse)	Установить свойство (**)

Синтаксис COM:

Object.get_Diffuse(&Diffuse)	Получить свойство
Object.put_Diffuse(Diffuse)	Установить свойство

Emission – Излучение

Интерфейс...

Тип данных: double

Значения свойства:

- Излучение (от 0 до 1).

Синтаксис Automation:

Emission = Object.Emission	Получить свойство (*)
Object.Emission = Emission	Установить свойство (*)
Emission = Object.GetEmission()	Получить свойство (**)
Object.SetEmission(Emission)	Установить свойство (**)

Синтаксис COM:

Object.get_Emission(&Emission)	Получить свойство
Object.put_Emission(Emission)	Установить свойство

Shininess – Блеск

Интерфейс...

Тип данных: double

Значения свойства:

- Блеск (от 0 до 1).

Синтаксис Automation:

Shininess = Object.Shininess	Получить свойство (*)
Object.Shininess = Shininess	Установить свойство (*)
Shininess = Object.GetShininess()	Получить свойство (**)
Object.SetShininess (Shininess)	Установить свойство (**)

Синтаксис COM:

Object.get_Shininess(&Shininess)	Получить свойство
Object.put_Shininess(Shininess)	Установить свойство

Specularity – Зеркальность

Интерфейс...

Тип данных: double

Значения свойства:

- Зеркальность (от 0 до 1).

Синтаксис Automation:

specularity = Object.specularity	Получить свойство (*)
Object.specularity = specularity	Установить свойство (*)
specularity = Object.GetSpecularity()	Получить свойство (**)
Object.SetSpecularity (specularity)	Установить свойство (**)

Синтаксис COM:

Object.get_Specularity(&Specularity)	Получить свойство
Object.put_Specularity(Specularity)	Установить свойство

Transparency – Прозрачность

Интерфейс...

Тип данных: double

Значения свойства:

- Прозрачность (от 0 до 1).

Синтаксис Automation:

Transparency = Object.Transparency	Получить свойство (*)
Object.Transparency = Transparency	Установить свойство (*)
Transparency = Object.GetTransparency()	Получить свойство (**)
Object.SetTransparency (Transparency)	Установить свойство (**)

Синтаксис COM:

Object.get_Transparency(&Transparency)	Получить свойство
Object.get_Transparency(&Transparency)	Установить свойство

Примечание:

Этот параметр характеризует способность поверхности пропускать падающий на нее свет. Если значение этого параметра равно 1 (или 100%), то поверхность совершенно непрозрачная. При моделировании деталей из стекла, органического стекла, слюды, бесцветного полиэтилена и подобных материалов значение прозрачности выбирают из диапазона между 0 и 1.

Если значение этого параметра равно 0,5 (или 50%), то поверхность полупрозрачная.

Если значение этого параметра равно 0, то поверхность полностью прозрачная. Такая поверхность невидима.

UseColor – Используемый цвет

Интерфейс...

Тип данных: из перечисления UseColor

Синтаксис Automation:

UseColor = Object.UseColor	Получить свойство (*)
Object.UseColor = UseColor	Установить свойство (*)
UseColor = Object.GetUseColor()	Получить свойство (**)
Object.SetUseColor(UseColor)	Установить свойство (**)

Синтаксис COM:

Object.get_UseColor(&UseColor)	Получить свойство
Object.put_UseColor(UseColor)	Установить свойство

Использование свойства для объектов:

Верхний компонент		
useColorOur	0	собственный цвет
Вставка в сборку		
useColorOur	0	собственный цвет
useColorOwner	1	цвет сборки
useColorSource	2	цвет источника
Грань, операции, объекты детали		

useColorOur	0	собственный цвет
useColorSource	2	цвет источника

Операции: вырезать выдавливанием, скругление, фаска, отверстие, оболочка, сечение плоскостью, сечение эскизом, удалить грань

useColorOur	0	собственный цвет
useColorOwner	1	цвет сборки
useColorSource	2	цвет источника

Тело

useColorOur	0	собственный цвет
useColorSource	2	цвет источника

Примечание:

В случае ошибки возвращается useColorUnknown.

IColorParam7 – методы

GetAdvancedColor – Получить параметры цвета объекта

Интерфейс...

Синтаксис Automation:

BOOL GetAdvancedColor(long* Color,
double* Ambient,
double* Diffuse,
double* Specularity,
double* Shininess,
double* Transparency,
double* Emission);

Синтаксис COM:

HRESULT GetAdvancedColor(long* Color,
double* Ambient,
double* Diffuse,
double* Specularity,
double* Shininess,
double* Transparency,
double* Emission,
BOOL * Result);

Выходные параметры:

color	цвет,
ambient	Общий свет,
diffuse	Диффузия,
specularity	Зеркальность,
shininess	Блеск,
transparency	Прозрачность,
emission	Излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Оптические свойства задаются в относительных единицах от 0..1.

SetAdvancedColor – Установить параметры цвета объекта

Интерфейс...

Синтаксис Automation:

BOOL SetAdvancedColor(long* Color,
double* Ambient,
double* Diffuse,
double* Specularity,
double* Shininess,
double* Transparency,
double* Emission);

Синтаксис COM:

HRESULT SetAdvancedColor(long* Color,
double* Ambient,
double* Diffuse,
double* Specularity,
double* Shininess,
double* Transparency,
double* Emission,
BOOL * Result);

Выходные параметры:

color	цвет,
ambient	Общий свет,
diffuse	Диффузия,
specularity	Зеркальность,
shininess	Блеск,
transparency	Прозрачность,
emission	Излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Оптические свойства задаются в относительных единицах от 0..1.

Интерфейс ITexturesParam

Интерфейс параметров наложения текстур.

Иерархия:

IDispatch

ITexturesParam

Интерфейс является дополнительным для интерфейса IColorParam7 объектов IModelObject. IPart7, IBody7,ILayer3D.

Для установки параметров отображения текстур для объектов модели требуется, чтобы у объекта был установлен признак использования собственного цвета,
т.е. свойство IColorParam7::UseColor требуется установить равным useColorOur.

Версия: KOMPAS v19

ITexturesParam – свойства

TextureFileName – Имя файла текстуры

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

TextureFileName Object.TextureFileName(Type)	=	Получить свойство (*)
Object.TextureFileName(Type) TextureFileName	=	Установить свойство (*)
TextureFileName Object.GetTextureFileName(Type)	=	Получить свойство (**)
Object.SetTextureFileName(Type, TextureFileName)		Установить свойство (**)

Синтаксис COM:

Object.get_TextureFileName(Type, &TextureFileName)	Получить свойство
Object.put_TextureFileName(Type, TextureFileName)	Установить свойство

Входные параметры:

ksTextureTypeEnum	- Type - тип текстуры.

TextureHeight – Высота текстуры

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextureHeight = Object.TextureHeight (Type)	=	Получить свойство (*)
Object.TextureHeight (Type) = TextureHeight	=	Установить свойство (*)
TextureHeight = Object.GetTextureHeight (Type)	=	Получить свойство (**)
Object.SetTextureHeight (Type, TextureHeight)		Установить свойство (**)

Синтаксис COM:

Object.get_TextureHeight (Type, &TextureHeight)	Получить свойство
Object.put_TextureHeight (Type, TextureHeight)	Установить свойство

Входные параметры:

ksTextureTypeEnum	- Type - тип текстуры.

TextureWidth – Ширина текстуры

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextureWidth = Object.TextureWidth (Type)	=	Получить свойство (*)
Object.TextureWidth (Type) = TextureWidth	=	Установить свойство (*)
TextureWidth = Object.GetTextureWidth (Type)	=	Получить свойство (**)
Object.SetTextureWidth (Type, TextureWidth)		Установить свойство (**)

Синтаксис COM:

Object.get_TextureWidth (Type, &TextureWidth)	Получить свойство

Object.put_TextureWidth (Type, TextureWidth) — Установить свойство

Входные параметры:

ksTextureTypeEnum - Type - тип текстуры.

TextureDx – Смещение по горизонтали

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextureDx = Object.TextureDx (Type) — Получить свойство (*)
Object.TextureDx (Type) = TextureDx — Установить свойство (*)
TextureDx = Object.GetTextureDx (Type) — Получить свойство (**)
Object.SetTextureDx (Type, TextureDx) — Установить свойство (**)

Синтаксис COM:

Object.get_TextureDx (Type, &TextureDx) — Получить свойство
Object.put_TextureDx (Type, TextureDx) — Установить свойство

Входные параметры:

ksTextureTypeEnum - Type - тип текстуры.

TextureDy – Смещение по вертикали

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextureDy = Object.TextureDy (Type) — Получить свойство (*)

Object.TextureDy (Type) = Установить свойство (*)
TextureDy
TextureDy = Получить свойство (**)
Object.GetTextureDy (Type)
Object.SetTextureDy (Type, Установить свойство (**)o
TextureDy)

Синтаксис COM:

Object.get_TextureDy (Type, Получить свойство
&TextureDy)
Object.put_TextureDy (Type, Установить свойство
TextureDy)

Входные параметры:

ksTextureTypeEnum - Type - тип текстуры.

TextureAngle – Поворот текстуры

Интерфейс...

Тип данных: double

Синтаксис Automation:

TextureAngle = Получить свойство (*)
Object.TextureAngle (Type)
Object.TextureAngle (Type) = Установить свойство (*)
TextureAngle
TextureAngle = Получить свойство (**)
Object.GetTextureAngle (Type)
Object.SetTextureAngle (Type, Установить свойство (**)
TextureAngle)

Синтаксис COM:

Object.get_TextureAngle (Type, Получить свойство
&TextureAngle)
Object.put_TextureAngle (Type, Установить свойство
TextureAngle)

Входные параметры:

ksTextureTypeEnum - Type - тип текстуры.

ITexturesParam – методы

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM :

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Текстовый документ

Интерфейс ITextDocumentSectionsManager

Интерфейс менеджера разделов текстового документа.

Иерархия:

IDispatch

ITextDocumentSectionsManager

Примечание:

Интерфейс является дополнительным к интерфейсу документа ITextDocument. Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

KOMPAS v19

ITextDocumentSectionsManager – свойства

Section – Получить параметры раздела по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ITextDocumentSection.

Синтаксис Automation:

Section = Object.Section	=	Получить свойство(*)
Section= Object.GetSection()		Получить свойство (**)

Синтаксис COM:

Object.get_Section(&Section)	Получить свойство

Входные параметры:

long Index	- индекс раздела.

Примечание:

Свойство доступно только для чтения.

SectionByTextLine – Получить параметры раздела по индексу строки

Интерфейс...

Тип данных: Указатель на интерфейс ITextDocumentSection

Синтаксис Automation:

SectionByTextLine Object.SectionByTextLine	=	Получить свойство(*)
SectionByTextLine= Object.GetSectionByTextLine()		Получить свойство (**)

Синтаксис COM:

Object.get_SectionByTextLine(&SectionByTextLine)	Получить свойство

Входные параметры:

long Index	- индекс раздела.

Примечание:

Свойство доступно только для чтения.

SectionsCount – Количество разделов

Интерфейс...

Тип данных: long

Синтаксис Automation:

SectionsCount Object.SectionsCount	=	Получить свойство(*)
SectionsCount Object.GetSectionsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SectionsCount(&SectionsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ITextDocumentSectionsManager – методы

AddSection – Добавить раздел

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddSection();

Синтаксис COM:

HRESULT AddSection(ITextDocumentSection * * Result);

Возвращаемое значение:

- Указатель на интерфейс ITextDocumentSection

Примечания:

1. Метод позволяет создать новый интерфейс параметров раздела текстового документа.
2. Метод добавляет новый раздел в конец документа.

AddSectionAt – Добавить раздел

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddSectionAt(long Index);

Синтаксис COM :

HRESULT AddSectionAt(long Index, ITextDocumentSection * * Result);

Возвращаемое значение:

- Указатель на интерфейс ITextDocumentSection

Входные параметры:

Index - индекс раздела.

Примечания:

1. Метод позволяет создать новый интерфейс параметров раздела текстового документа.
2. Метод добавляет новый раздел в определенное место текстового документа.

GetSectionLineIndexes – Получить индексы первой и последней строки раздела и число строк раздела

Интерфейс...

Синтаксис Automation:

long GetSectionLineIndexes(long SectionIndex, long * FirstLineIndex, long * LastLineIndex);

Синтаксис COM :

HRESULT GetSectionLineIndexes(long SectionIndex, long * FirstLineIndex, long * LastLineIndex, long * Result);

Возвращаемое значение:

- Число строк в разделе.

Входные параметры:

SectionIndex - Идентификатор раздела.

Выходные параметры:

FirstLineIndex - индекс строки, в которой начинается раздел,
LastLineIndex - индекс последней строки раздела.

MoveLinesToSection – Перенести строки в другой раздел

Интерфейс...

Синтаксис Automation:

BOOL MoveLinesToSection(ITextDocumentSection * Section, long FirstLineIndex, long LastLineIndex);

Синтаксис COM:

HRESULT MoveLinesToSection(ITextDocumentSection * Section, long FirstLineIndex, long LastLineIndex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,

Входные параметры:

Section - индекс раздела,
FirstLineIndex - индекс строки, в которой начинается раздел,
LastLineIndex - индекс последней строки раздела.

MoveSection – Переместить раздел

Интерфейс...

Синтаксис Automation:

BOOL MoveSection(long SectionIndex, long NewSectionIndex);

Синтаксис COM:

HRESULT MoveSection(long SectionIndex, long NewSectionIndex, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

Входные параметры:

SectionIndex	- индекс перемещаемого раздела,
NewSectionIndex	- индекс раздела, перед которым будет размещаться перемещаемый раздел.

Примечания:

Метод перемещает выбранный раздел в определенное место текстового документа.

Интерфейс ITextDocumentSection

Интерфейс параметров раздела текстового документа.

Иерархия:

IDispatch

IKompasAPIObject

ITextDocumentSection

Примечание:

Интерфейс можно получить с помощью методов менеджера разделов ITextDocumentSectionsManager.

ITextDocumentSection – свойства

Format – Формат листа

Интерфейс...

Тип данных: Указатель на интерфейс ISheetFormat

Синтаксис Automation:

Format = Object.Format	=	Получить свойство(*)
Format = Object.GetFormat()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Format(&Format)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LayoutLibraryFileName – Имя файла библиотеки стилей оформления для первого листа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LayoutLibraryFileName = Object.LayoutLibraryFileName	=	Получить свойство (*)
Object.LayoutLibraryFileName = LayoutLibraryFileName	=	Установить свойство (*)
LayoutLibraryFileName = Object.GetLayoutLibraryFileName()	=	Получить свойство (**)
Object.SetLayoutLibraryFileName(LayoutLibraryFileName)		Установить свойство (**)

Синтаксис COM:

Object.get_LayoutLibraryFileName(&LayoutLibraryFileName)	Получить свойство,
Object.put_LayoutLibraryFileName(LayoutLibraryFileName)	Установить свойство.

LayoutStyleNumber – Номер стиля оформления для первого листа

Интерфейс...

Тип данных: double

Синтаксис Automation:

LayoutStyleNumber = Object.LayoutStyleNumber	=	Получить свойство (*)
Object.LayoutStyleNumber = LayoutStyleNumber	=	Установить свойство (*)
LayoutStyleNumber = Object.GetLayoutStyleNumber()	=	Получить свойство (**)
Object.SetLayoutStyleNumber(LayoutStyleNumber)		Установить свойство (**)

Синтаксис COM:

Object.get_LayoutStyleNumber(&LayoutStyleNumber)	Получить свойство,
Object.put_LayoutStyleNumber(LayoutStyleNumber)	Установить свойство.

EvenLayoutLibraryFileName – Имя файла библиотеки стилей оформления для четных листов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

EvenLayoutLibraryFileName Object.EvenLayoutLibraryFileName	=	Получить свойство (*)
Object.EvenLayoutLibraryFileName EvenLayoutLibraryFileName	=	Установить свойство (*)
EvenLayoutLibraryFileName Object.GetEvenLayoutLibraryFileName( )	=	Получить свойство (**)
Object.SetEvenLayoutLibraryFileName(EvenLayoutLibraryFileName)		Установить свойство (**)

Синтаксис COM:

Object.get_EvenLayoutLibraryFileName (&EvenLayoutLibraryFileName)	Получить свойство,
Object.put_EvenLayoutLibraryFileName (EvenLayoutLibraryFileName)	Установить свойство.

EvenLayoutStyleNumber – Номер стиля оформления для четных листов

Интерфейс...

Тип данных: double

Синтаксис Automation:

EvenLayoutStyleNumber Object.EvenLayoutStyleNumber	=	Получить свойство (*)
Object.EvenLayoutStyleNumber EvenLayoutStyleNumber	=	Установить свойство (*)
EvenLayoutStyleNumber Object.GetEvenLayoutStyleNumber()	=	Получить свойство (**)

Object.SetEvenLayoutStyleNumber(EvenLayoutStyleNumber) — Установить свойство (**)

Синтаксис COM:

Object.get_EvenLayoutStyleNumber(&EvenLayoutStyleNumber) — Получить свойство,
Object.put_EvenLayoutStyleNumber(EvenLayoutStyleNumber) — Установить свойство.

OddLayoutLibraryFileName – Имя файла библиотеки стилей оформления для нечетных листов

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

OddLayoutLibraryFileName = Object.OddLayoutLibraryFileName — Получить свойство (*)
Object.OddLayoutLibraryFileName = OddLayoutLibraryFileName — Установить свойство (*)
OddLayoutLibraryFileName = Object.GetOddLayoutLibraryFileName() — Получить свойство (**)
Object.SetOddLayoutLibraryFileName(OddLayoutLibraryFileName) — Установить свойство (**)

Синтаксис COM:

Object.get_OddLayoutLibraryFileName(&OddLayoutLibraryFileName) — Получить свойство,
Object.put_OddLayoutLibraryFileName(OddLayoutLibraryFileName) — Установить свойство.

OddLayoutStyleNumber – Номер стиля оформления для первого листа

Интерфейс...

Тип данных: double

Синтаксис Automation:

OddLayoutStyleNumber	=	Получить свойство (*)
Object.OddLayoutStyleNumber		
Object.OddLayoutStyleNumber	=	Установить свойство (*)
OddLayoutStyleNumber		
OddLayoutStyleNumber	=	Получить свойство (**)
Object.GetOddLayoutStyleNumber()		
Object.SetOddLayoutStyleNumber(		Установить свойство (**)
OddLayoutStyleNumber)		

Синтаксис COM:

Object.get_OddLayoutStyleNumber(	Получить свойство,
&OddLayoutStyleNumber)	
Object.put_OddLayoutStyleNumber(	Установить свойство.
OddLayoutStyleNumber)	

ITextDocumentSection – методы

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Delete – Удалить раздел

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Менеджер библиотек

Интерфейс ILibraryManager

Интерфейс Менеджера библиотек.

Иерархия:

IKompasAPIObject

ILibraryManager

Примечание:

1. Данный интерфейс объединяет работу с библиотеками процедур (файлы *.rtw и *.dll) и библиотеками документов: фрагментов (файлы *.lfr) и моделей (файлы *.l3d). Интерфейс позволяет выполнять следующие действия:

- получить коллекции соответствующих библиотек, активный узел в менеджере библиотек, его название и комментарий;
- получить и управлять положением и видимостью Менеджера библиотек.

2. Данный интерфейс можно получить у интерфейса приложения IApplication с помощью свойства IApplication::LibraryManager.

ILibraryManager – свойства

ActiveFolder – Активная папка в окне Менеджера библиотек

Функция не поддерживается

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

ActiveFolder iObject.ActiveFolder	=	Получить свойство (*)
iObject.ActiveFolder ActiveFolder	=	Установить свойство (*)
ActiveFolder iObject.GetActiveFolder()	=	Получить свойство (**)
iObject.SetActiveFolder (ActiveFolder)		Установить свойство (**)

Синтаксис COM:

iObject->get_ActiveFolder (&ActiveFolder)	Получить свойство
iObject->put_ActiveFolder (ActiveFolder)	Установить свойство

Примечание:

Позволяет получить и установить имя текущей папки в Менеджере библиотек.

ActiveFolderComment – Комментарий активной папки

Функция не поддерживается

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ActiveFolderComment = iObject.ActiveFolderComment	=	Получить свойство (*)
iObject.ActiveFolderComment = ActiveFolderComment	=	Установить свойство (*)
ActiveFolderComment = iObject.GetActiveFolderComment ()	=	Получить свойство (**)
iObject.SetActiveFolderComment (ActiveFolderComment)		Установить свойство (**)

Синтаксис COM:

iObject->get_ActiveFolderComment (&ActiveFolderComment)	Получить свойство
iObject->put_ActiveFolderComment (ActiveFolderComment)	Установить свойство

Примечание:

Комментарий текущей папки может быть многострочным. Поэтому свойство возвращает или принимает переменную типа VARIANT на безопасный массив строк SAFEARRAY (VT_ARRAY | VT_BSTR).

CurrentLibrary – Текущая прикладная библиотека

Интерфейс...

Тип данных: указатель на интерфейс IProceduresLibrary

Синтаксис Automation:

CurrentLibrary = iObject.CurrentLibrary	=	Получить свойство (*)
CurrentLibrary = iObject.GetCurrentLibrary()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_CurrentLibrary (&CurrentLibrary) — Получить свойство

Примечание:

1. Подключенными одновременно могут быть несколько библиотек, но текущей библиотекой может быть одна.
2. Текущей является библиотека, у которой непосредственно выполняется команда или вызван обработчик события.

DocumentsLibraries – Коллекция библиотек документов

Интерфейс...

Тип данных: Указатель на интерфейс IInsertsLibraries

Синтаксис Automation:

DocumentsLibraries = Object.DocumentsLibraries — Получить свойство (*)

DocumentsLibraries = Object.GetDocumentsLibraries() — Получить свойство (**)

Синтаксис COM:

Object.get_DocumentsLibraries(&DocumentsLibraries) — Получить свойство

Примечание:

Свойство доступно только для чтения

FragmentsLibraries – Коллекция библиотек фрагментов

Интерфейс...

Тип данных: Указатель на коллекцию библиотек документов IInsertsLibraries

Синтаксис Automation:

FragmentsLibraries = iObject.FragmentsLibraries — Получить свойство (*)

LayerGroups = iObject.GetLayerGroups() — Получить свойство (**)

Синтаксис COM:

iObject->get_FragmentsLibraries (&FragmentsLibraries) — Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию библиотек фрагментов, которые зарегистрированы в Менеджере библиотек.
2. Под библиотекой фрагментов понимается библиотека документов КОМПАС, состоящая из фрагментов.
3. Следует различать следующие состояния библиотек фрагментов:

- зарегистрирована в Менеджере библиотек; имеется описание библиотеки в Менеджере,
- подключена; эта библиотека отображается в меню "Библиотеки", рядом с ее названием в списке показан знак «галочка»,

4. Если библиотека защищена лицензией, то при подключении библиотеки она захватывает лицензию.
5. Фрагменты из подключенной библиотеки могут быть вставлены в графические документ системы КОМПАС.
6. Если библиотека не защищена, ее можно редактировать.

Layout – Положение Менеджера библиотек (вверху, внизу, слева, справа, плавающий)

Интерфейс...

Тип данных: PropertyManagerLayout

Синтаксис Automation:

```
Layout = iObject.Layout          Получить свойство (* )
Layout = iObject.GetLayout()     Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Layout (&Layout)          Получить свойство
```

Примечание:

Позволяет получить положение Менеджера библиотек.

ModelsLibraries – Коллекция библиотек моделей

Интерфейс...

Тип данных: Указатель на коллекцию библиотек документов IInsertsLibraries

Синтаксис Automation:

```
ModelsLibraries                  =  Получить свойство (* )
iObject.ModelsLibraries
ModelsLibraries                  =  Получить свойство (**)
iObject.ModelsLibraries()
```

Синтаксис COM:

iObject->get_ModelsLibraries (&ModelsLibraries)		Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию библиотек моделей, которые зарегистрированы в Менеджере библиотек.
2. Под библиотекой фрагментов понимается библиотека документов КОМПАС, состоящая из деталей и сборок.
3. Следует различать следующие состояния библиотек фрагментов:

▼ зарегистрирована в Менеджере библиотек; имеется описание библиотеки в Менеджере,

▼ подключена; эта библиотека отображается в меню "Библиотеки", рядом с ее названием в списке показан знак «галочка»,

4. Если библиотека защищена лицензией, то при подключении библиотеки она захватывает лицензию.
5. Модели из подключенной библиотеки могут быть вставлены в документ-модель системы КОМПАС.
6. Если библиотека не защищена, ее можно редактировать.

ProceduresLibraries – Коллекция прикладных библиотек

Интерфейс...

Тип данных: IProceduresLibraries

Синтаксис Automation:

ProceduresLibraries = iObject.ProceduresLibraries	=	Получить свойство (*)
ProceduresLibraries = iObject.GetProceduresLibraries()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ProceduresLibraries (&ProceduresLibraries)		Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию процедурных библиотек, которые зарегистрированы в Менеджере библиотек.
2. Под процедурной библиотекой понимается библиотека, разработанная в какой-либо среде программирования с использованием Компас API.
3. Следует различать следующие состояния процедурных библиотек:

▼ зарегистрирована в Менеджере библиотек; имеется описание библиотеки в Менеджере,

- подключена; библиотека подключена к задаче, эта библиотека отображается в меню "Библиотеки", рядом с ее названием в списке показан знак «галочка»,
- библиотека выполняется (является текущей); вызвана команда (процедура) библиотеки или библиотека подписалась на события системы КОМПАС. В случае выполнения библиотечной команды библиотеку нельзя отключить и удалить описание библиотеки из Менеджера библиотек.

4. Если библиотека защищена лицензией, то при подключении библиотеки она захватывает лицензию.

SystemControlStartLibrary – Получить прикладную библиотеку, которая запустила SystemControlStart

Интерфейс...

Тип данных: IProceduresLibrary

Синтаксис Automation:

SystemControlStartLibrary = iObject.SystemControlStartLibrary	Получить свойство (*)
SystemControlStartLibrary = iObject.GetSystemControlStartLibrary()	Получить свойство (**)

Синтаксис COM:

iObject->get_SystemControlStartLibrary (&CurrentLibrary)	Получить свойство

Примечание:

1. Свойство позволяет вернуть интерфейс библиотеки.
2. Библиотека во время выполнения может вызвать функцию SystemControlStart для передачи управления системе КОМПАС. После вызова этой функции выполнение библиотеки приостанавливается.
3. Управление возвращается библиотеке при вызове функции SystemControlStop из этой же библиотеки или при выполнении соответствующей команды меню.

SystemControlStartResult – Получить результат работы SystemControlStart

Интерфейс...

Тип данных: ksSystemControlStartEnum.

Синтаксис Automation:

SystemControlStartResult = iObject.SystemControlStartResult	Получить свойство (*)
SystemControlStartResult = iObject.GetSystemControlStartResult()	Получить свойство (**)

Синтаксис COM:

iObject->get_SystemControlStartResult (&SystemControlStartResult)	Получить свойство

Visible – Свойство видимости Менеджера библиотек

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство (**)
iObject.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible (&Visible)	Получить свойство
iObject->put_Visible (Visible)	Установить свойство

Примечание:

Позволяет получить и установить свойство видимости Менеджера библиотек.

ILibraryManager– методы

AddFolder – Добавить новую папку в Менеджер библиотек

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

void AddFolder (BSTR PathFolder);

Синтаксис COM:

HRESULT AddFolder ([in] BSTR PathFolder);

Входные параметры:

PathFolder	- путь в Менеджере библиотек к папке. Если папка является вложенной, т. е. добавляется в другую папку, имена папок внутри пути отделяются символом "|".

RemoveFolder – Удалить папку из Менеджера библиотек

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

void RemoveFolder (BSTR PathFolder);

Синтаксис COM:

HRESULT RemoveFolder ([in] BSTR PathFolder);

Входные параметры:

PathFolder	- путь в Менеджере библиотек к папке. Если папка является вложенной, т. е. добавляется в другую папку, имена папок внутри пути отделяются символом "|".

SetCurrentLibrary – Установить текущую прикладную библиотеку

Интерфейс...

Синтаксис Automation:

BOOL SetCurrentLibrary (LPDISPATCH pVal);

Синтаксис COM:

HRESULT SetCurrentLibrary ([in] IProceduresLibrary* pVal, [out, retval] VARIANT_BOOL* result);

Входные параметры:

pVal	- интерфейс библиотеки IProceduresLibrary, которую нужно сделать текущей.

Возвращаемое значение:

TRUE	- библиотека стала текущей,
FALSE	- в случае ошибки.

Примечание:

Данный метод следует использовать осторожно, иначе можно нарушить последовательность выполнения библиотек.

Метод может быть применен к подключенной библиотеке.

Метод позволяет остановить режим SystemControlStart, запущенный не из выполняемой (текущей) библиотеки. Для этого необходимо выполнить следующие действия:

1. При помощи свойства SystemControlStartLibrary получить библиотеку, вызвавшую SystemControlStart.
2. Сделать ее текущей методом SetCurrentLibrary.
3. При помощи метода ksSystemControlStop вернуть управление библиотеке, которая запустила SystemControlStart.
4. Восстановить в качестве текущей выполняемую ранее библиотеку.

Интерфейс ILibrary

Базовый интерфейс всех библиотек системы КОМПАС.

Иерархия:

IKompasAPIObject

ILibrary

Описание:

Базовый интерфейс для прикладных библиотек, библиотек фрагментов и библиотек моделей. Включает в себя функционал, общий для этих библиотек. С помощью этого интерфейса можно узнать тип библиотеки, имя библиотеки, имя файла, признак лицензионной защиты, управлять подключением библиотеки.

ILibrary – свойства

Attach – Подключение библиотеки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Attach = iObject.Attach          Получить свойство (* )
iObject.Attach = Attach          Установить свойство (* )
Attach = iObject.GetAttach()     Получить свойство (**)
iObject.SetAttach (Attach)       Установить свойство (**)
```

Синтаксис COM:

```
iObject->get_Attach (&Attach)    Получить свойство
iObject->put_Attach (Attach)     Установить свойство
```

Значение свойства:

```
TRUE     - библиотека подключена,
FALSE    - библиотека не подключена.
```

Примечание:

Свойство позволяет получить и управлять подключением библиотеки. При подключении библиотеки в системе КОМПАС создается объект, отвечающий за работу данной библиотеки. В Менеджере библиотек напротив имени библиотеки появляется значок «галочка». “Захватывается лицензия на библиотеку. Если это прикладная библиотека, то подключенную библиотеку можно посылать на выполнение, вызвав метод Execute. Для библиотек документов можно производить вставку документа в документ КОМПАС.

Enable – Признак лицензионной защиты библиотеки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Enable = iObject.Enable	Получить свойство (*)
Enable = iObject.GetEnable()	Получить свойство (**)

Синтаксис COM:

iObject->get_Enable(&Enable)	Получить свойство

Значения свойства:

TRUE	- библиотека доступна для выполнения или вставки,
FALSE	- библиотека недоступна для выполнения или вставки.

Примечание:

Свойство возвращает признак лицензионной защиты библиотеки. Библиотека может быть не подключена.

LibraryManagerFolder – Папка библиотеки в Менеджере библиотек

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

LibraryManagerFolder = iObject.LibraryManagerFolder	=	Получить свойство (*)
iObject->get_LibraryManagerFolder(&LibraryManagerFolder)		Получить свойство (**)

Синтаксис COM:

iObject->get_LibraryManagerFolder(&LibraryManagerFolder)	Получить свойство

Примечание:

1. Свойство доступно только для чтения.
2. Если библиотека зарегистрирована в Менеджере библиотек, то свойство позволяет получить в виде строки папку или папки, где библиотека располагается. Если папка является вложенной в другую папку, имена папок отделяются символом "|".

LibraryType – Тип библиотеки

Интерфейс...

Тип данных: ksLibraryTypeEnum

Синтаксис Automation:

LibraryType = iObject.LibraryType	=	Получить свойство (*)
LibraryType = iObject.GetLibraryType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LibraryType(&LibraryType)	Получить свойство

Примечание:

Свойство возвращает тип библиотеки (прикладная, фрагментов, моделей).

Name – Имя библиотеки

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

iObject->get_Name(&Name)	Получить свойство

Примечание:

Свойство возвращает отображаемое имя библиотеки, задаваемое при помощи функции DisplayLibraryName.

PathName – Имя файла библиотеки

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

PathName = iObject.PathName		Получить свойство (*)
PathName	=	Получить свойство (**)
iObject.GetPathName()		

Синтаксис COM:

iObject->get_PathName(	Получить свойство
&PathName)	

Примечание:

Свойство возвращает имя файла библиотеки.

ILibrary – методы

Execute – Выполнить команду

Интерфейс...

Синтаксис Automation:

BOOL Execute (long command, LPDISPATCH external, VARIANT_BOOL post);

Синтаксис COM:

HRESULT Execute([in] long command,
[in] LPDISPATCH external,
[in] VARIANT_BOOL post,
[out, retval] VARIANT_BOOL* pVal);

Входные параметры:

command	- номер команды,
external	- интерфейс для обмена данными между библиотекой, вызвавшей Execute, и библиотекой, в которой реализована команда. Интерфейс определен разработчиком библиотек. Может быть NULL,
post	- отложенное выполнение, если TRUE, если FALSE - немедленное.

Возвращаемое значение:

TRUE	- команда выполнена.
FALSE	- команда не выполнена.

Примечание:

Метод позволяет выполнить команду прикладной библиотеки. При этом проверяется лицензионная защита, и может быть передан пользовательский интерфейс для обмена данными между библиотеками. Выполнение команды может быть отложенное или немедленное.

Получить интерфейс для обмена данными между библиотекой в функции LibraryEntry можно с помощью свойства IProceduresLibrary::ExternalInterface

Интерфейс IInsertsLibrary

Интерфейс библиотеки элементов (документов) Компас.

Иерархия:

IKompasAPIObject

ILibrary

IInsertsLibrary

Описание:

Позволяет получить доступ к библиотеке элементов (документов) системы КОМПАС (фрагментов или моделей), редактировать структуру библиотеки, получить доступ к коллекции элементов (документов) библиотеки.

Примечание:

Интерфейс библиотеки фрагментов или моделей системы КОМПАС можно получить у интерфейса IInsertsLibraries с помощью свойства IInsertsLibraries::Item.

IInsertsLibrary – свойства

ActiveFolder – Активная папка в окне библиотеки элементов

Функция не поддерживается

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

ActiveFolder iObject.ActiveFolder	=	Получить свойство (*)
ActiveFolder iObject.GetActiveFolder	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ActiveFolder (&ActiveFolder)	Получить свойство

Примечание:

Свойство позволяет получить активную папку в окне библиотеки.

Inserts – Коллекция элементов (документов) библиотеки

Интерфейс...

Тип данных: указатель на интерфейс IInserts

Синтаксис Automation:

Inserts = iObject.Inserts	Получить свойство (*)
Inserts = iObject.GetInserts	Получить свойство (**)

Синтаксис COM:

iObject->get_Inserts (&Inserts)	Получить свойство

Примечание:

Свойство позволяет получить коллекцию элементов (документов) библиотеки.

IInsertsLibrary – методы

AddFolder – Добавить папку в библиотеку элементов

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL AddFolder (BSTR PathFolder);

Синтаксис COM:

HRESULT AddFolder ([in] BSTR PathFolder, [out, retval] VARIANT_BOOL* pVal);

Входные параметры:

PathFolder	- путь к папке,
post	- TRUE - отложенное выполнение.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Добавляет новую папку в библиотеку элементов.

Delete – Удалить описание библиотеки

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL* PVal);

Возвращаемое значение :

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

RemoveFolder – Удалить папку из библиотеки элементов

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL RemoveFolder (BSTR PathFolder);

Синтаксис COM:

HRESULT RemoveFolder ([in] BSTR PathFolder, [out, retval] VARIANT_BOOL* pVal);

Входные параметры:

PathFolder	- путь к папке,
post	- TRUE - отложенное выполнение.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Удаляет папку из библиотеки элементов.

SetActiveFolder – Установить активную папку в окне библиотеки элементов (документов)

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL SetActiveFolder (BSTR PathFolder, BOOL LastIsFile, BOOL HaveLibName);

Синтаксис COM:

HRESULT SetActiveFolder ([in] BSTR PathFolder,
VARIANT_BOOL LastIsFile,
VARIANT_BOOL HaveLibName,
[out, retval] VARIANT_BOOL * pVal);

Входные параметры:

PathFolder	- путь к папке,

LastIsFile	TRUE - в пути имеется имя файла документа,
HaveLibName(BOOL)	- TRUE - в пути имеется имя библиотеки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет установить активную папку в окне библиотеки документов. Путь к папке может содержать или не содержать имя библиотеки и имя документа. Имя библиотеки, имя папки и имя документа разделяются символом "|". Если папка является вложенной, т. е. находится внутри другой папки, названия папок в пути отделяются символом "|".

Пример. Библиотека с именем 111, содержит папку с именем ddd, в которой находится файл с именем fragment. Путь к этому файлу должен быть задан следующим образом: 111|ddd|fragment.

Интерфейс IProceduresLibrary

Интерфейс прикладной библиотеки КОМПАС.

Иерархия:

IKompasAPIObject

ILibrary

IProceduresLibrary

Примечание:

1. Интерфейс позволяет работать с прикладной библиотекой (*.rtw или *.dll), зарегистрированной в Менеджере библиотек. Можно получить стиль отображения библиотеки (панель, окно, диалог, меню) и массив процедур, реализованных в библиотеке.
2. Данный интерфейс можно получить следующими способами:

- у интерфейса коллекции прикладных библиотек IProceduresLibraries с помощью свойства IProceduresLibraries::Item,
- у интерфейса ILibraryManager с помощью свойства ILibraryManager::CurrentLibrary, ILibraryManager::SystemControlStartLibrary.

IProceduresLibrary – свойства

AddIns – Признак AddIn-библиотеки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AddIns = iObject.AddIns	Получить свойство (*)

AddIns = iObject.GetAddIns Получить свойство (**)

Синтаксис COM:

iObject->get_AddIns (&AddIns) Получить свойство

Примечание:

Если библиотека является расширением, свойство вернет TRUE.

CurrentCommand – Текущая команда библиотеки

Интерфейс...

Тип данных: long

Синтаксис Automation:

CurrentCommand = Получить свойство (*)
iObject.CurrentCommand
CurrentCommand = Получить свойство (**)
iObject.GetCurrentCommand

Синтаксис COM:

iObject->get_CurrentCommand Получить свойство
(&CurrentCommand)

Примечание:

Свойство позволяет получить текущую команду библиотеки. Текущая команда - это команда, которая выполняется в данный момент времени.

Executable – Признак выполнения библиотеки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Executable = iObject.Executable Получить свойство (*)
Executable = Получить свойство (**)
iObject.GetExecutable()

Синтаксис COM:

iObject->get_Executable(Получить свойство
&Executable)

Примечание:

Если выполняется команда библиотеки или отрабатывается событие, на которое подписалась библиотека, то свойство вернет TRUE.

ExternalInterface – Получить указатель внешнего интерфейса

Интерфейс...

Тип данных: Указатель на интерфейс IDispatch

Синтаксис Automation:

```
ExternalInterface          =   Получить свойство (* )
Object.ExternalInterface
ExternalInterface          =   Получить свойство (**)
Object.GetExternalInterface()
```

Синтаксис COM:

```
Object.get_ExternalInterface(          Получить свойство
&ExternalInterface )
```

Примечание:

1. Свойство доступно только для чтения.
2. Addref на выдаваемый интерфейс не выполняется.
3. Внешний интерфейс можно использовать для обмена данными между двумя приложениями. Первое приложение передает интерфейс при вызове метода ILibrary::Execute.
4. После выполнения ILibrary::Execute свойство ExternalInterface возвращает NULL.

IconsFont – Имя файла шрифта для панели команд библиотеки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
IconsFont = Object.IconsFont       Получить свойство (* )
IconsFont                  =   Получить свойство (**)
Object.GetIconsFont()
```

Синтаксис COM:

```
Object.get_IconsFont(                  Получить свойство
&IconsFont )
```

Примечание:

Свойство доступно только для чтения.

LibraryName – Имя библиотеки, возвращаемое функцией LIBRARYNAME или LIBRARYID

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
LibraryName                 =   Получить свойство (* )
Object.LibraryName
LibraryName                 =   Получить свойство (**)
Object.GetLibraryName()
```

Синтаксис COM:

```
Object.get_LibraryName(                 Получить свойство
&LibraryName )
```

Примечание:

Свойство возвращает имя библиотеки, возвращаемое функцией LIBRARYNAME или LIBRARYID.

Отображаемое имя библиотеки может отличаться, если в библиотеке реализована функция DisplayLibraryName.

Для получения отображаемого имени библиотеки используется свойство ILibrary::Name.

При получении библиотеки по имени из коллекции можно использовать как отображаемое, так и внутреннее имя.

Свойство доступно только для чтения.

Procedures – Коллекция процедур библиотеки

Интерфейс...

Тип данных: указатель на интерфейс IProcedures

Синтаксис Automation:

```
Procedures = iObject.Procedures   Получить свойство (* )
Procedures                  =   Получить свойство (**)
iObject.GetProcedures
```

Синтаксис COM:

```
iObject->get_Procedures                 Получить свойство
(&Procedures)
```

Примечание:

Свойство позволяет получить коллекцию процедур библиотеки.

Style – Стиль отображения библиотеки

Интерфейс...

Тип данных: ksLibraryStyleEnum

Синтаксис Automation:

Style = iObject.Style	Получить свойство (*)
iObject.Style = Style	Установить свойство (*)
Style = iObject.GetStyle()	Получить свойство (**)
iObject.SetStyle (Style)	Установить свойство (**)

Синтаксис COM:

iObject->get_Style (&Style)	Получить свойство
iObject->put_Style (Style)	Установить свойство

Примечание:

Позволяет получить и установить стиль отображения библиотеки.

UniqueId – Идентификатор библиотеки

Интерфейс...

Тип данных: long

Синтаксис Automation:

UniqueId = iObject.UniqueId	Получить свойство (*)
UniqueId = iObject.GetUniqueId	Получить свойство (**)

Синтаксис COM:

iObject->get_UniqueId (&UniqueId)	Получить свойство

Примечание:

Свойство позволяет получить идентификатор библиотеки. Идентификатор присваивается библиотеке, когда она подключена к системе. В этот момент библиотека отображается в меню **Библиотеки** и в Менеджере библиотек рядом с именем библиотеки взводится «галочка». У подключенной библиотеки можно выполнять команды (процедуры).

IProceduresLibrary – методы

Delete – Удалить описание библиотеки

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL* PVal);

Возвращаемое значение :

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

SystemControlStart – Передать управление системе из под библиотеки

Интерфейс...

Синтаксис Automation:

ksSystemControlStartEnum SystemControlStart(BSTR menuCommand);

Синтаксис COM:

HRESULT SystemControlStart([in] BSTR menuCommand, [out, retval] ksSystemControlStartEnum* pVal);

Входные параметры:

PathFolder	- путь к папке,
post	- TRUE - отложенное выполнение.

Возвращаемое значение:

ksSystemControlStartEnum

Примечание:

Библиотечная функция отдает управление КОМПАС-ГРАФИК для интерактивной доработки документа. Возврат в библиотечную функцию будет осуществлен после нажатия команды в меню Библиотеки, обозначенной строкой menuCommand.

Если строка не задана, то ей автоматически будет присвоено значение "Вернуться в библиотеку" (функция вернет ksSCStoppedByMenuCommand). Или при вызове из этой же библиотеки SystemControlStop (функция вернет ksSCStopItself). Библиотека, находясь в этом режиме, как бы замирает на вызове SystemControlStart. Если этот режим вызван в другой библиотеке, то в данной библиотеке режим выполняться не будет (функция вернет ksSCStartedByAnotherLibrary). Только одна библиотека может находиться в режиме SystemControlStart.

SystemControlStop – Вернуть управление в библиотеку

Интерфейс...

Синтаксис Automation:

BOOL SystemControlStop();

Синтаксис COM:

HRESULT SystemControlStop ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет вернуть управление в библиотеку, если она находится в режиме SystemControlStart.

Интерфейс IProcedure

Интерфейс команды прикладной библиотеки КОМПАС.

Иерархия:

IKompasAPIObject

IProcedure

Примечание:

1. Интерфейс позволяет получить имя, идентификатор команды, ее папку в библиотеке, выполнить команду.
2. Данный интерфейс может быть получен у интерфейса коллекции команд IProcedures с помощью свойств IProcedures::Item, IProcedures::ItemByID.

IProcedure – свойства

ID – Идентификатор команды в прикладной библиотеке

Интерфейс...

Тип данных: long

Синтаксис Automation:

ID = iObject.ID	Получить свойство (*)
ID = iObject.GetID()	Получить свойство (**)

Синтаксис COM:

iObject->get_ID (&ID)	Получить свойство

Примечание:

Позволяет получить идентификатор команды в прикладной библиотеке.

LibraryFolder – Папка в прикладной библиотеке

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
LibraryFolder = Получить свойство (* )
iObject.LibraryFolder()
LibraryFolder = Получить свойство (**)
iObject.GetLibraryFolder()
```

Синтаксис COM:

```
iObject->get_LibraryFolder Получить свойство
(&LibraryFolder)
```

Примечание:

Позволяет получить папку в прикладной библиотеке.

Name – Имя команды

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
Name = iObject.Name Получить свойство (* )
Name = iObject.GetName() Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name (&Name) Получить свойство
```

Примечание:

Позволяет получить имя команды.

IProcedure – методы

Execute – Выполнить команду

Интерфейс...

Синтаксис Automation:

BOOL Execute (LPDISPATCH external, BOOL post);

Синтаксис COM:

HRESULT Execute([in] LPDISPATCH external,
[in] VARIANT_BOOL post,
[out, retval] VARIANT_BOOL* pVal);

Входные параметры:

external	- интерфейс для обмена данными между библиотекой, вызвавшей Execute, и библиотекой, в которой реализована команда. Интерфейс определен разработчиком библиотек. Может быть NULL,
post	- TRUE - отложенное выполнение.

Возвращаемое значение:

TRUE	- команда выполнена,
FALSE	- команда не выполнена.

Примечание:

Метод позволяет выполнить команду прикладной библиотеки. При этом проверяется лицензионная защита и может быть передан пользовательский интерфейс для обмена данными между библиотеками. Выполнение команды может быть отложенное или немедленное.

Интерфейс IInsert

Интерфейс элемента (фрагмента или модели) библиотеки элементов.

Иерархия:

IKompasAPIObject

IInsert

Описание:

Позволяет получить доступ к документу-вставке в библиотеку элементов (фрагментов или моделей).

Примечание:

Данный интерфейс можно получить у интерфейса IInserts с помощью свойства IInserts::Item или метода IInserts::Add.

IInsert – свойства

Comment – Комментарий

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Comment = iObject.Comment		Получить свойство (*)
iObject.Comment = Comment		Установить свойство (*)
Comment	=	Получить свойство (**)
iObject.GetComment()		
iObject.SetComment (Comment)		Установить свойство (**)

Синтаксис COM:

iObject->get_Comment (&Comment)	Получить свойство
iObject->put_Comment (Comment)	Установить свойство

Примечание:

Позволяет получить и установить комментарий для вставки. Комментарий может быть многострочным. Поэтому свойство возвращает или принимает переменную типа VARIANT на безопасный массив строк SAFEARRAY (VT_ARRAY | VT_BSTR).

InsertType – Тип элемента

Интерфейс...

Тип данных: из перечисления ksDocumentsLibraryInsertionTypeEnum

Синтаксис Automation:

InsertType = Object.InsertType		Получить свойство (*)
InsertType Object.GetInsertType()	=	Получить свойство (**)

Синтаксис COM:

Object.get_InsertType(&InsertType)	Получить свойство

Примечание:

Свойство доступно только для чтения

LibraryFolder – Папка в библиотеке элементов

Интерфейс...

Тип данных: строка

Синтаксис Automation:

LibraryFolder iObject.LibraryFolder()	=	Получить свойство (*)
LibraryFolder iObject.GetLibraryFolder()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LibraryFolder (&LibraryFolder)	Получить свойство

Примечание:

Свойство позволяет получить папку вставки внутри библиотеки. Папки отделяются символом "|".

Пример "детали|литье|фланец".

Name – Имя элемента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
iObject.Name = Name	Установить свойство (*)
Name = iObject.GetName()	Получить свойство (**)
iObject.SetName (Name)	Установить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство
iObject->put_Name (Name)	Установить свойство

Примечание:

Позволяет получить и установить имя вставки.

PathName – Полное имя элемента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

PathName = iObject.PathName()	Получить свойство (*)
PathName = iObject.GetPathName()	Получить свойство (**)

Синтаксис COM:

iObject->get_PathName (&PathName)	Получить свойство

Примечание:

Свойство позволяет получить полное имя вставки вместе с папками внутри библиотеки. Папки отделяются символом "|".

Пример "детали|литье|фланец".

IInsert – методы

Delete – Удалить элемент из библиотеки

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет удалить вставку из библиотеки документов.

Edit – Редактировать элемент

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

IKompasDocument* Edit();

Синтаксис COM:

HRESULT Edit ([out, retval] IKompasDocument** pVal);

Возвращаемое значение:

Указатель на интерфейс IKompasDocument созданной вставки.

Примечание:

Метод позволяет создать и добавить в библиотеку документ-вставку определенного типа. В библиотеки фрагментов можно вставлять фрагменты, в библиотеки моделей, как 3D модели, так и сборки. Имя вставки и папки в имени вставки отделяются символом "|".

Пример "детали|литье|фланец".

Интерфейс ICheckSum

Интерфейс контрольной суммы.

Иерархия:

IKompasAPIObject

ICheckSum

Описание.

Интерфейс позволяет формировать контрольную сумму с последовательности параметров, получаемых в прикладной библиотеке, и контролировать её работоспособность. Запись в файл полученной контрольной суммы после выполнения заданной последовательности команд библиотеки с последующим сравнением значений контрольной суммы, полученных в разное время, позволяет контролировать повторяемость результатов - получения одних и тех же параметров, а в случае внесения изменений в текст библиотеки - обнаружить скрытые ошибки (ошибки, приведшие к изменению получаемых параметров и, соответственно, изменению контрольной суммы). Если в библиотеке выполняется анализ документа, рекомендуется выполнять тестовые запуски библиотеки на одних и тех же документах.

Примечание:

Данный интерфейс можно получить от интерфейса IApplicationI, используя свойство IApplication::CheckSum.

ICheckSum – свойства

Result – Получить контрольную сумму в виде массива SAFEARRAY значений типа BYTE (VT_ARRAY | VT_UI1)

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Result = iObject.Result	Получить свойство (*)
Result = iObject.GetResult()	Получить свойство (**)

Синтаксис COM:

iObject->get_Result (&Result)	Получить свойство,

Примечание:

Свойство доступно только для чтения.

StrResult– Получить контрольную сумму в виде строки

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

StrResult = iObject.StrResult	Получить свойство (*)
StrResult = iObject.GetStrResult()	Получить свойство (**)

Синтаксис COM:

iObject->get_StrResult(&StrResult)	Получить свойство,

Примечание:

1. Свойство доступно только для чтения.
2. Строка контрольной суммы представляет собой последовательность из 32 буквенных латинских символов. Строка инициализированной (очищенной) контрольной суммы представляет собой последовательность aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.

Version – Версия контрольной суммы

Интерфейс...

Тип данных: long

Синтаксис Automation:

Version = iObject.Version	Получить свойство (*)
iObject.Version = Version	Установить свойство (*)
Version = iObject.GetVersion()	Получить свойство (**)
iObject.SetVersion(Version)	Установить свойство (**)

Синтаксис COM:

iObject->get_Version(&Version)	Получить свойство
iObject->put_Version(Version)	Установить свойство

Примечание:

Версия контрольной суммы это число, равное версии потока КОМПАС. ICheckSum используется для автоматизированного тестирования функционала.

Смысл тестирования заключается в получении значения контрольной суммы на тесте и сравнения его с эталонным. Контрольные суммы должны совпадать.

Причины различий могут быть следующими:

- ошибка в функционале,
- развитие функционала.

Введение свойства Version позволяет установить причину несовпадения контрольных сумм.

До введения свойства развитие функционала воспринималось как его ошибка.

Отслеживание версии контрольной суммы начинается с версии потока 0x0A001023L, что соответствует версии КОМПАС V10 SP2.

Получаем версию контрольной суммы всегда как версию потока КОМПАС.

Задать версию контрольной суммы можно следующими способами:

- непосредственно числом (версия потока),
- из перечисления ksCheckSumVersionEnum.

При получении интерфейса ICheckSum версия контрольной суммы инициируется версией потока текущей задачи КОМПАС.

ICheckSum – методы

Add – Добавить параметр к контрольной сумме

Интерфейс...

Синтаксис Automation:

void Add (VARIANT value, BOOL order);

Синтаксис COM:

HRESULT Add (VARIANT value, VARIANT_BOOL order);

Входные параметры:

value	- добавляемое значение,
order	- формировать суммарную контрольную сумму значений: TRUE - с учетом порядка следования значений, FALSE - без учета порядка следования значений.

Примечание:

Параметр value может принимать значения следующих типов:

- VT_INT,
- VT_I2,
- VT_I4,
- VT_R4,
- VT_R8,
- VT_UI1,
- VT_UINT,
- VT_BOOL,
- VT_BSTR,
- VT_ARRAY | VT_UI1.

AddInterface– Добавить к контрольной сумме параметр, получаемый из объекта по его интерфейсу

Интерфейс...

Синтаксис Automation:

BOOL AddInterface (LPUNKNOWN object, BOOL order);

Синтаксис COM:

HRESULT AddInterface(LPUNKNOWN object, VARIANT_BOOL order, VARIANT_BOOL * res);

Входные параметры:

object	- указатель на интерфейс,

order - формировать суммарную контрольную сумму значений:
TRUE - с учетом порядка следования значений,
FALSE - без учета порядка следования значений.

Возвращаемое значение:

TRUE - в случае успешного получения контрольной суммы присланного объекта и добавления полученного значения в суммарную контрольную сумму,
FALSE - в случае неудачи.

Примечание:

1. Метод получает значение контрольной суммы с присланного объекта по его интерфейсу и добавляет полученное значение в суммарную контрольную сумму.
2. Метод рекомендуется только для контроля интерфейсов API. Значение добавляемой контрольной суммы зависит от реализации интерфейса, и при развитии интерфейса (увеличении количества свойств и методов) контрольная сумма изменяется.
3. Метод реализован не для всех интерфейсов API.

AddReference – Добавить к контрольной сумме параметр, получаемый из объекта по его указателю reference

Интерфейс...

Синтаксис Automation:

BOOL AddReference (long object, long doc, BOOL order);

Синтаксис COM:

HRESULT AddReference (long object, long doc, VARIANT_BOOL order, VARIANT_BOOL * res);

Входные параметры:

object - указатель reference объекта,
doc - указатель reference документа,
order - формировать суммарную контрольную сумму значений:
TRUE - с учетом порядка следования значений,
FALSE - без учета порядка следования значений.

Возвращаемое значение:

TRUE - в случае успешного получения контрольной суммы присланного объекта и добавления полученного значения в суммарную контрольную сумму,
FALSE - в случае неудачи.

Примечание:

1. Метод получает значение контрольной суммы с присланного объекта по значению его указателя reference и добавляет полученное значение в суммарную контрольную сумму.
2. Метод рекомендуется только для контроля API. Значение добавляемой контрольной суммы зависит от реализации API, со временем добавляемая контрольная сумма изменяется.

Clear– Очистить контрольную сумму

Интерфейс...

Синтаксис Automation:

void Clear();

Синтаксис COM:

HRESULT Clear();

Примечание:

Строка инициализированной (очищенной) контрольной суммы представляет собой последовательность aaaaaaaaaaaaaaaaaaaaaaaaaaaaaaaa.

Интерфейс IInserts

Интерфейс коллекции элементов(документов) библиотеки.

Иерархия:

IKompasAPIObject

IKompasCollection

IInserts

Примечание:

Интерфейс позволяет получить документы, входящие в библиотеку. Интерфейс можно получить при помощи свойства IInsertsLibrary::Inserts от интерфейса библиотеки элементов IInsertsLibrary.

IInserts – свойства

Item – Элемент, заданный по имени или по индексу

Интерфейс...

Тип данных: указатель на интерфейс элемента (документа) в библиотеке элементов IInsert

Синтаксис Automation:

Item = iObject.Item (v)	Получить свойство (*)
Item = iObject.GetItem (v)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (v. &Item)	Получить свойство

Входные параметры:

Index (VARIANT)	Index (VARIANT) - если передано целочисленное значение, то оно интерпретируется, как индекс документа в коллекции, - если передано строковое значение, то оно интерпретируется, как имя документа.

Примечание:

Свойство позволяет получить интерфейс вставки библиотеки документов по имени документа или по индексу документа в коллекции вставок.

IInserts – методы

Add – Создать и добавить элемент в коллекцию

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

IInsert* Add(BSTR Name, DocumentTypeEnum docType);

Синтаксис COM:

HRESULT Add ([in] BSTR Name,
DocumentTypeEnum docType,
[out, retval] IInsert** Result);

Входные параметры:

Name	- имя вставки в библиотеке.
docType (DocumentTypeEnum)	- тип документа. Актуально для библиотек моделей (*.l3d). В эти библиотеки можно добавлять как сборки, так и модели. Если docType==ksDocumentPart, то это признак, что нужно добавить 3D модель.

Возвращаемое значение:

- интерфейс IInsert созданной вставки.

Примечание:

Метод позволяет создать и добавить в библиотеку документ-вставку определенного типа. В библиотеки фрагментов можно вставлять фрагменты, в библиотеки моделей, как 3D модели, так и сборки. Имя вставки и названия папок в имени вставки отделяются символом "|".

Пример "детали|литье|фланец".

Интерфейс IInsertsLibraries

Интерфейс коллекции библиотек элементов (документов) системы КОМПАС.

Иерархия:

IKompasAPIObject

IKompasCollection

IInsertsLibraries

Описание:

Позволяет получить коллекцию библиотек элементов (документов) системы КОМПАС (фрагментов или моделей) и получить интерфейс каждой из библиотек коллекции.

Примечание:

1. Интерфейс коллекции библиотек фрагментов системы КОМПАС можно получить у интерфейса ILibraryManager с помощью свойства ILibraryManager::FragmentsLibraries.
2. Интерфейс на коллекцию библиотек моделей системы КОМПАС можно получить у интерфейса ILibraryManager с помощью свойства ILibraryManager::ModelsLibraries.

IInsertsLibraries – свойства

Item – Элемент, заданный по имени или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IInsertsLibrary

Синтаксис Automation:

Item = iObject.Item (v)	Получить свойство (*)
Item = iObject.GetItem (v)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (v. &Item)	Получить свойство

Входные параметры:

Index (VARIANT)	- если передано целочисленное значение, то оно интерпретируется как индекс библиотеки в коллекции, - если передано строковое значение, то оно интерпретируется как имя библиотеки.

Примечание:

Свойство позволяет получить интерфейс библиотеки документов по имени библиотеки или по индексу библиотеки в коллекции библиотек.

IInsertsLibraries – методы

Add – Добавить элемент в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(LPCTSTR PathName, LPCTSTR LibManagerFolder);

Синтаксис COM:

HRESULT Add(BSTR PathName, BSTR LibManagerFolder, IInsertsLibrary** Result);

Входные параметры:

PathName	- путь к библиотеке документов,
LibManagerFolder	- раздел в менеджере библиотек.

Возвращаемое значение:

- указатель на интерфейс библиотеки документов IInsertsLibrary, библиотека регистрируется в менеджере библиотек (В V17 Конфигуратор).

Интерфейс IProcedures

Интерфейс коллекции процедур прикладной библиотеки КОМПАС.

Иерархия:

IKompasAPIObject

IKompasCollection

IProcedures

Примечание:

Интерфейс позволяет получить команды, входящие в библиотеку процедур.

Интерфейс можно получить при помощи свойства IProceduresLibrary::Procedures у интерфейса прикладной библиотеки IProceduresLibrary.

IProcedures – свойства

Item – Команда, заданная по имени или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IProcedure

Синтаксис Automation:

Item = iObject.Item (v)	Получить свойство (*)
Item = iObject.GetItem (v)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (v. &Item)	Получить свойство

Входные параметры:

Index (VARIANT)	Index (VARIANT), - если передано целочисленное значение, то оно интерпретируется как индекс процедуры в коллекции, - если передано строковое значение, то оно интерпретируется как имя процедуры.

Примечание:

Свойство позволяет получить интерфейс библиотечной команды по имени или по индексу в коллекции команд.

ItemByID – Команда, заданная по идентификатору команды

Интерфейс...

Тип данных: указатель на интерфейс IProcedure

Синтаксис Automation:

Item = iObject.ItemByID (id)	Получить свойство (*)
Item = iObject.GetItemByID (id)	Получить свойство (**)

Синтаксис COM:

iObject->get_ItemByID (id, &Item)	Получить свойство

Входные параметры:

ID (ID)	- идентификатор команды.

Примечание:

Свойство позволяет получить интерфейс библиотечной команды по идентификатору команды в коллекции команд.

Интерфейс IProceduresLibraries

Интерфейс коллекции прикладных библиотек КОМПАС.

Иерархия:

IKompasAPIObject

IKompasCollection

IProceduresLibraries

Примечание:

1. Интерфейс позволяет получить прикладные библиотеки, зарегистрированные в Менеджере библиотек.
2. Данный интерфейс можно получить у интерфейса менеджера библиотек ILibraryManager с помощью свойства ILibraryManager::ProceduresLibraries.

IProceduresLibraries – свойства

Item – Прикладная библиотека, заданная по имени или по индексу

Интерфейс...

Тип данных: указатель на интерфейс IProceduresLibrary

Синтаксис Automation:

Item = iObject.Item (v)	Получить свойство (*)
Item = iObject.GetItem (v)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (v, &Item)	Получить свойство

Входные параметры:

v (VARIANT)	- если передано целочисленное значение, то оно интерпретируется как индекс библиотеки в коллекции, - если передано строковое значение, то оно интерпретируется как имя библиотеки.

Примечание:

Свойство позволяет получить интерфейс прикладной библиотеки по имени библиотеки или по индексу библиотеки в коллекции библиотек.

IProceduresLibraries – методы

Add – Добавить элемент в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(LPCTSTR PathName, LPCTSTR LibManagerFolder);

Синтаксис COM:

HRESULT Add(BSTR PathName, BSTR LibManagerFolder, IProceduresLibrary** Result);

Входные параметры:

PathName	- путь к библиотеке документов,
LibManagerFolder	- раздел в менеджере библиотек.

Возвращаемое значение:

- указатель на интерфейс библиотеки документов IProceduresLibrary, библиотека регистрируется в менеджере библиотек (В V17 Конфигуратор).

Работа с настройками

Интерфейс IDrawingDocumentSettings

Настройки чертежа.

Иерархия:

IDispatch

IDrawingDocumentSettings

Описание:

Настройки могут быть для текущего документа (можно получить у документа) и для новых документов (можно получить у приложения).

Примечание:

Данный интерфейс можно получить от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::DocumentSettings с последующим использованием метода IUnknown QueryInterface для преобразования от интерфейса IDocumentSettings к данному интерфейсу.

IDrawingDocumentSettings – свойства

SheetAutoCount – Автоматическое определение количества листов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SheetAutoCount Object.SheetAutoCount	=	Получить свойство (*)
Object.SheetAutoCount SheetAutoCount	=	Установить свойство (*)
SheetAutoCount Object.GetSheetAutoCount()	=	Получить свойство (**)
Object.SetSheetAutoCount(SheetAutoCount)		Установить свойство (**)

Синтаксис COM:

Object.get_SheetAutoCount(&SheetAutoCount)	Получить свойство,
Object.put_SheetAutoCount(SheetAutoCount)	Установить свойство.

Примечание:

Свойство, позволяющее автоматически подсчитать фактическое количество листов текущего документа и занести полученное число в соответствующую графу основной надписи.

SheetAutoNumber – Автоматическая нумерация листов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SheetAutoNumber Object.SheetAutoNumber	=	Получить свойство (*)
Object.SheetAutoNumber SheetAutoNumber	=	Установить свойство (*)
SheetAutoNumber Object.GetSheetAutoNumber()	=	Получить свойство (**)
Object.SetSheetAutoNumber(SheetAutoNumber)		Установить свойство (**)

Синтаксис COM:

Object.get_SheetAutoNumber(&SheetAutoNumber)	Получить свойство,
Object.put_SheetAutoNumber(SheetAutoNumber)	Установить свойство.

Примечание:

Опция, управляющая автонумерацией листов.

- Если опция включена, то всем листам документа автоматически присваиваются порядковые номера.
- Если опция выключена, то графа *Номер листа* в основной надписи текущего документа не заполняется.

Вы можете ввести произвольный номер для каждого листа документа.

SheetsCount – Количество листов

Интерфейс...

Тип данных: long

Тип данных: BOOL

Синтаксис Automation:

SheetsCount = Object.SheetsCount	Получить свойство (*)
Object.SheetsCount = SheetsCount	Установить свойство (*)
SheetsCount = Object.GetSheetsCount()	Получить свойство (**)
Object.SetSheetsCount(SheetsCount)	Установить свойство (**)

Синтаксис COM:

Object.get_SheetsCount(&SheetsCount)	Получить свойство,
Object.put_SheetsCount(SheetsCount)	Установить свойство.

Примечание:

Свойство, позволяющее задать произвольное число, которое будет занесено в графу *Количество листов* основной надписи каждого листа текущего документа.

Таким образом можно сформировать документ, являющийся частью другого документа.

Свойство используется если отключена автоматическая нумерация листов (IDrawingDocumentSettings::SheetAutoCount).

SheetFirstNumber – Номер первого листа

Интерфейс...

Тип данных: long

Синтаксис Automation:

SheetFirstNumber Object.SheetFirstNumber	=	Получить свойство (*)
Object.SheetFirstNumber SheetFirstNumber	=	Установить свойство (*)
SheetFirstNumber Object.GetSheetFirstNumber()	=	Получить свойство (**)
Object.SetSheetFirstNumber(SheetFirstNumber)		Установить свойство (**)

Синтаксис COM:

Object.get_SheetFirstNumber(&SheetFirstNumber)	Получить свойство,
Object.put_SheetFirstNumber(SheetFirstNumber)	Установить свойство.

Примечание:

Свойство задает номер, с которого начнется автоматическая нумерация.

По умолчанию он равен единице. Вы можете задать нужное значение.

Свойство используется при включенной опции *Автоматическая нумерация листов* (IDrawingDocumentSettings::SheetAutoNumber).

TechnicalDemandSynchronize – Синхронизировать технические требования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TechnicalDemandSynchronize = Object. TechnicalDemandSynchronize	Получить свойство (*)
Object. TechnicalDemandSynchronize = TechnicalDemandSynchronize	Установить свойство (*)
TechnicalDemandSynchronize = Object.Get TechnicalDemandSynchronize ()	Получить свойство (**)
Object.Set TechnicalDemandSynchronize (TechnicalDemandSynchronize)	Установить свойство (**)

Синтаксис COM:

Object.get_TechnicalDemandSynchronize (&_TechnicalDemandSynchronize)	Получить свойство
Object.put_TechnicalDemandSynchronize (TechnicalDemandSynchronize)	Установить свойство

Примечание:

Свойство позволяет при создании ассоциативного вида синхронизировать технические требования чертежа с техническими требованиями документа 3D, с которого создается вид.

Интерфейс IFragmentDocumentSettings

Настройки фрагмента.

Иерархия:

IKompasAPIObject

IDocumentSettings

IDocument2DSettings

IFragmentDocumentSettings

Описание:

Настройки могут быть для текущего документа (можно получить у документа) и для новых документов (можно получить у приложения).

Примечание:

Данный интерфейс можно получить от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::DocumentSettings с последующим использованием метода IUnknown QueryInterface для преобразования от интерфейса IDocumentSettings к данному интерфейсу.

Интерфейс INewDocument3DSettings

Настройки новых 3D документов.

Иерархия:

IDispatch

INewDocument3DSettings

Примечание:

Интерфейс можно получить у интерфейса INewPartDocumentSettings с помощью метода IUnknown::QueryInterface.

INewDocument3DSettings – свойства

ColorParam – Параметры цвета

Интерфейс...

Тип данных: указатель на интерфейс IColorParam7

Синтаксис Automation:

ColorParam = Object. ColorParam	Получить свойство (*)
ColorParam = Object.GetColorParam()	Получить свойство (**)

Синтаксис COM:

Object.get_ColorParam(&ColorParam)	Получить свойство

Примечание:

1. Свойство позволяет получить интерфейс параметров цвета объекта.
2. Свойство доступно только для чтения.

Интерфейс INewPartDocumentSettings

Настройки новых документов – деталей.

Иерархия:

IDispatch

INewPartDocumentSettings

Примечание:

Интерфейс можно получить с помощью свойства ISystemSettings::NewDocumentSettings.

INewPartDocumentSettings – свойства

Density – Модель: Деталь: Свойства: Плотность (базовые ед.)

Интерфейс...

Тип данных: double

Синтаксис Automation:

Density = Object.Density	Получить свойство (*)
Object.Density = Density	Установить свойство (*)
Density = Object.GetDensity()	Получить свойство (**)
Object.SetDensity(Density)	Установить свойство (**)

Синтаксис COM:

Object.get_Density(&Density)	Получить свойство
Object.put_Density(Density)	Установить свойство

Свойство позволяет устанавливать и получать плотность для новых создаваемых деталей.

HatchParam – Параметры штриховки

Интерфейс...

Тип данных: указатель на интерфейс IHatchParam

HatchParam = Object.HatchParam	Получить свойство (*)
HatchParam = Object.GetHatchParam()	Получить свойство (**)

Синтаксис COM:

Object.get_HatchParam(&HatchParam)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Material – Новые документы: Модель: Деталь: Свойства: Материал

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Material = Object.Material	Получить свойство (*)
Object.Material = Material	Установить свойство (*)
Material = Object.GetMaterial()	Получить свойство (**)
Object.SetMaterial(Material)	Установить свойство (**)

Синтаксис COM:

Object.get_Material(&Material)	Получить свойство
Object.put_Material(Material)	Установить свойство

Свойство позволяет устанавливать и получать материал для новых создаваемых деталей.

MaterialLocation – Новые документы: Модель: Деталь: Свойства: Идентификатор материала

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

MaterialLocation = Object.MaterialLocation	Получить свойство (*)
Object.MaterialLocation = MaterialLocation	Установить свойство (*)
MaterialLocation = Object.GetMaterialLocation()	Получить свойство (**)
Object.SetMaterialLocation(MaterialLocation)	Установить свойство (**)

Синтаксис COM:

Object.get_MaterialLocation(&MaterialLocation)	Получить свойство
Object.put_MaterialLocation(MaterialLocation)	Установить свойство

Свойство позволяет устанавливать и получать идентификатор материала для новых создаваемых деталей.

Интерфейс ILibItemSettings

Интерфейс для выбора состояний элементов библиотек в настройках

Иерархия

IDispatch

ILibItemSettings

Описание:

Позволяет управлять состоянием элементов библиотек свойств.

Примечание:

Дополнительный интерфейс. Данный интерфейс можно получить у ILibArraySettings посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

ILibItemSettings – свойства

ItemCount – Количество подключенных элементов

Интерфейс...

Тип данных: long

ItemCount = Object.ItemCount		Получить свойство (*)
ItemCount = Object.GetItemCount()		Получить свойство (**)

Синтаксис COM:

Object.get_ItemCount(&ItemCount)	Получить свойство

Примечание:

Свойство только для чтения.

ILibItemSettings – методы

GetItem – Получить элемент библиотеки и признак использования по индексу

Интерфейс...

Синтаксис Automation:

double GetItem(VARIANT index, BOOL * use);

Синтаксис COM:

HRESULT GetItem(VARIANT Index, BOOL * Use, double * Result);

Входные параметры:

Index	- индекс в массиве VT_I4, либо уникальный номер элемента VT_R8,
use	- признак использования.

Возвращаемое значение:

Уникальный номер элемента	- в случае успешного завершения,
-1	- в случае неудачи.

GetItems – Получить массив всех элементов библиотек и признаков использования

Интерфейс...

Синтаксис Automation:

BOOL GetItems(VARIANT * uniqIds, VARIANT * uses);

Синтаксис COM:

HRESULT GetItems(VARIANT * UniqIds, VARIANT * Uses, BOOL * Result);

Входные параметры:

UniqIds	- массив уникальных номеров библиотек VT_ARRAY|VT_R8,

uses - массив признаков использования VT_ARRAY|VT_BOOL.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetItemsEx – Получить массив всех элементов библиотек и признаков использования

Интерфейс...

Синтаксис Automation:

BOOL GetItemsEx(VARIANT * UniqIds, VARIANT * Uses, VARIANT * ItemNames, VARIANT * FileNames);

Синтаксис COM:

HRESULT GetItemsEx(VARIANT * UniqIds, VARIANT * Uses, VARIANT * ItemNames, VARIANT * FileNames, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Выходные параметры:

UniqIds - массив уникальных номеров библиотек VT_ARRAY|VT_R8,
Uses - массив признаков использования VT_ARRAY|VT_BOOL,
ItemNames - массив имен элементов VT_ARRAY|VT_BSTR,
FileNames - массив имен файлов библиотек VT_ARRAY|VT_BSTR.

SetItem – Установить признак использования

Интерфейс...

Синтаксис Automation:

BOOL SetItem(VARIANT index, BOOL use);

Синтаксис COM:

HRESULT SetItem(VARIANT Index, BOOL Use, BOOL * Result);

Входные параметры:

Index - индекс в массиве VT_I4, либо уникальный номер элемента VT_R8,
use - признак использования.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс IDocumentSettings

Настройки документа.

Иерархия:

IKompasAPIObject

IDocumentSettings

Описание:

Настройки могут быть для текущего документа (можно получить у документа) и для новых документов (можно получить у приложения).

Примечание:

Данный интерфейс можно получить от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::DocumentSettings.

Интерфейс IDocument2DSettings

Настройки графического документа.

Иерархия:

IKompasAPIObject

IDocumentSettings

IDocument2DSettings

Описание:

Настройки могут быть для текущего документа (можно получить у документа) и для новых документов (можно получить у приложения).

Примечание:

1. Данный интерфейс можно получить от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::DocumentSettings с последующим использованием метода IUnknown QueryInterface для преобразования от интерфейса IDocumentSettings к данному интерфейсу.
2. Данный интерфейс можно получить от интерфейса настроек чертежа IDrawingDocumentSettings и интерфейса настроек фрагмента IFragmentDocumentSettings с помощью метода IUnknown QueryInterface.

IDocument2DSettings – свойства

CompositionInherit – Наследовать состав родительского объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CompositionInherit iObject.CompositionInherit	=	Получить свойство (*)
iObject.CompositionInherit CompositionInherit	=	Установить свойство (*)
CompositionInherit iObject.GetCompositionInherit()	=	Получить свойство (**)
iObject.SetCompositionInherit (CompositionInherit)		Установить свойство (**)

Синтаксис COM:

iObject->get_CompositionInherit (&CompositionInherit)	Получить свойство,
iObject->put_CompositionInherit (CompositionInherit)	Установить свойство.

Примечание:

1. Данное свойство используется, если свойству LayersGroupWay установлено значение wgLayersCharacteristics.
2. Если установлено значение TRUE, то при создании группы свойств слоев будет выполнено следующее:

- если родительским элементом является группа свойств, то в новую группу будут скопированы все слои родительской группы и назначаемые им в данной группе свойства,
- если родительским элементом является вид, то новая группа будет содержать все слои вида и их свойства,
- если родительским элементом является корневой элемент Документ, то новая группа будет содержать все слои всех видов.

3. Изменение способа группировки в документе приводит к необратимым изменениям созданных групп.

LayersGroupWay – Способ группировки слоев

Интерфейс...

Тип данных: из перечисления LayersGroupWayEnum

Синтаксис Automation:

LayersGroupWay iObject.LayersGroupWay	=	Получить свойство (*)
iObject.LayersGroupWay LayersGroupWay	=	Установить свойство (*)
LayersGroupWay iObject.GetLayersGroupWay()	=	Получить свойство (**)
iObject.SetLayersGroupWay (LayersGroupWay)		Установить свойство (**)

Синтаксис COM:

iObject->get_LayersGroupWay (&LayersGroupWay)	Получить свойство,
iObject->put_LayersGroupWay (LayersGroupWay)	Установить свойство.

Примечание:

1. Если данному свойству установлено значение wgLayers, то в документе будут создаваться и использоваться статические и динамические (фильтры) группы слоев.
2. Если свойству установлено значение wgLayersCharacteristics, то в документе будут создаваться и использоваться группы свойств слоев.
3. Изменение способа группировки в документе приводит к необратимым изменениям созданных групп и фильтров.

Интерфейс IDocument3DSettings

Интерфейс настроек 3D документа.

Иерархия:

IDispatch

IKompasAPIObject

IDocumentSettings

IDocument3DSettings

Интерфейс является дополнительным к интерфейсу настроек документа IDocumentSettings (можно получить с помощью QueryInterface).

IDocument3DSettings – свойства

AccuracyMassProperties – Точность расчета МЦХ

Интерфейс...

Тип данных: double

Синтаксис Automation:

AccuracyMassProperties Object.AccuracyMassProperties	=	Получить свойство (*)
Object.AccuracyMassProperties AccuracyMassProperties	=	Установить свойство (*)
AccuracyMassProperties Object.GetAccuracyMassProperties()	=	Получить свойство (**)
Object.SetAccuracyMassProperties (AccuracyMassProperties)		Установить свойство (**)

Синтаксис COM:

Object->get_AccuracyMassProperties (&AccuracyMassProperties)	Получить свойство,
Object->put_AccuracyMassProperties (AccuracyMassProperties)	Установить свойство.

Примечание:

Свойство позволяет увеличить или уменьшить погрешность вычисления массово-центровочных характеристик модели и площадей граней.

AccuracyModelDisplaying – Точность отрисовки

Интерфейс...

Тип данных: double

Синтаксис Automation:

AccuracyModelDisplaying Object.AccuracyModelDisplaying	=	Получить свойство (*)
Object.AccuracyModelDisplaying AccuracyModelDisplaying	=	Установить свойство (*)
AccuracyModelDisplaying Object.GetAccuracyModelDisplaying()	=	Получить свойство (**)
Object.SetAccuracyModelDisplaying (AccuracyModelDisplaying)		Установить свойство (**)

Синтаксис COM:

Object->get_AccuracyModelDisplaying (&AccuracyModelDisplaying)	Получить свойство,
Object->put_AccuracyModelDisplaying (AccuracyModelDisplaying)	Установить свойство.

Примечание:

Свойство позволяет увеличить или уменьшить точность аппроксимации криволинейных ребер и линий очерка модели отрезками прямых линий, а криволинейных поверхностей — треугольниками. Чем выше точность, тем более «гладким» выглядит изображение и тем больше времени требуется на расчеты.

Интерфейс ILibArraySettings

Интерфейс для выбора состояний библиотек в настройках

Иерархия:

IKompasAPIObject

ILibArraySettings

ILibItemSettings

Описание:

Интерфейс позволяет задавать параметры свойства.

Примечание:

1. Настройка списка подключенных библиотек стилей отчета - ISystemSettings::ReportStyleListSettings.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс ILibItemSettings.

ILibArraySettings – свойства

LibraryCount – Количество подключенных библиотек

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
LibraryCount                =   Получить свойство (* )
Object.LibraryCount
LibraryCount                =   Получить свойство (**)
Object.GetLibraryCount()
```

Синтаксис COM:

```
Object.get_LibraryCount(                    Получить свойство
&LibraryCount )
```

Примечание:

Свойство только для чтения.

ILibArraySettings – методы

AddLibrary – Добавить библиотеку и признак использования

Интерфейс...

Синтаксис Automation:

BOOL AddLibrary(LPCTSTR fileName, BOOL use);

Синтаксис COM:

HRESULT AddLibrary(BSTR FileName, BOOL Use, BOOL* Result);

Входные параметры:

```
fileName        - полный путь к библиотеке на диске VT_BSTR,
use             - признак использования.
```

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetLibrary – Получить имя библиотеки и признак использования по индексу

Интерфейс...

Синтаксис Automation:

BSTR GetLibrary(long index, BOOL * use);

Синтаксис COM:

HRESULT GetLibrary(long Index, BOOL * Use, BSTR * Result);

Входные параметры:

index	- индекс в массиве,
use	- признак использования.

Возвращаемое значение:

- Полный путь к библиотеке на диске.

GetLibrarys – Получить массив всех библиотек и флагов использования

Интерфейс...

Синтаксис Automation:

BOOL GetLibrarys(VARIANT * fileNames, VARIANT * uses);

Синтаксис COM:

HRESULT GetLibrarys(VARIANT * FileNames, VARIANT * Uses, BOOL * Result);

Входные параметры:

fileNames	- массив имён файлов на диске VT_ARRAY|VT_BSTR,
uses	- массив признаков использования VT_ARRAY|VT_BOOL.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

RemoveLibrary – Удалить библиотеку и признак использования

Интерфейс...

Синтаксис Automation:

BOOL RemoveLibrary(VARIANT index);

Синтаксис COM:

HRESULT RemoveLibrary(VARIANT Index, BOOL * Result);

Входные параметры:

Index	- полный путь к библиотеке на диске VT_BSTR, либо индекс в массиве VT_I4.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetLibraryUse – Установить флаг использования по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetLibraryUse(VARIANT index, BOOL use);

Синтаксис COM:

HRESULT SetLibraryUse(VARIANT Index, BOOL Use, BOOL * Result);

Входные параметры:

Index	- Полный путь к библиотеке на диске VT_BSTR, либо индекс в массиве VT_I4,
use	- признак использования.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс ISystemSettings

Параметры системы.

Иерархия:

IDispatch

IKompasAPIObject

ISystemSettings

Примечание:

Данный интерфейс можно получить от интерфейса IApplication с помощью метода IApplication::SystemSettings.

ISystemSettings – свойства

AssociationViewAutoSaveBeforeRebuild – Общее: Управление системой: Выполнять автосохранение перед обработкой

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AssociationViewAutoSaveBeforeRebuild = Object.AssociationViewAutoSaveBeforeRebuild	=	Получить свойство (*)
Object.AssociationViewAutoSaveBeforeRebuild = AssociationViewAutoSaveBeforeRebuild	=	Установить свойство (*)
AssociationViewAutoSaveBeforeRebuild = Object.GetAssociationViewAutoSaveBeforeRebuild()	=	Получить свойство (**)
Object.SetAssociationViewAutoSaveBeforeRebuild(AssociationViewAutoSaveBeforeRebuild)		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationViewAutoSaveBeforeRebuild(&AssociationViewAutoSaveBeforeRebuild)	Получить свойство
Object.put_AssociationViewAutoSaveBeforeRebuild(AssociationViewAutoSaveBeforeRebuild)	Установить свойство

AssociationViewRebuildParallel – Общее: Управление системой: Разрешить параллельную обработку

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AssociationViewRebuildParallel = Object.AssociationViewRebuildParallel	=	Получить свойство (*)
Object.AssociationViewRebuildParallel = AssociationViewRebuildParallel	=	Установить свойство (*)

AssociationViewRebuildParallel Object.GetAssociationViewRebuildParallel()	=	Получить свойство (**)
Object.SetAssociationViewRebuildParallel(AssociationViewRebuildParallel)		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationViewRebuildParallel(&AssociationViewRebuildParallel)	Получить свойство
Object.put_AssociationViewRebuildParallel(AssociationViewRebuildParallel)	Установить свойство

AssociationViewRebuildParallelLowPriority – Общее: Управление системой: Разрешить параллельную обработку. С пониженным приоритетом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AssociationViewRebuildParallelLowPriority Object.AssociationViewRebuildParallelLowPriority	=	Получить свойство (*)
Object.AssociationViewRebuildParallelLowPriority AssociationViewRebuildParallelLowPriority	=	Установить свойство (*)
AssociationViewRebuildParallelLowPriority Object.GetAssociationViewRebuildParallelLowPriority()	=	Получить свойство (**)
Object.SetAssociationViewRebuildParallelLowPriority(AssociationViewRebuildParallelLowPriority)		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationViewRebuildParallelLowPriority(&AssociationViewRebuildParallelLowPriority)	Получить свойство
Object.put_AssociationViewRebuildParallelLowPriority(AssociationViewRebuildParallelLowPriority)	Установить свойство

EnablesAddSystemDelimersInMarking – Добавлять системные разделители в обозначение в функциях API

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnablesAddSystemDelimersInMarking Object.EnablesAddSystemDelimersInMarking	=	Получить свойство(*)
Object.EnablesAddSystemDelimersInMarking EnablesAddSystemDelimersInMarking	=	Установить свойство (*)
EnablesAddSystemDelimersInMarking Object.GetEnablesAddSystemDelimersInMarking()	=	Получить свойство (**)
Object.SetEnablesAddSystemDelimersInMarking(EnablesAddSystemDelimersInMarking)		Установить свойство (**)

Синтаксис COM:

Object.get_EnablesAddSystemDelimersInMarking(&EnablesAddSystemDelimersInMarking)	Получить свойство
Object.put_EnablesAddSystemDelimersInMarking(EnablesAddSystemDelimersInMarking)	Установить свойство

Примечание:

Если в обозначении есть системная часть, то обозначение может возвращаться без системных разделителей.

UseHardwareAcceleration – Общее: Управление системой: Использовать аппаратное ускорение

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseHardwareAcceleration Object.UseHardwareAcceleration	=	Получить свойство (*)
UseHardwareAcceleration Object.GetUseHardwareAcceleration()	=	Получить свойство (**)

Синтаксис COM:

Object.get_UseHardwareAcceleration(&UseHardwareAcceleration)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ModelRenderType – Общее: Управление системой: Вариант отрисовки

Интерфейс...

Тип данных: из перечисления ksModelRenderTypeEnum.

Синтаксис Automation:

ModelRenderType = Object.ModelRenderType	=	Получить свойство (*)
Object.ModelRenderType = ModelRenderType	=	Установить свойство (*)
ModelRenderType = Object.GetModelRenderType()	=	Получить свойство (**)
Object.SetModelRenderType(ModelRenderType)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelRenderType(&ModelRenderType)	Получить свойство
Object.put_ModelRenderType(ModelRenderType)	Установить свойство

ModelStepMoveDetail – Редактор моделей: Шаг перемещения изображения модели (% окна)

Интерфейс...

Тип данных: double

Синтаксис Automation:

ModelStepMoveDetail = Object.ModelStepMoveDetail	=	Получить свойство (*)
Object.ModelStepMoveDetail = ModelStepMoveDetail	=	Установить свойство (*)
ModelStepMoveDetail = Object.GetModelStepMoveDetail ()	=	Получить свойство (**)
Object.SetModelStepMoveDetail (ModelStepMoveDetail)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelStepMoveDetail(&ModelStepMoveDetail)	Получить свойство
Object.put_ModelStepMoveDetail(ModelStepMoveDetail)	Установить свойство

ModelStepRotateDetail – Редактор моделей: Шаг угла поворота модели (гр.)

Интерфейс...

Тип данных: double

Синтаксис Automation:

ModelStepRotateDetail = Object.ModelStepRotateDetail	=	Получить свойство (*)
Object.ModelStepRotateDetail = ModelStepRotateDetail	=	Установить свойство (*)
ModelStepRotateDetail = Object.GetModelStepRotateDetail()	=	Получить свойство (**)
Object.SetModelStepRotateDetail(ModelStepRotateDetail)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelStepRotateDetail(&ModelStepRotateDetail)	Получить свойство
Object.put_ModelStepRotateDetail(ModelStepRotateDetail)	Установить свойство

ModelScaleFactor – Редактор моделей: Коэффициент изменения масштаба

Интерфейс...

Тип данных: double

Синтаксис Automation:

ModelScaleFactor = Object.ModelScaleFactor	=	Получить свойство (*)
Object.ModelScaleFactor = ModelScaleFactor	=	Установить свойство (*)
ModelScaleFactor = Object.GetModelScaleFactor()	=	Получить свойство (**)
Object.SetModelScaleFactor(ModelScaleFactor)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelScaleFactor(&ModelScaleFactor)	Получить свойство

Object.put_ModelScaleFactor(ModelScaleFactor)	Установить свойство

ModelPerformanceLevel – Редактор моделей: Качество сглаживания

Интерфейс...

Тип данных: из перечисления ksModelPerformanceLevelEnum

Синтаксис Automation:

ModelPerformanceLevel = Object.ModelPerformanceLevel	Получить свойство (*)
Object.ModelPerformanceLevel = ModelPerformanceLevel	Установить свойство (*)
ModelPerformanceLevel = Object.GetModelPerformanceLevel()	Получить свойство (**)
Object.SetModelPerformanceLevel(ModelPerformanceLevel)	Установить свойство (**)

Синтаксис COM:

Object.get_ModelPerformanceLevel(&ModelPerformanceLevel)	Получить свойство
Object.put_ModelPerformanceLevel(ModelPerformanceLevel)	Установить свойство

ModelTransparencyType – Редактор моделей: Прозрачность

Интерфейс...

Тип данных: из перечисления ksModelTransparencyTypeEnum

Синтаксис Automation:

ModelTransparencyType = Object.ModelTransparencyType	Получить свойство (*)
Object.ModelTransparencyType = ModelTransparencyType	Установить свойство (*)
ModelTransparencyType = Object.GetModelTransparencyType()	Получить свойство (**)

Object.SetModelTransparencyType(ModelTransparencyType)	Установить свойство (**)

Синтаксис COM:

Object.get_ModelTransparencyType(&ModelTransparencyType)	Получить свойство
Object.put_ModelTransparencyType(ModelTransparencyType)	Установить свойство

EnableAddFilesToRecentList – Файлы: Добавлять открываемые файлы в список недавних документов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EnableAddFilesToRecentList = Object.EnableAddFilesToRecentList	Получить свойство (*)
Object.EnableAddFilesToRecentList = EnableAddFilesToRecentList	Установить свойство (*)
EnableAddFilesToRecentList = Object.GetEnableAddFilesToRecentList()	Получить свойство (**)
Object.SetEnableAddFilesToRecentList(EnableAddFilesToRecentList)	Установить свойство (**)

Синтаксис COM:

Object.get_EnableAddFilesToRecentList(&EnableAddFilesToRecentList)	Получить свойство
Object.put_EnableAddFilesToRecentList(EnableAddFilesToRecentList)	Установить свойство

FilesAutoSaveSwitchOn – Автоматическое сохранение файлов документов

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1070_127_6_2_Ispolqzovanie_fajl.htm

Тип данных: BOOL

Значение свойства:

TRUE	- автосохранение включено,
FALSE	- автосохранение выключено.

Синтаксис Automation:

FilesAutoSaveSwitchOn = iObject.FilesAutoSaveSwitchOn	=	Получить свойство (*)
iObject.FilesAutoSaveSwitchOn = FilesAutoSaveSwitchOn	=	Установить свойство (*)
FilesAutoSaveSwitchOn = iObject.GetFilesAutoSaveSwitchOn()	=	Получить свойство (**)
iObject.SetFilesAutoSaveSwitchOn (FilesAutoSaveSwitchOn)		Установить свойство (**)

Синтаксис COM:

iObject->get_FilesAutoSaveSwitchOn (&FilesAutoSaveSwitchOn)	Получить свойство,
iObject->put_FilesAutoSaveSwitchOn (FilesAutoSaveSwitchOn)	Установить свойство.

FilesBackupPrevCopySwitchOn – Автоматическое сохранение предыдущей копии файла документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1069_127_6_1_Ispolqzovanie_reze.htm

Тип данных: BOOL

Значение свойства:

TRUE	- автосохранение включено,
FALSE	- автосохранение выключено.

Синтаксис Automation:

FilesBackupPrevCopySwitchOn = iObject.FilesBackupPrevCopySwitchOn	=	Получить свойство (*)
iObject.FilesBackupPrevCopySwitchOn = FilesBackupPrevCopySwitchOn		Установить свойство (*)

FilesBackupPrevCopySwitchOn = iObject.GetFilesBackupPrevCopySwitchOn()		Получить свойство (**)
iObject.SetFilesBackupPrevCopySwitchOn (FilesAutoSaveSwitchOn)		Установить\ свойство (**)

Синтаксис COM:

iObject->get_FilesAutoSaveSwitchOn (&FilesBackupPrevCopySwitchOn)	Получить свойство,
iObject->put_FilesBackupPrevCopySwitchOn (&FilesBackupPrevCopySwitchOn)	Установить свойство.

ModelEditColor – Редактор моделей: Цвета редактирования объектов

Интерфейс...

Тип данных: Long

Синтаксис Automation:

ModelEditColor = Object.ModelEditColor(ColorType)		Получить свойство (*)
Object.ModelEditColor(ColorType) = ModelEditColor		Установить свойство (*)
ModelEditColor = Object.GetModelEditColor(ColorType)		Получить свойство (**)
Object.SetModelEditColor(ColorType, ModelEditColor)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelEditColor(ColorType, &ModelEditColor)	Получить свойство,
Object.put_ModelEditColor(ColorType, ModelEditColor)	Установить свойство.

Входные параметры:

ksEditColorTypeEnum	- ColorType - тип цвета редактирования.

ModelFillChooseFace – Редактор моделей: Закрашивать грани при подсвечивании

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ModelFillChooseFace = Получить свойство (*)
Object.ModelFillChooseFace
Object.ModelFillChooseFace = Установить свойство (*)
ModelFillChooseFace
ModelFillChooseFace = Получить свойство (**)
Object.GetModelFillChooseFace()
Object.SetModelFillChooseFace(Установить свойство (**)
ModelFillChooseFace)

Синтаксис COM:

Object.get_ModelFillChooseFace(Получить свойство,
&ModelFillChooseFace)
Object.put_ModelFillChooseFace(Установить свойство.
ModelFillChooseFace)

ModelInverseInDynamicSelect – Редактор моделей: Инверсия при динамическом подсвечивании

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ModelInverseInDynamicSelect = Получить свойство (*)
Object.ModelInverseInDynamicSelect
Object.ModelInverseInDynamicSelect = Установить свойство (*)
ModelInverseInDynamicSelect
ModelInverseInDynamicSelect = Получить свойство (**)
Object.GetModelInverseInDynamicSele
ct()
Object.SetModelInverseInDynamicSele Установить свойство (**)
ct(ModelInverseInDynamicSelect)

Синтаксис COM:

Object.get_ModelInverseInDynamicSel Получить свойство,
ect(&ModelInverseInDynamicSelect)
Object.put_ModelInverseInDynamicSel Установить свойство.
ect(ModelInverseInDynamicSelect)

Language – Марка. Текст префикса

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Language = Object.Language		Получить свойство (*)
Language = Object.GetLanguage()		Получить свойство (**)

Синтаксис COM:

Object.get_Language(&Language)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Список возможных значений свойства см. https://msdn.microsoft.com/ru-ru/library/ee825488(v=cs.20).aspx

В зависимости от текущего языка выбирается словарь (текстовый UNICOD-ный документ с расширением *.dic)*. Имя файла для приложений формируется как имя файла приложения - подчеркивание - имя текущего языка.

Пример:

komlib_ru_RU.dic - словарь для русского языка
komlib_ru_EN.dic - словарь для английского языка.

ModelLocalCSCreateInAbsoluteCS – Создавать ЛСК только в абсолютной СК

Интерфейс...

Тип данных: BOOL

Значение свойства:

TRUE	- новая локальная система координат будет создана в абсолютной СК,
FALSE	- новая локальная система координат будет создана в текущей СК.

Синтаксис Automation:

ModelLocalCSCreateInAbsoluteCS = iObject.ModelLocalCSCreateInAbsoluteCS		Получить свойство (*)
iObject.ModelLocalCSCreateInAbsoluteCS = ModelLocalCSCreateInAbsoluteCS		Установить свойство (*)
ModelLocalCSCreateInAbsoluteCS = iObject.GetModelLocalCSCreateInAbsoluteCS()		Получить свойство (**)

iObject.SetModelLocalCSCreateInAbsoluteCS (ModelLocalCSCreateInAbsoluteCS)	Установить свойство (**)

Синтаксис COM:

iObject->get_ModelLocalCSCreateInAbsoluteCS (&ModelLocalCSCreateInAbsoluteCS)	Получить свойство,
iObject->put_ModelLocalCSCreateInAbsoluteCS (&ModelLocalCSCreateInAbsoluteCS)	Установить свойство.

ModelLocalCSSetActive – При создании ЛСК назначать ее текущей СК

Интерфейс...

Тип данных: BOOL

Значение свойства:

TRUE	- новая локальная система координат станет текущей,
FALSE	- текущей останется старая СК.

Синтаксис Automation:

ModelLocalCSSetActive = iObject.ModelLocalCSSetActive	=	Получить свойство (*)
iObject.ModelLocalCSSetActive = ModelLocalCSSetActive	=	Установить свойство (*)
ModelLocalCSSetActive = iObject.GetModelLocalCSSetActive()	=	Получить свойство (**)
iObject.SetModelLocalCSSetActive(ModelLocalCSSetActive)		Установить свойство (**)

Синтаксис COM:

iObject->get_ModelLocalCSSetActive (&ModelLocalCSSetActive)	Получить свойство,
iObject->put_ModelLocalCSSetActive (&ModelLocalCSSetActive)	Установить свойство.

ModelSmoothMotion – Редактор моделей: Изменение ориентации: Плавность

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ModelSmoothMotion                =  Получить свойство (* )
Object.ModelSmoothMotion
Object.ModelSmoothMotion         =  Установить свойство (* )
ModelSmoothMotion
ModelSmoothMotion                =  Получить свойство (**)
Object.GetModelSmoothMotion()
Object.SetModelSmoothMotion(        Установить свойство (**)
ModelSmoothMotion )
```

Синтаксис COM:

```
Object.get_ModelSmoothMotion(       Получить свойство,
&ModelSmoothMotion )
Object.put_ModelSmoothMotion(       Установить свойство.
ModelSmoothMotion )
```

Примечание:

Свойство позволяет включить или отключить плавность поворота модели при редактировании.

ModelUseOpenGLSearch – Редактор моделей: При указании использовать OpenGL-поиск

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ModelUseOpenGLSearch             =  Получить свойство (* )
Object.ModelUseOpenGLSearch
Object.ModelUseOpenGLSearch      =  Установить свойство (* )
ModelUseOpenGLSearch
ModelUseOpenGLSearch             =  Получить свойство (**)
Object.GetModelUseOpenGLSearch()
Object.SetModelUseOpenGLSearch(     Установить свойство (**)
ModelUseOpenGLSearch )
```

Синтаксис COM:

```
Object.get_ModelUseOpenGLSearch(    Получить свойство,
&ModelUseOpenGLSearch )
Object.put_ModelUseOpenGLSearch(    Установить свойство.
ModelUseOpenGLSearch )
```

ModelUsePartColorForEdit – Редактор моделей: Собственные цвета компонентов при контекстном редактировании

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ModelUsePartColorForEdit = Object.ModelUsePartColorForEdit	=	Получить свойство (*)
Object.ModelUsePartColorForEdit = ModelUsePartColorForEdit	=	Установить свойство (*)
ModelUsePartColorForEdit = Object.GetModelUsePartColorForEdit()	=	Получить свойство (**)
Object.SetModelUsePartColorForEdit(ModelUsePartColorForEdit)		Установить свойство (**)

Синтаксис COM:

Object.get_ModelUsePartColorForEdit(&ModelUsePartColorForEdit)	Получить свойство,
Object.put_ModelUsePartColorForEdit(ModelUsePartColorForEdit)	Установить свойство.

NewDocumentSettings – Настройки новых документов

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

NewDocumentSettings = Object.NewDocumentSettings(SettingsType)	=	Получить свойство(*)
NewDocumentSettings = Object.GetNewDocumentSettings(SettingsType)	=	Получить свойство (**)

Синтаксис COM:

Object.get_NewDocumentSettings(SettingsType, &NewDocumentSettings)	Получить свойство

Параметры:

ksNewDocumentSettingsTypeEnum SettingsType - тип настроек.

Примечание:

Свойство доступно только для чтения.

ObjectsFilter3D – Способ фильтрации 3D объектов

Интерфейс...

Тип данных: BOOL.

Синтаксис Automation:

ObjectsFilter3D(Type) = Получить свойство (*) Object.ObjectsFilter3D
Object.ObjectsFilter3D(Type) = Установить свойство (*) ObjectsFilter3D
ObjectsFilter3D = Получить свойство (**) Object.GetObjectsFilter3D(Type)
Object.SetObjectsFilter3D(Type, Установить свойство (**) ObjectsFilter3D)

Синтаксис COM:

Object.get_ObjectsFilter3D(Type, Получить свойство, &ObjectsFilter3D)
Object.put_ObjectsFilter3D(Type, Установить свойство. ObjectsFilter3D)

Входные параметры:

Type - Способ фильтрации 3D объектов ksObjectsFilter3DEnum

Возвращаемое значение:

Признак фильтрации для данного способа.

ReportStyleListSettings – Настройка списка подключенных библиотек стилей отчета

Интерфейс...

Тип данных: указатель на интерфейс ILibArraySettings

Синтаксис Automation:

ReportStyleListSettings = Получить свойство (*) Object.ReportStyleListSettings

ReportStyleListSettings = Получить свойство (**)
Object.GetReportStyleListSettings()

Синтаксис COM:

Object.get_ReportStyleListSettings(Получить свойство
&ReportStyleListSettings)

Примечание:

Свойство доступно только для чтения.

StandardsThreadsListSettings – Настройка списка подключаемых стандартов резьб

Интерфейс...

Тип данных: указатель на интерфейс ILibArraySettings

Синтаксис Automation:

StandardsThreadsListSettings = Получить свойство(*)
Object.StandardsThreadsListSettings(
SettingsType)
StandardsThreadsListSettings = Получить свойство (**)
Object.GetStandardsThreadsListSetting
s(SettingsType)

Синтаксис COM:

Object.get_StandardsThreadsListSettin Получить свойство
gs(&StandardsThreadsListSettings)

Примечание:

Свойство доступно только для чтения.

Theme – Общее: Управление системой: Выполнять автосохранение перед обработкой

Интерфейс...

Тип данных: значение из перечисления ksThemeEnum

Синтаксис Automation:

Theme = Object.Theme Получить свойство(*)
ThreadPattern = Theme = Получить свойство (**)
Object.GetTheme()

Синтаксис COM:

Object.get_Theme(&Theme) Получить свойство

Примечание:

Свойство доступно только для чтения.

Версия: КОМПАС v18

ThreadPattern – Параметры стандартной резьбы

Интерфейс...

Тип данных: Указатель на интерфейс IThreadPattern

Синтаксис Automation:

ThreadPattern = Object.ThreadPattern(ThreadTableName) Получить свойство(*)

ThreadPattern = Object.GetThreadPattern(ThreadTableName) Получить свойство (**)

Синтаксис COM:

Object.get_ThreadPattern(ThreadTableName, &ThreadPattern) Получить свойство

Примечание:

Свойство доступно только для чтения.

Работа со стилями

Интерфейс IStylesManager

Менеджер стилей.

Иерархия:

IDispatch

IStylesManager

Примечание:

Интерфейс является дополнительным для интерфейсов:

- IApplication - позволяет получать системные стили.
- IKompasDocument - позволяет получать\создавать пользовательские стили.

Интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

KOMPAS v19

IStylesManager– свойства

CurvesStyles – Стили линий

Интерфейс...

Тип данных: Указатель на интерфейс IStyles.

Синтаксис Automation:

```
CurvesStyles = Object.CurvesStyles      Получить свойство(* )
CurvesStyles                        =   Получить свойство (**)
Object.GetCurvesStyles()
```

Синтаксис COM:

```
Object.get_CurvesStyles(                Получить свойство
&CurvesStyles )
```

Примечание:

Свойство доступно только для чтения.

Свойство не работает в текстовом документе и спецификации.

HatchsStyles – Стили штриховок

Интерфейс...

Тип данных: Указатель на интерфейс IStyles.

Синтаксис Automation:

```
HatchsStyles = Object.HatchsStyles      Получить свойство(* )
HatchsStyles                        =   Получить свойство (**)
Object.GetHatchsStyles()
```

Синтаксис COM:

```
Object.get_HatchsStyles(                Получить свойство
&HatchsStyles )
```

Примечание:

Свойство доступно только для чтения.

Свойство не работает в текстовом документе и спецификации.

TextsStyles – Стили текстов

Интерфейс...

Тип данных: Указатель на интерфейс IStyles.

Синтаксис Automation:

```
TextsStyles = Object.TextsStyles          Получить свойство(* )
TextsStyles = Object.GetTextsStyles()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_TextsStyles( &TextsStyles )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Интерфейс IStyles

Интерфейс коллекции стилей.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IStyles

Примечание:

Интерфейс можно получить с помощью метода IApplication::LibraryStyles или с помощью методов менеджера стилей IStylesManager.

KOMPAS v19

IStyles – свойства

Item – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IStyle

Синтаксис Automation

```
Item = Object.Item(Index)          Получить свойство (* )
Item = Object.GetItem(Index)       Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Item( Index, &Item )          Получить свойство
```

Входные параметры:

```
VARIANT Index          - индекс или имя элемента.
```

Примечание:

Свойство доступно только для чтения.

StyleByApiId – Возвращает элемент, заданный по идентификатору стиля API

Интерфейс...

Тип данных: Указатель на интерфейс IStyle

Синтаксис Automation:

StyleByApiId Object.StyleByApiId(Id)	=	Получить свойство (*)
StyleByApiId Object.GetStyleByApiId(Id)	=	Получить свойство (**)

Синтаксис COM:

Object.get_StyleByApiId(Id, &StyleByApiId)	Получить свойство

Входные параметры:

long Id	- идентификатор стиля в API.

Примечание:

Свойство доступно только для чтения.

IStyles – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IStyle * * Result);

Возвращаемое значение:

- Указатель на интерфейс IStyle

Примечания:

Метод позволяет создать новый интерфейс стиля.

AddStyleFromLibrary – Добавить стиль из библиотеки

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddStyleFromLibrary (BSTR LibraryPath, long Id, BOOL External :

Синтаксис COM:

HRESULT AddStyleFromLibrary(BSTR LibraryPath, long Id, BOOL External, IStyle * * Result);

Возвращаемое значение:

- Указатель на интерфейс IStyle

Входные параметры:

LibraryPath	- имя файла библиотеки стилей,
Id	- номер стиля в библиотеке,
External	- признак внешнего стиля.

Copy – Копировать стиль

Интерфейс...

Синтаксис Automation:

LPDISPATCH Copy (IStyle * Style);

Синтаксис COM:

HRESULT Copy(IStyle * Style, IStyle ** Result);

Возвращаемое значение:

- Указатель на интерфейс IStyle

Входные параметры:

Style	- стиль.

FindStyleFromLibrary – Возвращает внешний стиль, если он был добавлен из библиотеки

Интерфейс...

Синтаксис Automation:

LPDISPATCH FindStyleFromLibrary(BSTR LibraryPath, long LibraryStyleId);

Синтаксис COM:

HRESULT FindStyleFromLibrary(BSTR LibraryPath, long LibraryStyleId, IStyle * * Result);

Возвращаемое значение:

Указатель на интерфейс IStyle .

Входные параметры:

LibraryPath	- имя файла библиотеки стилей,
LibraryStyleId	уникальный номер стиля.

Интерфейс IStyle

Интерфейс стиля.

Иерархия:

IDispatch

IKompasAPIObject

IStyle

Интерфейс можно получить с помощью методов коллекции стилей IStyles.

KOMPAS v19

IStyle – свойства

ApiStyleId – Идентификатор стиля в API

Интерфейс...

Тип данных: long

Синтаксис Automation:

ApiStyleId = Object.ApiStyleId		Получить свойство (*)
ApiStyleId	=	Получить свойство (**)
Object.GetApiStyleId()		

Синтаксис COM:

Object.get_ApiStyleId(	Получить свойство
&ApiStyleId)	

Примечание:

Свойство доступно только для чтения.

DisplayStyleId – Отображаемый идентификатор стиля

Интерфейс...

Тип данных: long

Синтаксис Automation:

DisplayStyleId = Object.DisplayStyleId		Получить свойство (*)
Object.DisplayStyleId = DisplayStyleId		Установить свойство (*)
DisplayStyleId = Object.GetDisplayStyleId()	=	Получить свойство (**)
Object.SetDisplayStyleId(DisplayStyleId)		Установить свойство (**)

Синтаксис COM:

Object.get_DisplayStyleId(&DisplayStyleId)	Получить свойство,
Object.put_DisplayStyleId(DisplayStyleId)	Установить свойство.

IsExternalStyle – Внешний стиль, загружаемый из библиотеки стилей

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsExternalStyle = Object.IsExternalStyle		Получить свойство (*)
IsExternalStyle = Object.GetIsExternalStyle	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsExternalStyle(&IsExternalStyle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LibraryPath – Путь к файлу библиотеки стилей

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LibraryPath = Object.LibraryPath		Получить свойство (*)
LibraryPath Object.GetLibraryPath()	=	Получить свойство (**)

Синтаксис COM:

Object.get_LibraryPath(&LibraryPath)	Получить свойство

Примечание:

Свойство доступно только для чтения.

LibraryStyleId – Идентификатор стиля в библиотеке стилей

Интерфейс...

Тип данных: long

Синтаксис Automation:

LibraryStyleId Object.LibraryStyleId	=	Получить свойство (*)
LibraryStyleId Object.GetLibraryStyleId()	=	Получить свойство (**)

Синтаксис COM:

Object.get_LibraryStyleId(&LibraryStyleId)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Name – Имя стиля

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName(Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство,
Object.put_Name(Name)	Установить свойство.

IStyle – методы

Delete – Удалить стиль

Интерфейс...

Синтаксис Automation :

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс ICurveStyle

Интерфейс стиля линии.

Иерархия:

IDispatch

IKompasAPIObject

IStyle

ICurveStyle

Примечание:

Интерфейс можно получить с помощью методов коллекции стилей линий, полученной с помощью

- метода IStylesManager::CurvesStyles,
- коллекции стилей библиотеки стилей линий (*.lcs) IApplication::LibraryStyles.

Версия: KOMPAS v19

ICurveStyle – свойства

Color – Цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Color = Object.Color                    Получить свойство (* )
Object.Color = Color                    Установить свойство (**)
Color = Object.GetColor()               Получить свойство (**)
Object.SetColor( Color )                Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Color( &Color )              Получить свойство
Object.put_Color(  Color )              Установить свойство
```

CurvePenType – Способ задания параметров пера

Интерфейс...

Тип данных: из перечисления ksCurvePenTypeEnum.

Синтаксис Automation:

```
CurvePenType = Object.CurvePenType                  Получить свойство (* )
Object.CurvePenType = CurvePenType                  Установить свойство (**)
CurvePenType = Object.GetCurvePenType()             Получить свойство (**)
Object.SetCurvePenType( CurvePenType )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_CurvePenType( &CurvePenType )            Получить свойство
Object.put_CurvePenType(  CurvePenType )            Установить свойство
```

CurveStyleType – Тип стиля кривой

Интерфейс...

Тип данных: из перечисления ksCurveStyleTypeEnum

Синтаксис Automation:

CurveStyleType = Object.CurveStyleType	Получить свойство (*)
Object.CurveStyleType = CurveStyleType	Установить свойство (**)
CurveStyleType = Object.GetCurveStyleType()	Получить свойство (**)
Object.SetCurveStyleType(CurveStyleType)	Установить свойство (**)

Синтаксис COM:

Object.get_CurveStyleType(&CurveStyleType)	Получить свойство
Object.put_CurveStyleType(CurveStyleType)	Установить свойство

ForHatch – Является границей для штриховки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ForHatch = Object.ForHatch	Получить свойство (*)
Object.ForHatch = ForHatch	Установить свойство (**)
ForHatch = Object.GetForHatch()	Получить свойство (**)
Object.SetForHatch(ForHatch)	Установить свойство (**)

Синтаксис COM:

Object.get_ForHatch(&ForHatch)	Получить свойство
Object.put_ForHatch(ForHatch)	Установить свойство

IgnoreFragmentStyle – Игнорировать стиль линий фрагментов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IgnoreFragmentStyle = Object.IgnoreFragmentStyle	=	Получить свойство (*)
Object.IgnoreFragmentStyle = IgnoreFragmentStyle	=	Установить свойство (**)
IgnoreFragmentStyle = Object.GetIgnoreFragmentStyle()	=	Получить свойство (**)
Object.SetIgnoreFragmentStyle(IgnoreFragmentStyle)		Установить свойство (**)

Синтаксис COM:

Object.get_IgnoreFragmentStyle(&IgnoreFragmentStyle)	Получить свойство
Object.put_IgnoreFragmentStyle(IgnoreFragmentStyle)	Установить свойство

PatternsCount – Количество параметров пары штрих–промежуток

Интерфейс...

Тип данных: long

Синтаксис Automation:

PatternsCount = Object.PatternsCount	=	Получить свойство(*)
PatternsCount = Object.GetPatternsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_PatternsCount(&PatternsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PatternFragmentDx – Величина смещения фрагмента по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

PatternFragmentDx = Object.PatternFragmentDx(PatternIndex)	=	Получить свойство (*)
Object.PatternFragmentDx(PatternIndex) = PatternFragmentDx	=	Установить свойство (**)
PatternFragmentDx = Object.GetPatternFragmentDx(PatternIndex))	=	Получить свойство (**)
Object.SetPatternFragmentDx(PatternIndex, PatternFragmentDx)		Установить свойство (**)

Синтаксис COM:

Object.get_PatternFragmentDx(PatternIndex, &PatternFragmentDx) Получить свойство
Object.put_PatternFragmentDx(PatternIndex, PatternFragmentDx) Установить свойство

Входные параметры:

long PatternIndex - индекс параметров.

PatternFragmentDy – Величина смещения фрагмента по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

PatternFragmentDy = Object.PatternFragmentDy(PatternIndex) Получить свойство (*)
Object.PatternFragmentDy(PatternIndex) = PatternFragmentDy Установить свойство (**)
PatternFragmentDy = Object.GetPatternFragmentDy(PatternIndex)) Получить свойство (**)
Object.SetPatternFragmentDy(PatternIndex, PatternFragmentDy) Установить свойство (**)

Синтаксис COM:

Object.get_PatternFragmentDy(PatternIndex, &PatternFragmentDy) Получить свойство
Object.put_PatternFragmentDy(PatternIndex, PatternFragmentDy) Установить свойство

Входные параметры:

long PatternIndex - индекс параметров.

PatternFragmentPoligon – Массив точек полигона

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

PatternFragmentPoligon = Object.PatternFragmentPoligon(PatternIndex, PoligonIndex) — Получить свойство (*)

Object.PatternFragmentPoligon(PatternIndex, PoligonIndex) = PatternFragmentPoligon — Установить свойство (**)

PatternFragmentPoligon = Object.GetPatternFragmentPoligon(PatternIndex, PoligonIndex) — Получить свойство (**)

Object.SetPatternFragmentPoligon(PatternIndex, PoligonIndex, PatternFragmentPoligon) — Установить свойство (**)

Синтаксис COM:

Object.get_PatternFragmentPoligon(PatternIndex, PoligonIndex, &PatternFragmentPoligon) — Получить свойство

Object.put_PatternFragmentPoligon(PatternIndex, PoligonIndex, PatternFragmentPoligon) — Установить свойство

Входные параметры:

long PatternIndex	- индекс параметров,
long PoligonIndex	- индекс полигона.

PatternFragmentFilletsPoligon – Массив точек полигона границ заливок

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

PatternFragmentFilletsPoligon = Object.PatternFragmentFilletsPoligon(PatternIndex, PoligonIndex) — Получить свойство (*)

Object.PatternFragmentFilletsPoligon(PatternIndex, PoligonIndex) = PatternFragmentFilletsPoligon — Установить свойство (**)

PatternFragmentFilletsPoligon = Object.GetPatternFragmentFilletsPoligon(PatternIndex, PoligonIndex)	Получить свойство (**)
Object.SetPatternFragmentFilletsPoligon(PatternIndex, PoligonIndex, PatternFragmentFilletsPoligon)	Установить свойство (**)

Синтаксис COM:

Object.get_PatternFragmentFilletsPoligon(PatternIndex, PoligonIndex, &PatternFragmentFilletsPoligon)	Получить свойство
Object.put_PatternFragmentFilletsPoligon(PatternIndex, PoligonIndex, PatternFragmentFilletsPoligon)	Установить свойство

Входные параметры:

long PatternIndex	- индекс параметров,
long PoligonIndex	- индекс полигона.

PatternFragmentPoligonsCount – Количество ломаных во фрагменте

Интерфейс...

Тип данных: long

Синтаксис Automation:

PatternFragmentPoligonsCount = Object.PatternFragmentPoligonsCount(PatternIndex)	Получить свойство(*)
PatternFragmentPoligonsCount = Object.GetPatternFragmentPoligonsCount(PatternIndex)	Получить свойство (**)

Синтаксис COM:

Object.get_PatternFragmentPoligonsCount(PatternIndex, &PatternFragmentPoligonsCount)	Получить свойство

Входные параметры:

long PatternIndex - индекс параметров.

Примечание:

Свойство доступно только для чтения.

PatternInvisibleSegmentLenght – Длина невидимого участка

Интерфейс...

Тип данных: double

Синтаксис Automation:

PatternInvisibleSegmentLenght = Object.PatternInvisibleSegmentLenght(PatternIndex)	=	Получить свойство (*)
Object.PatternInvisibleSegmentLenght(PatternIndex) = PatternInvisibleSegmentLenght		Установить свойство (**)
PatternInvisibleSegmentLenght = Object.GetPatternInvisibleSegmentLenght(PatternIndex)	=	Получить свойство (**)
Object.SetPatternInvisibleSegmentLenght(PatternIndex, PatternInvisibleSegmentLenght)		Установить свойство (**)

Синтаксис COM:

Object.get_PatternInvisibleSegmentLenght(PatternIndex, &PatternInvisibleSegmentLenght)	Получить свойство
Object.put_PatternInvisibleSegmentLenght(PatternIndex, PatternInvisibleSegmentLenght)	Установить свойство

Входные параметры:

long PatternIndex	- индекс параметров.

PatternVisibleSegmentLenght – Длина видимого участка

Интерфейс...

Тип данных: double

Синтаксис Automation:

PatternVisibleSegmentLenght = Object.PatternVisibleSegmentLenght(PatternIndex)	=	Получить свойство (*)
Object.PatternVisibleSegmentLenght(PatternIndex) = PatternVisibleSegmentLenght		Установить свойство (**)
PatternVisibleSegmentLenght = Object.GetPatternVisibleSegmentLenght(PatternIndex)	=	Получить свойство (**)
Object.SetPatternVisibleSegmentLenght(PatternIndex, PatternVisibleSegmentLenght)		Установить свойство (**)

Синтаксис COM:

Object.get_PatternVisibleSegmentLenght(PatternIndex, &PatternVisibleSegmentLenght)	Получить свойство
Object.put_PatternVisibleSegmentLenght(PatternIndex, PatternVisibleSegmentLenght)	Установить свойство

Входные параметры:

long PatternIndex	- индекс параметров.

PaperWidth – Толщина на бумаге (мм)

Интерфейс...

Тип данных: double

Синтаксис Automation:

PaperWidth = Object.PaperWidth	Получить свойство (*)
Object.PaperWidth = PaperWidth	Установить свойство (**)
PaperWidth = Object.GetPaperWidth()	Получить свойство (**)
Object.SetPaperWidth(PaperWidth)	Установить свойство (**)

Синтаксис COM:

Object.get_PaperWidth(&PaperWidth)	Получить свойство
Object.put_PaperWidth(PaperWidth)	Установить свойство

ScreenWidth – Толщина на экране (пикс)

Интерфейс...

Тип данных: long

Синтаксис Automation:

ScreenWidth = Object.ScreenWidth	Получить свойство (*)
Object.ScreenWidth = ScreenWidth	Установить свойство (**)
ScreenWidth = Object.GetScreenWidth()	Получить свойство (**)
Object.SetScreenWidth(ScreenWidth)	Установить свойство (**)

Синтаксис COM:

Object.get_ScreenWidth(&ScreenWidth)	Получить свойство
Object.put_ScreenWidth(ScreenWidth)	Установить свойство

SmartParts – Кривая всегда заканчивается штрихом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SmartParts = Object.SmartParts	Получить свойство (*)
Object.SmartParts = SmartParts	Установить свойство (**)
SmartParts = Object.GetSmartParts()	Получить свойство (**)
Object.SetSmartParts(SmartParts)	Установить свойство (**)

Синтаксис COM:

Object.get_SmartParts(&SmartParts)	Получить свойство
Object.put_SmartParts(SmartParts)	Установить свойство

ICurveStyle – методы

AddPattern – Добавить параметры пары штрих-промежуток

Интерфейс...

Синтаксис Automation:

BOOL AddPattern(double VisibleSegmentLenght, double invisibleSegmentLenght);

Синтаксис COM:

HRESULT AddPattern(double VisibleSegmentLenght, double invisibleSegmentLenght, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

VisibleSegmentLenght - длина видимого участка,
invisibleSegmentLenght - длина невидимого участка.

AddPatternFragmentPoligon – Добавить полигон точек

Интерфейс...

Синтаксис Automation:

BOOL AddPatternFragmentPoligon(long PatternIndex, VARIANT Points);

Синтаксис COM:

HRESULT AddPatternFragmentPoligon(long PatternIndex, VARIANT Points, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс параметров,
Points - безопасный массив 2D точек SAFEARRAY VT_ARRAY | VT_R8.

AddPatternFragmentFilletsPoligon – Добавить заливку, ограниченную ломаной, заданной полигоном точек

Интерфейс...

Синтаксис Automation:

BOOL AddPatternFragmentFilletsPoligon(long PatternIndex, VARIANT Points);

Синтаксис COM:

HRESULT AddPatternFragmentFilletsPoligon(long PatternIndex, VARIANT Points, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс параметров,
Points - безопасный массив 2D точек SAFEARRAY VT_ARRAY | VT_R8, задающих полигон границы заливки.

ClearPatterns – Очистить параметры участков штрих–промежуток

Интерфейс...

Синтаксис Automation :

BOOL ClearPatterns();

Синтаксис COM :

HRESULT ClearPatterns(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

ClearPatternFragmentPoligons – Очистить список полигонов

Интерфейс...

Синтаксис Automation:

BOOL ClearPatternFragmentPoligons(long PatternIndex);

Синтаксис COM:

HRESULT ClearPatternFragmentPoligons(long PatternIndex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс параметров.

ClearPatternFragmentFilletsPoligons – Удалить все заливки

Интерфейс...

Синтаксис Automation:

BOOL ClearPatternFragmentFilletsPoligons(long PatternIndex);

Синтаксис COM:

HRESULT ClearPatternFragmentFilletsPoligons(long PatternIndex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс образца.

DeletePattern – Удалить параметры пары штрих–промежуток

Интерфейс...

Синтаксис Automation:

BOOL DeletePattern(long PatternIndex);

Синтаксис COM:

HRESULT DeletePattern(long PatternIndex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс параметров.

DeletePatternFragmentPoligon – Удалить полигон

Интерфейс...

Синтаксис Automation:

BOOL DeletePatternFragmentPoligon(long PatternIndex, long PoligonIndex);

Синтаксис COM:

HRESULT DeletePatternFragmentPoligon(long PatternIndex, long PoligonIndex, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

PatternIndex - индекс параметров,
long PoligonIndex - индекс полигона.

DeletePatternFragmentFilletsPoligon – Удалить заливку

Интерфейс...

Синтаксис Automation:

BOOL DeletePatternFragmentFilletsPoligon(long PatternIndex, long PoligonIndex);

Синтаксис COM:

HRESULT DeletePatternFragmentFilletsPoligon(long PatternIndex, long PoligonIndex, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

PatternIndex	- индекс образца,
long PoligonIndex	- индекс полигона.

LoadPatternFragment – Загрузить фрагмент из файла

Интерфейс...

Синтаксис Automation:

BOOL LoadPatternFragment(long PatternIndex, BSTR FileName);

Синтаксис COM:

HRESULT LoadPatternFragment(long PatternIndex, BSTR FileName, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

PatternIndex	- индекс образца,
FileNamex	- имя фрагмента.

SetPatternFragment – Установить фрагмент по геометрии

Интерфейс...

Синтаксис Automation:

BOOL SetPatternFragment(long PatternIndex, VARIANT Geom);

Синтаксис COM:

HRESULT SetPatternFragment(long PatternIndex, VARIANT Geom, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

PatternIndex	- индекс образца,
Geom	- массив объектов SAFEARRAY VT_DISPATCH - VT_ARRAY | VT_DISPATCH.

Интерфейс IHatchStyle

Интерфейс стиля штриховки.

Иерархия:

IDispatch

IKompasAPIObject

IStyle

IHatchStyle

Примечание:

Интерфейс можно получить с помощью методов коллекции стилей штриховок, полученной с помощью:

- метода IStylesManager::HatchsStyles,
- коллекции стилей библиотеки стилей штриховок (*.lhs) IApplication::LibraryStyles.

KOMPAS v19

IHatchStyle – свойства

Angle – Угол наклона

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Object.Angle = Angle	Установить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство
Object.put_Angle(Angle)	Установить свойство

CurvesStyles – Стили линий

Интерфейс...

Тип данных: Указатель на интерфейс IStyles

Синтаксис Automation:

CurvesStyles Object.CurvesStyles	=	Получить свойство (*)
CurvesStyles Object.GetCurvesStyles()	=	Получить свойство (**)

Синтаксис COM:

Object.get_CurvesStyles(&CurvesStyles)	Получить свойство

Примечание:

Свойство доступно только для чтения.

HatchType – Тип штриховки. TRUE – Область FALSE – полоса

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

HatchType = Object.HatchType		Получить свойство (*)
Object.HatchType = HatchType		Установить свойство (*)
HatchType Object.GetHatchType()	=	Получить свойство (**)
Object.SetHatchType(HatchType)		Установить свойство (**)

Синтаксис COM:

Object.get_HatchType(&HatchType)	Получить свойство

Object.put_HatchType(HatchType)	Установить свойство

KeepAngle – Не изменять угол наклона

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

KeepAngle = Object.KeepAngle	Получить свойство (*)
Object.KeepAngle = KeepAngle	Установить свойство (*)
KeepAngle = Object.GetKeepAngle()	Получить свойство (**)
Object.SetKeepAngle(Keep Angle)	Установить свойство (**)

Синтаксис COM:

Object.get_KeepAngle(&KeepAngle)	Получить свойство
Object.put_KeepAngle(KeepAngle)	Установить свойство

KeepScale – Не изменять масштаб

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

KeepScale = Object.KeepScale	Получить свойство (*)
Object.KeepScale = KeepScale	Установить свойство (*)
KeepScale = Object.GetKeepScale()	Получить свойство (**)
Object.SetKeepScale(KeepScale)	Установить свойство (**)

Синтаксис COM:

Object.get_KeepScale(&KeepScale)	Получить свойство
Object.put_KeepScale(KeepScale)	Установить свойство

LineAngle – Добавочный угол поворота относительно угла штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineAngle = Object.LineAngle(Index)	Получить свойство (*)
Object.LineAngle(Index) = LineAngle	Установить свойство (*)
LineAngle = Object.GetLineAngle(Index)	Получить свойство (**)
Object.SetLineAngle(Index, LineAngle)	Установить свойство (**)

Синтаксис COM:

Object.get_LineAngle(Index, &LineAngle)	Получить свойство
Object.put_LineAngle(Index, LineAngle)	Установить свойство

Входные параметры:

long Index	- индекс линии.

LineBeginX – Начальное положение относительно базовой точки штриховки по X

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineBeginX = Object.LineBeginX(Index)	Получить свойство (*)
Object.LineBeginX(Index) = LineBeginX	Установить свойство (*)
LineBeginX = Object.GetLineBeginX(Index)	Получить свойство (**)
Object.SetLineBeginX(Index, LineBeginX)	Установить свойство (**)

Синтаксис COM:

Object.get_LineBeginX(Index, &LineBeginX)		Получить свойство
Object.put_LineBeginX(Index, LineBeginX)		Установить свойство

Входные параметры:

long Index	- индекс линии.

LineBeginY – Начальное положение относительно базовой точки штриховки по Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineBeginY = Object.LineBeginY(Index)		Получить свойство (*)
Object.LineBeginY(Index) = LineBeginY		Установить свойство (*)
LineBeginY = Object.GetLineBeginY(Index)		Получить свойство (**)
Object.SetLineBeginY(Index, LineBeginY)		Установить свойство (**)

Синтаксис COM:

Object.get_LineBeginY(Index, &LineBeginY)		Получить свойство
Object.put_LineBeginY(Index, LineBeginY)		Установить свойство

Входные параметры:

long Index	- индекс линии.

LinesCount – Количество линий

Интерфейс...

Тип данных: long

Синтаксис Automation:

LinesCount = Object.LinesCount	Получить свойство (*)

LinesCount = Получить свойство (**)
Object.GetLinesCount()

Синтаксис COM:

Object.get_LinesCount(Получить свойство
&LinesCount)

Примечание:

Свойство доступно только для чтения.

LineDx – Смещение по направлению линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineDx = Object.LineDx(Index) Получить свойство (*)
Object.LineDx(Index) = LineDx Установить свойство (*)
LineDx = Object.GetLineDx(Получить свойство (**)
Index)
Object.SetLineDx(Index, LineDx Установить свойство (**)
)

Синтаксис COM:

Object.get_LineDx(Index, Получить свойство
&LineDx)
Object.put_LineDx(Index, Установить свойство
LineDx)

Входные параметры:

long Index - индекс линии.

LineDy – Смещение по нормали к линии

Интерфейс...

Тип данных: double

Синтаксис Automation:

LineDy = Object.LineDy(Index) Получить свойство (*)
Object.LineDy(Index) = LineDy Установить свойство (*)
LineDy = Object.GetLineDy(Получить свойство (**)
Index)

Object.SetLineDy(Index, LineDy)	Установить свойство (**)

Синтаксис COM:

Object.get_LineDy(Index, &LineDy)	Получить свойство
Object.put_LineDy(Index, LineDy)	Установить свойство

Входные параметры:

long Index	- индекс линии.

LineStyle – Стиль линии штриховки

Интерфейс...

Тип данных: Указатель на интерфейс ICurveStyle.

Синтаксис Automation:

LineStyle = Object.LineStyle(Index)	Получить свойство (*)
Object.LineStyle(Index) = LineStyle	Установить свойство (*)
LineStyle = Object.GetLineStyle(Index)	Получить свойство (**)
Object.SetLineStyle(Index, LineStyle)	Установить свойство (**)

Синтаксис COM:

Object.get_LineStyle(Index, &LineStyle)	Получить свойство
Object.put_LineStyle(Index, LineStyle)	Установить свойство

Входные параметры:

long Index	- индекс линии.

Scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Scale = Object.Scale              Получить свойство (* )
Object.Scale = Scale              Установить свойство (* )
Scale = Object.GetScale()         Получить свойство (**)
Object.SetScale(Scale )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Scale( &Scale )        Получить свойство
Object.put_Scale(  Scale )        Установить свойство
```

Width – Ширина полосы

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Width = Object.Width              Получить свойство (* )
Object.Width = Width              Установить свойство (* )
Width = Object.GetWidth()         Получить свойство (**)
Object.SetWidth(Width)            Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Width( &Width )        Получить свойство
Object.put_Width(  Width )        Установить свойство
```

IHatchStyle –методы

AddLine – Добавить линию

Интерфейс...

Синтаксис Automation:

BOOL AddLine(ICurveStyle * Style, double X, double Y, double Angle, double DX, double DY);

Синтаксис COM:

HRESULT AddLine(ICurveStyle * Style, double X, double Y, double Angle, double DX, double DY, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,

Входные параметры:

Style - пользовательский стиль кривой ICurveStyle,
X - начальное положение по x относительно базовой точки,
Y - начальное положение по y относительно базовой точки,
Angle - добавочный угол поворота относительно угла штриховки,
DX - смещение по направлению линии,
DY - смещение по нормали к линии.

ClearLines – Удалить все параметры линий

Интерфейс...

Синтаксис Automation :

BOOL ClearLines();

Синтаксис COM :

HRESULT ClearLines(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,

DeleteLine – Удалить параметры линии по индексу

Интерфейс...

Синтаксис Automation:

BOOL DeleteLine(long Index);

Синтаксис COM:

HRESULT DeleteLine(long Index, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,

Входные параметры:

Index - индекс линии.

Работа со спецификацией

Интерфейс IAdditionalBlockSectionTuning

Интерфейс настройки раздела блока дополнительных (вложенных) разделов.

Иерархия:

IKompasAPIObject

IAdditionalBlockSectionTuning

Описание:

Данный интерфейс позволяет получить доступ к настройкам раздела блока дополнительных (вложенных) разделов спецификации. Чтобы использовать в блоке этот раздел, установите свойству Use значение TRUE.

Если настройки взяты у стиля спецификации, то они являются умолчательными для данного стиля, и изменять их нельзя. В этом случае данный интерфейс отражает содержимое настроек стиля в библиотеке стилей на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.

Если настройки взяты у описания спецификации, то они отображают и отслеживают текущее состояние описания спецификации. Эти настройки можно изменять и они могут отличаться от умолчательных настроек стиля спецификации, по которому создано данное описание спецификации.

После изменения настроек (для того чтобы эти изменения реально отобразились в документе) необходимо вызвать метод ISpecificationTuning::Update.

Примечание:

Данный интерфейс можно получить у коллекции IAdditionalBlockSectionTunings.

IAdditionalBlockSectionTuning – свойства

Number – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер раздела блока дополнительных (вложенных) разделов в данном стиле спецификации.
2. Свойство доступно только для чтения.

Use – Использовать этот раздел

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Use = iObject.Use	Получить свойство (*)
iObject.Use = Use	Установить свойство (*)
Use = iObject.GetUse()	Получить свойство (**)
iObject.SetUse (Use)	Установить свойство (**)

Синтаксис COM:

iObject->get_Use (&Use)	Получить свойство
iObject->put_Use (Use)	Установить свойство

Значения свойства:

TRUE	- использовать данный раздел,
FALSE	- раздел использоваться не будет.

Примечание:

Позволяет получить\изменить доступность данного раздела при заполнении спецификации.

Интерфейс IAdditionalBlockStyle

Интерфейс стиля блока дополнительных (вложенных) разделов спецификации.

Иерархия:

IKompasAPIObject

IAdditionalBlockStyle

Описание:

Вложенные разделы располагаются внутри раздела, после всех объектов этого раздела.

Дополнительные разделы располагаются в конце спецификации, после всех ее разделов.

Блок состоит из тех же разделов, что и сама спецификация.

Примечание:

Данный интерфейс можно получить от интерфейса коллекции стилей дополнительных (вложенных) разделов спецификации IAdditionalBlockStyles, IAdditionalBlockStyles::Item.

IAdditionalBlockStyle – свойства

Name – Имя блока дополнительных (вложенных) разделов

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Number – Номер блока дополнительных (вложенных) разделов

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Блоки разделов в спецификации располагаются в порядке возрастания или убывания их номеров Number, в зависимости от значения свойства ISpecificationStyle::SortSectionDown.
2. Свойство доступно только для чтения.

Интерфейс IAdditionalBlockTuning

Интерфейс настройки блока дополнительных (вложенных) разделов..

Иерархия:

IKompasAPIObject

IAdditionalBlockTuning

Описание:

Данный интерфейс позволяет получить доступ к настройкам блока дополнительных (вложенных) разделов спецификации. Если настройки взяты у стиля спецификации, то они являются умолчательными для данного стиля, и изменять их нельзя. В этом случае данный интерфейс отражает содержимое настроек стиля в библиотеке стилей на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.

Если настройки взяты у описания спецификации, то они отображают и отслеживают текущее состояние описания спецификации. Эти настройки можно изменять и они могут отличаться от умолчательных настроек стиля спецификации, по которому создано данное описание спецификации.

После изменения настроек (для того чтобы эти изменения реально отобразились в документе) необходимо вызвать метод ISpecificationTuning::Update.

Примечание:

Данный интерфейс можно получить у коллекции IAdditionalBlockTunings.

IAdditionalBlockTuning – свойства

DocumentName – Имя файла документа, по которому устанавливается имя блока дополнительных (вложенных) разделов

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

DocumentName = iObject.DocumentName	=	Получить свойство (*)
iObject.DocumentName = DocumentName	=	Установить свойство (*)
DocumentName = iObject.GetDocumentName()	=	Получить свойство (**)
iObject.SetDocumentName (DocumentName)		Установить свойство (**)

Синтаксис COM:

iObject->get_DocumentName (&DocumentName)	Получить свойство
iObject->put_DocumentName (DocumentName)	Установить свойство

Примечание:

Позволяет получить\изменить полное имя файла КОМПАС-документа для связи с заголовком блока. Обозначение из этого файла может включаться в заголовок блока. Чтобы

это включение было возможно, имя блока (заданное при настройке стиля спецификации) должно содержать синтаксическую конструкцию вида #XXX# (Количество и тип символов между «решетками» не имеет значения). Вместо этой конструкции в заголовок блока в спецификации будет подставлено обозначение, полученное из документа, путь к которому указан в этом свойстве. Таким образом, указание документа имеет смысл лишь при настройке блоков, заголовки которых содержат конструкцию вида #XXX#. Обычно для связи с заголовком блока выбирается электромонтажный чертеж или таблица соединений.

FirstOnSheet – Блок размещать с новой страницы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FirstOnSheet = iObject.FirstOnSheet	=	Получить свойство (*)
iObject.FirstOnSheet = FirstOnSheet	=	Установить свойство (*)
FirstOnSheet = iObject.GetFirstOnSheet()	=	Получить свойство (**)
iObject.SetFirstOnSheet (FirstOnSheet)		Установить свойство (**)

Синтаксис COM:

iObject->get_FirstOnSheet (&FirstOnSheet)	Получить свойство
iObject->put_FirstOnSheet (FirstOnSheet)	Установить свойство

Значения свойства:

TRUE	- располагать блок на новой странице,
FALSE	- располагать блок сразу за предыдущим.

Примечание:

Позволяет включать и отключать размещение блока на новом листе спецификации.

IndependentPosition – Независимая нумерация позиций

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IndependentPosition = Object.IndependentPosition	=	Получить свойство (*)

Object.IndependentPosition = IndependentPosition	Установить свойство (*)
IndependentPosition = Object.GetIndependentPosition()	Получить свойство (**)
Object.SetIndependentPosition(IndependentPosition)	Установить свойство (**)

Синтаксис COM:

Object.get_IndependentPosition(&IndependentPosition)	Получить свойство
Object.put_IndependentPosition(IndependentPosition)	Установить свойство

MarkOn – Марка. Включена простановка позиций с префиксом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MarkOn = Object.MarkOn	Получить свойство (*)
Object.MarkOn = MarkOn	Установить свойство (*)
MarkOn = Object.GetMarkOn()	Получить свойство (**)
Object.SetMarkOn(MarkOn)	Установить свойство (**)

Синтаксис COM:

Object.get_MarkOn(&MarkOn)	Получить свойство
Object.put_MarkOn(MarkOn)	Установить свойство

Mark – Марка. Текст префикса

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Mark = Object.Mark	Получить свойство (*)
Object.Mark = Mark	Установить свойство (*)
Mark = Object.GetMark()	Получить свойство (**)
Object.SetMark(Mark)	Установить свойство (**)

Синтаксис COM:

Object.get_Mark(&Mark)	Получить свойство
Object.put_Mark(Mark)	Установить свойство

Number – Номер блока дополнительных (вложенных) разделов

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер блока дополнительных (вложенных) разделов в данном стиле спецификации.
2. Свойство доступно только для чтения.

Sections – Разделы блока дополнительных (вложенных) разделов

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockSectionTunings коллекции настроек разделов блока дополнительных (вложенных) разделов

Синтаксис Automation:

Sections = iObject.Sections	Получить свойство (*)
Sections = iObject.GetSections()	Получить свойство (**)

Синтаксис COM:

iObject->get_Sections (&Sections)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Use – Использовать этот блок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Use = iObject.Use	Получить свойство (*)
iObject.Use = Use	Установить свойство (*)
Use = iObject.GetUse()	Получить свойство (**)
iObject.SetUse (Use)	Установить свойство (**)

Синтаксис COM:

iObject->get_Use (&Use)	Получить свойство
iObject->put_Use (Use)	Установить свойство

Значения свойства:

TRUE	- использовать данный блок,
FALSE	- блок использоваться не будет.

Примечание:

Позволяет получить\изменить доступность данного блока при заполнении спецификации.

Интерфейс IAttachedDocument

Интерфейс параметров присоединенного документа к объекту спецификации.

Иерархия:

IKompasAPIObject

IAttachedDocument

Описание:

Объект спецификации можно связать с документом КОМПАС-3D. Эта связь является двусторонней и ассоциативной и позволяет передавать данные об объекте спецификации в подключенный документ, или наоборот, данные из документа в соответствующий ему объект спецификации.

В результате подключения документа в соответствующие колонки объекта спецификации автоматически будут переданы наименование и обозначение из модели или основной надписи чертежа, а также обозначение формата, на котором выполнен подключенный чертеж.

Информация из подключенного документа может передаваться не только в колонки бланка спецификации, но и в дополнительные колонки. Например, в дополнительную колонку объекта может быть автоматически передана масса детали из модели или из соответствующей графы основной надписи.

Примечание:

Данный интерфейс можно получить у коллекции присоединенных документов к объекту спецификации.IAttachedDocuments с помощью свойства IAttachedDocuments::Item и метода IAttachedDocuments::Add.

IAttachedDocument – свойства

Comment – Комментарий для документа источника

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Comment = iObject.Comment	Получить свойство (*)
iObject.Comment = Comment	Установить свойство (*)
Comment = iObject.GetComment()	Получить свойство (**)
iObject.SetComment(Comment)	Установить свойство (**)

Синтаксис COM:

iObject->get_Comment(&Comment)	Получить свойство
iObject->put_Comment(Comment)	Установить свойство

Name – Имя файла документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

iObject->get_Name(&Name)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Transmit – Передавать в изменения в документ

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Transmit = iObject.Transmit	Получить свойство (*)
iObject.Transmit = Transmit	Установить свойство (*)
Transmit = iObject.GetTransmit()	Получить свойство (**)
iObject.SetTransmit(Transmit)	Установить свойство (**)

Синтаксис COM:

iObject->get_Transmit(&Transmit)	Получить свойство
iObject->put_Transmit(Transmit)	Установить свойство

Значение свойства:

TRUE	- включена передача информации из текстовой части и дополнительных колонок объекта спецификации в подключенный к нему документ,
FALSE	- передача информации выключена.

Примечание:

Передача изменений производится в момент сохранения документа или с помощью специальной команды синхронизации. Обратная передача данных - из подключенного документа в спецификацию - осуществляется вне зависимости от состояния данного свойства. Запрос на эту передачу выдается при сохранении документа, подключенного к объекту спецификации, если обозначение или наименование этого документа было изменено.

IAttachedDocument – методы

Delete – Удалить ссылку на документ

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete([out, retval] BOOL* Result);

Примечание:

1. Метод позволяет удалить ссылку на документ.
2. После удаления нужных документов вызвать метод ISpecificationObject::Update.

Интерфейс ISpecificationBaseObject

Интерфейс базового объекта спецификации.

Иерархия:

IKompasAPIObject

ISpecificationObject

ISpecificationBaseObject

Описание:

Интерфейс позволяет получить и изменить параметры для базовых объектов спецификации. Для базовых объектов предусмотрена возможность автоматического заполнения колонок, сортировки, подключения графических объектов из сборочного чертежа, подключения деталей из сборки и т.д.

Примечание:

1. После изменения параметров объекта спецификации требуется вызвать метод ISpecificationObject::Update.
2. Данный интерфейс можно получить у коллекции базовых объектов спецификации ISpecificationBaseObjects.

ISpecificationBaseObject – свойства

AttributeNumber – Номер типа атрибута

Интерфейс...

Тип данных: double

Синтаксис Automation:

AttributeNumber = iObject.AttributeNumber	=	Получить свойство (*)
iObject.AttributeNumber = AttributeNumber	=	Установить свойство (*)
AttributeNumber = iObject.GetAttributeNumber()	=	Получить свойство (**)
iObject.SetAttributeNumber(AttributeNumber)		Установить свойство (**)

Синтаксис COM:

iObject->get_AttributeNumber(&AttributeNumber)	Получить свойство
iObject->put_AttributeNumber(AttributeNumber)	Установить свойство

Примечание:

Позволяет получить используемый тип атрибута для колонки с типом значения ksValueTypeRecord ISpecificationColumn::ValueType, и установить новый. Если требуется просто удалить тип атрибута, то в качестве значения этого свойства надо присвоить 0. В этом случае значение колонки с типом ksValueTypeRecord будет аналогично типу ksValueTypeString. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

CommentObjects – Вспомогательные объекты, прикрепленные к объекту

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationCommentObjects коллекции вспомогательных объектов спецификации

Синтаксис Automation:

CommentObjects = iObject.CommentObjects		Получить свойство (*)
CommentObjects = iObject.GetCommentObjects()		Получить свойство (**)

Синтаксис COM:

iObject->get_CommentObjects(&CommentObjects)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Documents – Документы, связанные с объектом спецификации

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Documents = iObject.Documents		Получить свойство (*)
Documents = iObject.GetDocuments()		Получить свойство (**)

Синтаксис COM:

iObject->get_Documents(&Documents)	Получить свойство

Примечание:

1. Связанными документами являются документы, в которых находится геометрия, подключенная к данному объекту, поэтому список связанных с объектом спецификации до-

кументов можно получить только для объектов в документе спецификации, в остальных документах геометрию можно подключить только из этого же документа.

2. Свойство возвращает переменную типа VARIANT на безопасный массив имен файлов, связанных с объектом спецификации SAFEARRAY (VT_ARRAY | VT_BSTR).
3. Свойство доступно только для чтения.

Draw – Показывать объект

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Draw = iObject.Draw	Получить свойство (*)
iObject.Draw = Draw	Установить свойство (*)
Draw = iObject.GetDraw()	Получить свойство (**)
iObject.SetDraw(Draw)	Установить свойство (**)

Синтаксис COM:

iObject->get_Draw(&Draw)	Получить свойство
iObject->put_Draw(Draw)	Установить свойство

Значения свойства:

TRUE	- объект виден в таблице спецификации в любом режиме работы с ней,
FALSE	- показывать объект в таблице спецификации не нужно; при этом объект не удаляется (он по-прежнему участвует в расчете номеров позиций).

Примечание:

Свойство позволяет скрыть объект спецификации.

DrawPosition – Показывать позицию

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DrawPosition = iObject.DrawPosition	=	Получить свойство (*)
iObject.DrawPosition = DrawPosition	=	Установить свойство (*)
DrawPosition = iObject.GetDrawPosition()	=	Получить свойство (**)

iObject.SetDrawPosition(DrawPosition)	Установить свойство (**)

Синтаксис COM:

iObject->get_DrawPosition(&DrawPosition)	Получить свойство
iObject->put_DrawPosition(DrawPosition)	Установить свойство

Значения свойства:

TRUE	- в колонке "Позиция" объекта отображается ее содержимое,
FALSE	- колонка "Позиция" будет показываться пустой.

Примечание:

Свойство позволяет скрыть номер позиции объекта в спецификации.

EditSourceObject – Передавать изменения в документ – владелец объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EditSourceObject = Object.EditSourceObject	=	Получить свойство (*)
Object.EditSourceObject = EditSourceObject	=	Установить свойство (*)
EditSourceObject = Object.GetEditSourceObject()	=	Получить свойство (**)
Object.SetEditSourceObject(EditSourceObject)		Установить свойство (**)

Синтаксис COM:

Object.get_EditSourceObject(&EditSourceObject)	Получить свойство
Object.put_EditSourceObject(EditSourceObject)	Установить свойство

SpcUsed – Признаки использования в спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SpcUsed = Object.SpcUsed(SpcUsedType)	Получить свойство (*)
Object.SpcUsed(SpcUsedType) = SpcUsed	Установить свойство (*)
SpcUsed = Object.GetSpcUsed(SpcUsedType)	Получить свойство (**)
Object.SetSpcUsed(SpcUsedType, SpcUsed)	Установить свойство (**)

Синтаксис COM:

Object.get_SpcUsed(SpcUsedType, &SpcUsed)	Получить свойство
Object.put_SpcUsed(SpcUsedType, SpcUsed)	Установить свойство

Входные параметры:

SpcUsedType	- Признаки использования в спецификации из перечисления ksSpcUsedTypeEnum .

Geometry – Геометрия объекта спецификации

Интерфейс...

Тип данных: VARIANT

Значения свойства:

Массив SafeArray типа VT_ARRAY | VT_DISPATCH

Синтаксис Automation:

Geometry = iObject.Geometry	Получить свойство (*)
iObject.Geometry = Geometry	Установить свойство (*)
Geometry = iObject.GetGeometry()	Получить свойство (**)

iObject.SetGeometry(Geometry)	Установить свойство (**)

Синтаксис COM:

iObject->get_Geometry(&Geometry)	Получить свойство
iObject->put_Geometry(Geometry)	Установить свойство

Примечание:

У базового объекта СП может быть установлена геометрия, которая ассоциируется с данным объектом.
Для 2D документа это массив IDrawingObject.
Для 3D документа это массив объектов:

- IPart7(может быть один компонент),
- IBody7 (тела верхнего уровня сборки),
- IPositionLeader (обозначения позиций верхнего уровня сборки).

Получить можно всегда массив объектов. Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
Установить можно как массив, так и одиночный объект. При установке геометрии старая геометрия очищается.

При установке геометрии используется текущее значение флага ISpecificationBaseObject::SyncronizeWithProperties.

Performance – Объект является исполнением

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Performance = iObject.Performance	=	Получить свойство (*)
iObject.Performance = Performance	=	Установить свойство (*)
Performance = iObject.GetPerformance()	=	Получить свойство (**)
iObject.SetPerformance(Performance)		Установить свойство (**)

Синтаксис COM:

iObject->get_Performance(&Performance)	Получить свойство

iObject->put_Performance(Performance)	Установить свойство

Значения свойства:

TRUE	- Объект является исполнением,
FALSE	- Объект не является исполнением.

Примечание:

Объекты спецификации, являющиеся исполнением, отличаются от простого объекта наличием суффикса исполнения. Их обозначение должно заканчиваться суффиксом с номером исполнения. Например: -02, -03, -10.

Отображение объектов, являющимися исполнениями зависит от настроек стиля спецификации.

SummaryCount – Суммарное количество для исполнения

Интерфейс...

Тип данных: double

Синтаксис Automation:

SummaryCount iObject.SummaryCount(ColumnTypeNumber, BlockNumber)	=	Получить свойство (*)
SummaryCount iObject.GetSummaryCount(ColumnTypeNumber, BlockNumber)	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SummaryCount(ColumnTypeNumber, BlockNumber, &SummaryCount)	Получить свойство

Входные параметры:

ColumnTypeNumber	- номер колонки данного типа,
BlockNumber	- номер блока исполнений.

Примечание:

1. Свойство доступно только для чтения.
2. Данное свойство возвращает суммарное значение в колонке "Количество" указанного номера колонки для одинаковых объектов указанного блока исполнений.

SyncronizeWithProperties – Брать данные из свойств и передавать данные в свойства объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SyncronizeWithProperties = Object.SyncronizeWithProperties	Получить свойство (*)
Object.SyncronizeWithProperties = SyncronizeWithProperties	Установить свойство (*)
SyncronizeWithProperties = Object.GetSyncronizeWithProperties()	Получить свойство (**)
Object.SetSyncronizeWithProperties(SyncronizeWithProperties)	Установить свойство (**)

Синтаксис COM:

Object.get_SyncronizeWithProperties(&SyncronizeWithProperties)	Получить свойство
Object.put_SyncronizeWithProperties(SyncronizeWithProperties)	Установить свойство

UniqueMetaObjectKey – Уникальный идентификатор объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UniqueMetaObjectKey = Object.UniqueMetaObjectKey	Получить свойство (*)
UniqueMetaObjectKey = Object.GetUniqueMetaObjectKey()	Получить свойство (**)

Синтаксис COM:

Object.get_UniqueMetaObjectKey(&UniqueMetaObjectKey))	Получить свойство

Примечание

Свойство доступно только для чтения

ISpecificationBaseObject – методы

ClearGeometry – Очистить геометрию объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL ClearGeometry(BOOL ClearGeometry, BOOL ClearLeaders);

Синтаксис COM:

HRESULT ClearGeometry(BOOL ClearGeometry, BOOL ClearLeaders, BOOL * Result);

Входные параметры:

ClearGeometry	- очистить геометрию,
ClearLeaders	- очистить линии выноски.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

IncludeGeometry – Добавить геометрию к объекту спецификации

Интерфейс...

Справка системы КОМПАС

KOMPAS.chm::/1631_171_8_Avto_sozd_spec_po_sborke.htm

Синтаксис Automation:

BOOL IncludeGeometry(VARIANT PVal, BOOL FillText);

Синтаксис COM:

HRESULT IncludeGeometry(VARIANT PVal, BOOL FillText, BOOL * Result);

Входные параметры:

PVal	- Объект VT_DISPATCH или список объектов VT_ARRAY I VT_DISPATCH,

FillText	- признак автоматического заполнения колонок: наименование, обозначение, масса взяты из свойств объекта.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Если при подключении компонента присылается флаг fillTexts = true, в объекте спецификации взводится флаг ISpecificationBaseObject::SyncronizeWithProperties.

SetMaterial – Установить материал в объект спецификации и связанный с ним документ

Интерфейс...

Синтаксис Automation:

BOOL SetMaterial(BSTR Material, double Density);

Синтаксис COM:

HRESULT SetMaterial(BSTR Material, double Density, BOOL * Result);

Входные параметры:

Material	- материал,
Density	- плотность.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetSection – Установить номер раздела

Интерфейс...

Синтаксис Automation:

BOOL SetSection(long Val);

Синтаксис COM:

HRESULT SetSection(long Val, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Входные параметры:

Val - номер раздела.

Примечание:

Задаваемый номер раздела должен присутствовать в данной спецификации.

Узнавать текущий номер раздела объекта надо у базового интерфейса ISpecificationObject.

Интерфейс ISpecificationColumn

Интерфейс колонки объекта спецификации.

Иерархия:

IKompasAPIObject

ISpecificationColumn

Описание:

Колонка спецификации представляет ячейку строки объекта спецификации в таблице спецификации. Колонка содержит информацию определенного типа, который определяется типом колонки ColumnType.

Если в спецификации необходимо иметь несколько колонок с одинаковым типом, то они отличаются друг от друга номером колонки данного типа ColumnTypeNumber.

Различаются основные и дополнительные колонки объектов спецификации. Основные колонки отображаются в таблице спецификации и выводятся на печать. Дополнительные колонки содержат сведения, дополняющие информацию, включаемую в стандартный бланк. В таблице спецификации такие колонки не видны и на печать не выводятся. Примером информации в дополнительных колонках могут служить масса и стоимость объекта. В дополнительные колонки вводят и любую другую информацию об объекте (код ОКП, материал, текстовый комментарий и т.д.). Их количество и состав определяются потребностями пользователя и хранятся в стиле спецификации.

Примечание:

Данный интерфейс можно получить у коллекции колонок объекта спецификации ISpecificationColumns.

ISpecificationColumn – свойства

AttributeNumber – Номер типа атрибута

Интерфейс...

Тип данных: double

Синтаксис Automation:

AttributeNumber = iObject.AttributeNumber	=	Получить свойство (*)
iObject.AttributeNumber = AttributeNumber	=	Установить свойство (*)
AttributeNumber = iObject.GetAttributeNumber()	=	Получить свойство (**)
iObject.SetAttributeNumber(AttributeNumber)		Установить свойство (**)

Синтаксис COM:

iObject->get_AttributeNumber(&AttributeNumber)	Получить свойство
iObject->put_AttributeNumber(AttributeNumber)	Установить свойство

Примечание:

1. Используется только для колонки с типом значения ksValueTypeRecord ISpecificationColumn::ValueType.

   Если требуется просто удалить тип атрибута, то в качестве значения этого свойства надо присвоить 0. В этом случае значение колонки с типом ksValueTypeRecord будет аналогично типу ksValueTypeString.

2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

BlockNumber – Номер блока

Интерфейс...

Тип данных: long

Синтаксис Automation:

BlockNumber = iObject.BlockNumber	=	Получить свойство (*)
BlockNumber = iObject.GetBlockNumber()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_BlockNumber(&BlockNumber)	Получить свойство

Примечание:

1. Определяет какому блоку исполнений соответствует данная колонка. Используется для базовых объектов групповых спецификаций.
2. Свойство доступно только для чтения.

ColumnItems – Элементы колонки объекта спецификации

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationColumnItems коллекции элементов колонки объекта спецификации

Синтаксис Automation:

ColumnItems = iObject.ColumnItems	=	Получить свойство (*)
ColumnItems = iObject.GetColumnItems()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ColumnItems(&ColumnItems)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ColumnName – Наименование колонки

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ColumnName = Object.ColumnName	=	Получить свойство (*)
ColumnName = Object.GetColumnName()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ColumnName(&ColumnName)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ColumnType – Тип колонки

Интерфейс...

Тип данных: из перечисления ksSpecificationColumnTypeEnum

Синтаксис Automation:

ColumnType = iObject.ColumnType	=	Получить свойство (*)
ColumnType = iObject.GetColumnType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ColumnType(&ColumnType)	Получить свойство

Примечание:

1. Тип колонки определяет, какая именно информация будет храниться в этой колонке (независимо от ее названия). Тип колонки используется при автоматическом выполнении различных операций. Например:

- если тип колонки - ПОЗИЦИЯ (ksSColumnPosition), то именно в эту колонку будут заноситься номера позиций при их автоматическом перерасчете;
- если тип колонки - ЗОНА (ksSColumnZone), то именно в эту колонку будут попадать номера зон;
- если тип колонки - ФОРМАТ (ksSColumnFormat), то в нее будут переноситься обозначения форматов из штампов чертежей деталей.

   При перерождении спецификации в спецификацию другого стиля (например, простой спецификации в групповую) колонки превращаются в колонки такого же типа (если он есть в стиле той спецификации, в которую перерождается исходная).

2. Свойство доступно только для чтения.

ColumnTypeNumber – Номер колонки данного типа

Интерфейс...

Тип данных: long

Синтаксис Automation:

ColumnTypeNumber = iObject.ColumnTypeNumber	=	Получить свойство (*)
ColumnTypeNumber = iObject.GetColumnTypeNumber( )	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ColumnTypeNumber(&ColumnTypeNumber)	Получить свойство

Примечание:

1. Иногда однородная (но не одинаковая) информация должна храниться в нескольких колонках и по этой причине они имеют одинаковый тип. Например, несколько колонок групповой спецификации содержат информацию о количестве, но числа в них разные, т.к. они относятся к разным исполнениям. Чтобы отличать типы таких колонок, вместе с типом задается еще и номер, отличающий колонку этого типа от других колонок такого же типа. Если разные колонки имеют одинаковый тип, то номера колонок этого типа должны быть разными.

   Обратите внимание, что во вкладке Колонки диалога настройки стиля спецификации в скобках рядом с типом колонки показан этот номер (например, КОЛИЧЕСТВО[2]). Если несколько колонок имеют одинаковый тип, то при автоматическом выполнении операций будет использоваться та из них, тип которой имеет номер 1. Например, в спецификации есть колонки с типом ЗОНА[1] и ЗОНА[2]; при автоматическом расчете номера зон будут заноситься в колонку с типом ЗОНА[1].
2. Свойство доступно только для чтения.

CountUniteCells – Количество объединяемых с текущей ячеек

Интерфейс...

Тип данных: long

Синтаксис Automation:

CountUniteCells = Object.CountUniteCells	=	Получить свойство (*)
Object.CountUniteCells = CountUniteCells	=	Установить свойство (*)
CountUniteCells = Object.GetCountUniteCells()	=	Получить свойство (**)
Object.SetCountUniteCells(CountUniteCells)		Установить свойство (**)

Синтаксис COM:

Object.get_CountUniteCells(&CountUniteCells)	Получить свойство
Object.put_CountUniteCells(CountUniteCells)	Установить свойство

Number – Номер колонки, начиная с единицы

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = Object.Number	Получить свойство (*)
Number = Object.GetNumber()	Получить свойство (**)

Синтаксис COM:

Object.get_Number(&Number)	Получить свойство

Примечание:

Свойство доступно только для чтения

Text – Текст колонки объекта спецификации

Интерфейс...

Тип данных: Указатель на интерфейс IText

Синтаксис Automation:

Text = Object.Text	Получить свойство (*)
Text = Object.GetText()	Получить свойство (**)

Синтаксис COM:

Object.get_Text(&Text)	Получить свойство

Примечание:

Свойство позволяет получить и изменить текст колонки объекта спецификации.

Свойство доступно только для чтения

ValueType – Тип значения колонки

Интерфейс...

Тип данных: из перечисления ksValueTypeEnum

Синтаксис Automation:

ValueType = iObject.ValueType	Получить свойство (*)
ValueType = iObject.GetValueType()	Получить свойство (**)

Синтаксис COM:

iObject->get_ValueType(&ValueType)	Получить свойство

Значение свойства:

ksValueTypeInteger	- Колонка содержит целое со знаком и состоит из одного элемента,
ksValueTypeFloat	- Колонка содержит вещественное число и состоит из одного элемента,
ksValueTypeString	- Колонка содержит строку длиной 255 символов и состоит из одного элемента,
ksValueTypeRecord	- Колонка содержит запись, количество элементов в колонке определяется типом атрибута.

Примечание:

1. Тип ksValueTypeRecord доступен только для базовых объектов спецификации, и может быть только одна колонка данного типа. Тип атрибута для такой колонки определяется при создании базового объекта (см. AttributeNumber, ISpecificationBaseObject::AttributeNumber и ISpecificationBaseObjects::Add). Если тип атрибута 0 или объект не является базовым объектом спецификации, то данная колонка используется аналогично колонкам с типом значения ksValueTypeString.
2. Свойство доступно только для чтения.

Интерфейс ISpecificationColumnItem

Интерфейс элемента колонки объекта спецификации.

Иерархия:

IKompasAPIObject

ISpecificationColumnItem

Описание:

Элемент колонки позволяет работать с данными колонки определенного типа. Этот тип соответствует типу колонки объекта спецификации или типу колонки атрибута для колонок типа Запись (ksValueTypeRecord). Если у колонки типа Запись тип атрибута не задан, то она состоит из одного элемента с типом Строка (ksValueTypeString).

Спецификация на текущем этапе позволяет иметь тип Запись только в одной колонке.

Примечание:

Данный интерфейс можно получить у коллекции элементов колонки объекта спецификации ISpecificationColumnItems.

ISpecificationColumnItem – свойства

Key – Ключ

Интерфейс...

Тип данных: short

Синтаксис Automation:

Key = iObject.Key	Получить свойство (*)
iObject.Key = Key	Установить свойство (*)
Key = iObject.GetKey()	Получить свойство (**)
iObject.SetKey(Key)	Установить свойство (**)

Синтаксис COM:

iObject->get_Key(&Key)	Получить свойство
iObject->put_Key(Key)	Установить свойство

Примечание:

Свойство задает приоритет для сортировки. Если ключ равен 0, сортировки нет. Используется для колонки типа Запись, если задан тип атрибута и позволяет определить приоритет каждого из элементов колонки для сортировки. Обратите внимание, что во вкладке Колонки диалога настройки стиля спецификации в скобках рядом с типом колонки показан этот номер (например, КОЛИЧЕСТВО[2]). Если несколько колонок имеют одинаковый тип, то при автоматическом выполнении операций будет использоваться та из них, тип которой имеет номер 1. Например, в спецификации есть колонки с типом ЗОНА[1] и ЗОНА[2]; при автоматическом расчете номера зон будут заноситься в колонку с типом ЗОНА[1].

Value – Значение элемента

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Value = iObject.Value	Получить свойство (*)
iObject.Value = Value	Установить свойство (*)
Value = iObject.GetValue()	Получить свойство (**)
iObject.SetValue(Value)	Установить свойство (**)

Синтаксис COM:

iObject->get_Value(&Value)	Получить свойство
iObject->put_Value(Value)	Установить свойство

Примечание:

Позволяет получить и установить значение элемента колонки объекта спецификации.

В зависимости от типа значения ValueType возвращаемое значение имеет тип:

- ksValueTypeInteger - VT_I4 (long),
- ksValueTypeFloat - VT_R8 (double),
- ksValueTypeString - VT_BSTR (строка).

ValueType – Тип значения элемента колонки

Интерфейс...

Тип данных: из перечисления ksValueTypeEnum

Синтаксис Automation:

ValueType = iObject.ValueType		Получить свойство (*)
ValueType iObject.GetValueType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ValueType(&ValueType)	Получить свойство

Значение свойства:

ksValueTypeInteger	- Элемент колонки содержит целое со знаком и состоит из одного элемента,
ksValueTypeFloat	- Элемент колонки – вещественное число и состоит из одного элемента,
ksValueTypeString	- Элемент колонки – строка длиной 255 символов и состоит из одного элемента,
ksValueTypeRecord	- не используется.

Примечание:

1. См также ISpecificationColumnItem::Value.
2. Свойство доступно только для чтения.

Visible – Видимость элемента

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = iObject.Visible	Получить свойство (*)
iObject.Visible = Visible	Установить свойство (*)
Visible = iObject.GetVisible()	Получить свойство (**)
iObject.SetVisible(Visible)	Установить свойство (**)

Синтаксис COM:

iObject->get_Visible(&Visible)	Получить свойство
iObject->put_Visible(Visible)	Установить свойство

Значение свойства:

TRUE	- содержимое элемента выводится в колонке объекта спецификации,
FALSE	- содержимое элемента не выводится в колонке объекта спецификации.

Интерфейс ISpecificationColumnStyle

Интерфейс стиля колонки спецификации.

Иерархия:

IKompasAPIObject

ISpecificationColumnStyle

Примечание:

Данный интерфейс можно получить от интерфейса стиля колонки спецификации ISpecificationColumnStyles, используя свойство ISpecificationColumnStyles::Item.

ISpecificationColumnStyle – свойства

AttributeKey1 – Первый ключ атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

AttributeKey1	=	Получить свойство
iObject.AttributeKey1		(*)

AttributeKey1 iObject.GetAttributeKey1()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AttributeKey1 (&AttributeKey1)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Запись (см. ValueType).
2. Для данной колонки можно будет использовать только те типы атрибутов, первый ключ которых совпадает с указанным.
3. Свойство доступно только для чтения.

AttributeKey2 – Второй ключ атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

AttributeKey2 iObject.AttributeKey2	=	Получить свойство (*)
AttributeKey2 iObject.GetAttributeKey2()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AttributeKey2 (&AttributeKey2)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Запись (см. ValueType).
2. Для данной колонки можно будет использовать только те типы атрибутов, второй ключ которых совпадает с указанным.
3. Свойство доступно только для чтения.

AttributeKey3 – Третий ключ атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

AttributeKey3 iObject.AttributeKey3	=	Получить свойство (*)

AttributeKey3 iObject.GetAttributeKey3()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AttributeKey3 (&AttributeKey3)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Запись (см. ValueType).
2. Для данной колонки можно будет использовать только те типы атрибутов, третий ключ которых совпадает с указанным.
3. Свойство доступно только для чтения.

AttributeKey4 – Четвертый ключ атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

AttributeKey4 iObject.AttributeKey4	=	Получить свойство (*)
AttributeKey4 iObject.GetAttributeKey4()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AttributeKey4 (&AttributeKey4)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Вещественный или Целый (см. ValueType).
2. Определяет минимальное значение числа в колонке.
3. Свойство доступно только для чтения.

AttributeLibraryName – Имя файла библиотеки типов атрибутов

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

AttributeLibraryName iObject.AttributeLibraryName	=	Получить свойство (*)

AttributeLibraryName = iObject.GetAttributeLibraryName ()		Получить свойство (**)

Синтаксис COM:

iObject->get_AttributeLibraryName (&AttributeLibraryName)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Запись (см. ValueType).
2. Структура записи в колонке определяется типом атрибута из библиотеки типов атрибутов.
3. Для данной колонки можно будет использовать типы атрибутов только из указанной библиотеки
4. Свойство доступно только для чтения.

CalculateSum – Рассчитывать сумму значений для колонки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CalculateSum iObject.CalculateSum	=	Получить свойство (*)
CalculateSum iObject.GetCalculateSum()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_CalculateSum (&CalculateSum)	Получить свойство

Значения свойства:

TRUE	- складывать значения в таблице при вызове команды "Сложить значения в колонках",
FALSE	- складывать значения в таблице запрещено.

Примечание:

1. Используется только для колонок со значением типа Вещественный или Целый (см. ValueType).
2. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
3. Свойство доступно только для чтения.

ColumnType – Тип колонки

Интерфейс...

Тип данных: из перечисления ksSpecificationColumnTypeEnum

Синтаксис Automation:

ColumnType = iObject.ColumnType	=	Получить свойство (*)
ColumnType = iObject.GetColumnType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ColumnType (&ColumnType)	Получить свойство

Примечание:

1. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
2. Свойство доступно только для чтения.

Edit – Редактируемая колонка в данном разделе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Edit = iObject.Edit	Получить свойство (*)
Edit = iObject.GetEdit()	Получить свойство (**)

Синтаксис COM:

iObject->get_Edit (&Edit)	Получить свойство

Значения свойства:

TRUE	- в колонку можно вводить данные,
FALSE	- заполнение колонки в настраиваемом разделе запрещено.

Примечание:

1. Используется только для колонок, полученных у стиля раздела спецификации ISpecificationSectionStyle..
2. Свойство доступно только для чтения.

MaxValue – Максимальное значение

Интерфейс...

Тип данных: long

Синтаксис Automation:

MaxValue = iObject.MaxValue		Получить свойство (*)
MaxValue iObject.GetMaxValue()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_MaxValue (&MaxValue)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Вещественный или Целый (см. ValueType).
2. Определяет минимальное значение числа в колонке.
3. Свойство доступно только для чтения.

MinValue – Минимальное значение

Интерфейс...

Тип данных: long

Синтаксис Automation:

MinValue = iObject.MinValue		Получить свойство (*)
MinValue iObject.GetMinValue()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_MinValue (&MinValue)	Получить свойство

Примечание:

1. Используется только для колонок со значением типа Вещественный или Целый (см. ValueType).
2. Определяет минимальное значение числа в колонке.
3. Свойство доступно только для чтения.

MultiplyToCount – При расчете суммы домножать на количество

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MultiplyToCount = iObject.MultiplyToCount	=	Получить свойство (*)
MultiplyToCount = iObject.GetMultiplyToCount()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_MultiplyToCount (&MultiplyToCount)	Получить свойство

Значения свойства:

TRUE	- перед сложением значение в колонке должно быть умножено на число в колонке "Количество",
FALSE	- складывать без изменений.

Примечание:

1. Используется только для колонок со значением типа Вещественный или Целый (см. ValueType), если ISpecificationColumnStyle::CalculateSum равно TRUE.
2. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
3. Свойство доступно только для чтения.

Name – Имя колонки

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Number – Номер колонки данного типа

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер колонки для колонок с таким же типом. Иногда однородная (но не одинаковая) информация должна храниться в нескольких колонках, и по этой причине они имеют одинаковый тип ColumnType Например, несколько колонок групповой спецификации содержат информацию о количестве, но числа в них разные, т.к. они относятся к разным исполнениям. Чтобы отличать такие колонки, вместе с типом задается еще и номер, отличающий колонку этого типа от других колонок такого же типа. Если разные колонки имеют одинаковый тип, то номера колонок этого типа должны быть разными.
2. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
3. Свойство доступно только для чтения.

StampLinkID – Номер ячейки штампа для связи

Интерфейс...

Тип данных: long

Синтаксис Automation:

StampLinkID = iObject.StampLinkID	=	Получить свойство (*)
StampLinkID = iObject.GetStampLinkID()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_StampLinkID (&StampLinkID)	Получить свойство

Значения свойства:

Номер ячейки штампа.

Примечание:

1. Данное свойство определяет ячейку штампа чертежа детали, данные из которой могут автоматически передаваться в данную колонку спецификации. Например, в колонку Формат можно передавать текст из ячейки Формат штампа чертежа детали.
2. Свойство доступно только для чтения.

TextDown – Текст расположен в нижней части колонки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

TextDown = iObject.TextDown		Получить свойство (*)
TextDown iObject.GetTextDown()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_TextDown (&TextDown)	Получить свойство

Значения свойства:

TRUE	- в колонке однострочные тексты будут располагаться в той строке, в которой кончается текст предыдущей колонки,
FALSE	- тексты будут располагаться в верхней части колонки.

Примечание:

1. Чаще всего эту опцию включают при настройке колонки Количество, когда требуется, чтобы количество было записано в той строке, где заканчивается наименование.
2. Используется только для основных колонок спецификации.
3. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
4. Свойство доступно только для чтения.

UseForSectionTitle – Использовать колонку для вывода имени раздела

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseForSectionTitle = iObject.UseForSectionTitle		Получить свойство (*)
UseForSectionTitle = iObject.GetUseForSectionTitle()		Получить свойство (**)

Синтаксис COM:

iObject->get_UseForSectionTitle (&UseForSectionTitle)	Получить свойство

Значения свойства:

TRUE	- в колонке будут располагаться заголовки разделов спецификации,
FALSE	- колонка будет пустой для объектов спецификации "Имя раздела".

Примечание:

1. В стандартных спецификациях заголовки разделов обычно располагаются в колонке Наименование. Значение этого свойства TRUE - может быть только для одной колонки спецификации. Если у всех колонок значение этого свойства FALSE, заголовки разделов показываться не будут (даже при включении в настройке разделов опции Показывать заголовки разделов).
2. Используется только для основных колонок спецификации.
3. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
4. Свойство доступно только для чтения.

UseIn3D – Используется в модели

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseIn3D = iObject.UseIn3D	Получить свойство (*)
UseIn3D = iObject.GetUseIn3D()	Получить свойство (**)

Синтаксис COM:

iObject->get_UseIn3D (&UseIn3D)	Получить свойство

Значения свойства:

TRUE	- в колонку можно было вводить данные при формировании объектов спецификации в документе-модели,
FALSE	- заполнение колонки в объектах спецификации модели невозможно.

Примечание:

1. Свойство позволяет настроить стиль спецификации так, чтобы в документах-моделях формировались частично заполненные объекты спецификации. Это позволяет избежать конфликтов взаимодействия документа-модели со спецификацией и чертежом.
2. Используется только для колонок, полученных у стиля спецификации ISpecificationStyle.
3. Свойство доступно только для чтения.

ValueType – Тип значения колонки

Интерфейс...

Тип данных: из перечисления ksValueTypeEnum

Синтаксис Automation:

ValueType = iObject.ValueType		Получить свойство (*)
ValueType iObject.GetValueType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_ValueType (&ValueType)	Получить свойство

Примечание:

1. Тип "Запись" доступен только для основных колонок.
2. Свойство доступно только для чтения.

Интерфейс ISpecificationCommentObject

Интерфейс вспомогательного объекта спецификации.

Иерархия:

IKompasAPIObject

ISpecificationObject

ISpecificationCommentObject

Описание:

Интерфейс позволяет получить и изменить параметры вспомогательного объекта спецификации. В отличие от базового для вспомогательного объекта не предусмотрены сервисные функции, выполнение которых обеспечивает спецификация. Вспомогательные объекты не сортируются автоматически и т.д. Вспомогательные объекты рекомендуется использовать для выполнения таких приемов оформления спецификации, которые не могут быть обеспечены вводом базовых объектов. Например, при помощи вспомогательного объекта спецификации можно ввести произвольный текст (комментарий) в таблицу спецификации или создать пустую строку в середине раздела.

Примечание:

1. После изменения параметров объекта спецификации требуется вызвать метод ISpecificationObject::Update.
2. Данный интерфейс можно получить у коллекции базовых объектов спецификации ISpecificationCommentObjects.

ISpecificationCommentObject – свойства

BaseObject – Базовый объект спецификации, к которому прикреплен данный объект

Интерфейс...

Тип данных: указатель на интерфейс базового объекта спецификации ISpecificationBaseObject

Синтаксис Automation:

BaseObject = iObject.BaseObject	Получить свойство (*)
BaseObject = iObject.GetBaseObject()	Получить свойство (**)

Синтаксис COM:

iObject->get_BaseObject(&BaseObject)	Получить свойство

Примечание:

Свойство доступно только для чтения.

BlockNumber – Номер блока

Интерфейс...

Тип данных: long

Синтаксис Automation:

BlockNumber	=	Получить свойство
iObject.BlockNumber		(*)
iObject.BlockNumber	=	Установить свойство
BlockNumber		(*)
BlockNumber	=	Получить свойство
iObject.GetBlockNumber()		(**)
iObject.SetBlockNumber(		Установить свойство
BlockNumber)		(**)

Синтаксис COM:

iObject->get_BlockNumber(	Получить свойство
&BlockNumber)	
iObject->put_BlockNumber(	Установить свойство
BlockNumber)	

Примечание:

Установить номер блока можно только для нового вспомогательного объекта спецификации.

EditSourceObject – Передавать изменения в документ – владелец объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

EditSourceObject	=	Получить свойство
Object.EditSourceObject		(*)
Object.EditSourceObject	=	Установить свойство
EditSourceObject		(*)
EditSourceObject	=	Получить свойство
Object.GetEditSourceObject()		(**)
Object.SetEditSourceObject(		Установить свойство
EditSourceObject)		(**)

Синтаксис COM:

Object.get_EditSourceObject(	Получить свойство
&EditSourceObject)	
Object.put_EditSourceObject(	Установить свойство
EditSourceObject)	

ISpecificationCommentObject – методы

SetSection – Установить номер раздела

Интерфейс...

Синтаксис Automation:

BOOL SetSection(long Val);

Синтаксис COM:

HRESULT SetSection(long Val, BOOL * Result);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Входные параметры

Val - номер раздела.

Примечание:

Задаваемый номер раздела должен присутствовать в данной спецификации.

Узнавать текущий номер раздела объекта надо у базового интерфейса ISpecificationObject.

Интерфейс ISpecificationDescription

Интерфейс описания спецификации у документа.

Иерархия:

IKompasAPIObject

ISpecificationDescription

Описание:

Описание спецификации включает в себя имя файла спецификации, подключенной к графическому документу и стиль этой спецификации.

Для фрагмента и детали описание спецификации содержит только стиль спецификации. Это связано с тем, что фрагмент и деталь непосредственно к спецификации не подключаются, но в них можно создавать объекты спецификации для передачи в чертежи и сборки.

Примечание:

Установка свойств доступна только для графических документов и документов-моделей. Для документа спецификации свойства интерфейса работают в режиме "только для чтения". После изменения параметров оформления нужно вызвать метод Update, чтобы новые данные вступили в силу.

Данный интерфейс можно получить у коллекции описаний спецификации ISpecificationDescriptions.

ISpecificationDescription – свойства

Active – Текущее описание

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Active = iObject.Active	Получить свойство (*)
iObject.Active = Active	Установить свойство (*)
Active = iObject.GetActive()	Получить свойство (**)
iObject.SetActive (Active)	Установить свойство (**)

Синтаксис COM:

iObject->get_Active (&Active)	Получить свойство
iObject->put_Active (Active)	Установить свойство

Значения свойства:

TRUE	- описание активно,
FALSE	- описание неактивно.

Примечание:

1. Свойство возвращает признак активности описания и позволяет сделать описание активным. То есть установить свойство можно только в TRUE.
2. После изменения параметров оформления метод Update вызывать не требуется.

BaseObjects – Базовые объекты спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationBaseObjects коллекции базовых объектов спецификации

Синтаксис Automation:

Objects = iObject.BaseObjects	Получить свойство (*)
Objects = iObject.GetBaseObjects()	Получить свойство (**)

Синтаксис COM:

iObject->get_BaseObjects (&Objects)	Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию базовых объектов спецификации.
2. Свойство доступно только для чтения.

CommentObjects – Объекты спецификации – комментарии

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationCommentObjects коллекции вспомогательных объектов спецификации

Синтаксис Automation:

Objects iObject.CommentObjects	=	Получить свойство (*)
Objects iObject.GetCommentObjects()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_CommentObjects (&Objects)	Получить свойство

Примечание:

1. Свойство позволяет получить коллекцию вспомогательных объектов спецификации.
2. Свойство доступно только для чтения.

CurrentObject – Текущий объект спецификации (выделенный или редактируемый в таблице спецификации)

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationObject объекта спецификации.

Синтаксис Automation:

CurrentObject iObject.CurrentObject	=	Получить свойство (*)
iObject.CurrentObject CurrentObject	=	Установить свойство (*)
CurrentObject iObject.GetCurrentObject()	=	Получить свойство (**)
iObject.SetCurrentObject (CurrentObject)		Установить свойство (**)

Синтаксис COM:

iObject->get_CurrentObject (&CurrentObject)	Получить свойство
iObject->put_CurrentObject (CurrentObject)	Установить свойство

Примечание:

1. С помощью данного свойства можно получить и установить текущий объект спецификации.
2. В графическом документе или документе-модели метод работает, если открыто окно спецификации в подчиненном режиме.

DelegateMode – Режим редактирования внешних объектов для описания спецификации сборки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DelegateMode = Object.DelegateMode	=	Получить свойство (*)
Object.DelegateMode = DelegateMode	=	Установить свойство (*)
DelegateMode = Object.GetDelegateMode()	=	Получить свойство (**)
Object.SetDelegateMode(DelegateMode)		Установить свойство (**)

Синтаксис COM:

Object.get_DelegateMode(&DelegateMode)	Получить свойство
Object.put_DelegateMode(DelegateMode)	Установить свойство

Примечание:

Позволяет получать и создавать внешние объекты спецификации в документе типа "сборка".

LayoutName – Имя файла библиотеки стилей

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

LayoutName	=	Получить свойство
iObject.LayoutName		(*)
iObject.LayoutName	=	Установить свойство
LayoutName		(*)
LayoutName	=	Получить свойство
iObject.GetLayoutName()		(**)
iObject.SetLayoutName		Установить свойство
(LayoutName)		(**)

Синтаксис COM:

iObject->get_LayoutName	Получить свойство
(&LayoutName)	
iObject->put_LayoutName	Установить свойство
(LayoutName)	

Примечание:

1. Свойство позволяет получить и установить имя библиотеки стилей, из которой будет взят стиль спецификации для данного описания.
2. После изменения параметров оформления нужно вызвать метод Update, чтобы новые данные вступили в силу.

NeedRebuild – Необходимость перестроения спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

NeedRebuild	=	Получить свойство
Object.NeedRebuild		(*)
Object.NeedRebuild	=	Установить свойство
NeedRebuild		(*)
NeedRebuild	=	Получить свойство
Object.GetNeedRebuild()		(**)
Object.SetNeedRebuild		Установить свойство
(NeedRebuild)		(**)

Синтаксис COM:

Object.get_NeedRebuild(	Получить свойство
&NeedRebuild)	
Object.put_NeedRebuild(	Установить свойство
NeedRebuild)	

Примечание:

1. Свойство позволяет временно отключить обновление окна документа спецификации из функций API при создании и изменении объектов спецификации. А также ускорить:

- создание и изменение большого количества объектов,
- работу функции изменения имен исполнений в групповых спецификациях.

2. При изменении объектов спецификации можно воспользоваться свойством ISpecificationDescription::PerformanceName или методом ISpecificationObject::Update.

Objects – Объекты спецификации

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Objects = iObject.Objects	Получить свойство (*)
Objects = iObject.GetObjects()	Получить свойство (**)

Синтаксис COM:

iObject->get_Objects (&Objects)	Получить свойство

Примечание:

1. Свойство позволяет получить полный список объектов спецификации. Порядок и состав списка соответствует визуальному отображению данного описания спецификации.
2. Свойство возвращает переменную типа VARIANT на безопасный массив интерфейсов SAFEARRAY (VT_ARRAY | VT_DISPATCH). Если возвращается один объект, то тип VARIANT-а будет VT_DISPATCH, если возвращается несколько объектов, то тип VARIANT-а будет VT_ARRAY | VT_DISPATCH.
3. Свойство доступно только для чтения.

PerformanceCount – Количество исполнений

Интерфейс...

Тип данных: long

Синтаксис Automation:

PerformanceCount = Object. PerformanceCount	Получить свойство (*)
Object. PerformanceCount = PerformanceCount	Установить свойство (*)
PerformanceCount = Object.Get PerformanceCount()	Получить свойство (**)
Object.Set PerformanceCount (PerformanceCount)	Установить свойство (**)

Синтаксис COM:

Object.get_ PerformanceCount(& PerformanceCount)	Получить свойство
Object.put_ PerformanceCount(PerformanceCount)	Установить свойство

PerformanceCountInBlock – Количество исполнений в блоке

Интерфейс...

Тип данных: long

Синтаксис Automation:

PerformanceCountInBlock = Object.PerformanceCountInBlock	Получить свойство (*)
PerformanceCountInBlock = Object.GetPerformanceCountInBlock()	Получить свойство (**)

Синтаксис COM:

Object.get_PerformanceCountInBlock(&PerformanceCountInBlock)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PerformanceName – Отображаемое имя исполнения

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

PerformanceName = iObject.PerformanceName(PerformanceIndex, BlockIndex)	Получить свойство (*)
iObject.PerformanceName(PerformanceIndex, BlockIndex) = PerformanceName	Установить свойство (*)

PerformanceName iObject.GetPerformanceName(PerformanceIndex, BlockIndex)	=	Получить свойство (**)
iObject.SetPerformanceName(PerformanceIndex, BlockIndex, PerformanceName)		Установить свойство (**)

Синтаксис COM:

iObject.get_PerformanceName(PerformanceIndex, BlockIndex, &PerformanceName)	Получить свойство
iObject.put_PerformanceName(PerformanceIndex, BlockIndex, PerformanceName)	Установить свойство

Входные параметры:

PerformanceIndex	- номер колонки типа "количество", начиная с 1(актуально для СП по варианту Б),
BlockIndex	- номер блока, начиная с 0 (актуально для СП по варианту А).

Примечание:

1. В текст обозначения исполнения, которое редактируется, нужно включать разделитель.
2. Для получения имен исполнений рекомендуется использовать метод ISpecificationDescription::GetPerformanceParam.
3. Для ускорения работы функции при массовом изменении спецификации рекомендуется воспользоваться методом ISpecificationDescription::NeedRebuild.

ShowAllObjects – Показывать все объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowAllObjects iObject.ShowAllObjects	=	Получить свойство (*)
iObject.ShowAllObjects ShowAllObjects	=	Установить свойство (*)
ShowAllObjects iObject.GetShowAllObjects()	=	Получить свойство (**)
iObject.SetShowAllObjects (ShowAllObjects)		Установить свойство (**)

Синтаксис COM:

iObject->get_ShowAllObjects (&ShowAllObjects)	Получить свойство
iObject->put_ShowAllObjects (ShowAllObjects)	Установить свойство

Значения свойства:

TRUE	- отображаются все объекты в описании,
FALSE	- одинаковые объекты отображаются одной строкой, а в колонке количество отображается.

Примечание:

Свойство позволяет получить или установить состояние режима работы, в котором на экране отображаются все объекты спецификации, в том числе базовые объекты спецификации, имеющие одинаковую текстовую часть, а также базовые объекты спецификации, в настройках дополнительных параметров которых выключена опция показа объекта в таблице.

Одинаковые объекты могут иметь одинаковые или разные номера позиций, могут находиться в соседних строках или в разных местах таблицы (в пределах одного раздела спецификации). Если объекты с одинаковой текстовой частью находятся в разных разделах спецификации, то они считаются разными.

ShowOnSheet – Показывать на листе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowOnSheet = iObject.ShowOnSheet	=	Получить свойство (*)
iObject.ShowOnSheet = ShowOnSheet	=	Установить свойство (*)
ShowOnSheet = iObject.GetShowOnSheet()	=	Получить свойство (**)
iObject.SetShowOnSheet (ShowOnSheet)		Установить свойство (**)

Синтаксис COM:

iObject->get_ShowOnSheet (&ShowOnSheet)	Получить свойство
iObject->put_ShowOnSheet (ShowOnSheet)	Установить свойство

Значения свойства:

TRUE - описание отображается на чертеже,
FALS - описание не отображается на чертеже.
E

Примечание:

1. Свойство позволяет отобразить или скрыть спецификацию в чертеже.
2. Свойство работает только в чертеже.

ShowExcludedObjects – Показывать только исключенные объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowExcludedObjects = Object.ShowExcludedObjects	Получить свойство (*)
Object.ShowExcludedObjects = ShowExcludedObjects	Установить свойство (*)
Object.SetShowExcludedObjects (ShowExcludedObjects))	Получить свойство (**)
iObject.SetShowOnSheet (ShowOnSheet)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowExcludedObjects(&ShowExcludedObjects)	Получить свойство
Object.put_ShowExcludedObjects(ShowExcludedObjects)	Установить свойство

SpecificationDocumentName – Имя подключенного файла документа спецификации

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DocumentName = iObject.SpecificationDocumentName	Получить свойство (*)

iObject.SpecificationDocumentName = DocumentName	Установить свойство (*)
DocumentName = iObject.GetSpecificationDocumentName()	Получить свойство (**)
iObject.SetSpecificationDocumentName (DocumentName)	Установить свойство (**)

Синтаксис COM:

iObject->get_SpecificationDocumentName (&DocumentName)	Получить свойство
iObject->put_SpecificationDocumentName (DocumentName)	Установить свойство

Примечание:

1. Свойство позволяет получить и установить полное имя файла документа спецификации подключенной к данному чертежу или сборке. Доступно только для чертежа или сборки.
2. После изменения параметров оформления нужно вызвать метод Update, чтобы новые данные вступили в силу.

SpecificationStyle – Стиль спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationStyle стиля спецификации

Синтаксис Automation:

Style = iObject.SpecificationStyle	Получить свойство (*)
Style = iObject.GetSpecificationStyle()	Получить свойство (**)

Синтаксис COM:

iObject->get_SpecificationStyle(&Style)	Получить свойство

Примечание:

1. Полученный интерфейс отражает содержимое стиля в библиотеке стилей для данного описания на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.
2. Свойство доступно только для чтения.

SpecificationTuning – Настройки спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationTuning настроек спецификации

Синтаксис Automation:

SpecificationTuning = iObject.SpecificationTuning	Получить свойство (*)
SpecificationTuning = iObject.GetSpecificationTuning()	Получить свойство (**)

Синтаксис COM:

iObject->get_SpecificationTuning (&SpecificationTuning)	Получить свойство

Примечание:

Свойство доступно только для чтения.

StyleID – Номер стиля в библиотеке стилей

Интерфейс...

Тип данных: long

Синтаксис Automation:

StyleID = iObject.StyleID	Получить свойство (*)
iObject.StyleID = StyleID	Установить свойство (*)
StyleID = iObject.GetStyleID()	Получить свойство (**)
iObject.SetStyleID (StyleID)	Установить свойство (**)

Синтаксис COM:

iObject->get_StyleID (&StyleID)	Получить свойство
iObject->put_StyleID (StyleID)	Установить свойство

Примечание:

1. Свойство позволяет получить и установить номер стиля в библиотеке стилей, из которой будет взят стиль спецификации для данного описания (см. LayoutName).
2. После изменения параметров оформления нужно вызвать метод Update, чтобы новые данные вступили в силу.

ISpecificationDescription – методы

CompareStyleWithLibStyle – Отличие стиля СП от библиотечного: 0 – не отличается, 1 –отличается, –1 – стиль не найден

Интерфейс...

Синтаксис Automation:

ksSpecificationStyleDifferenceTypeEnum CompareStyleWithLibStyle();

Синтаксис COM:

HRESULT CompareStyleWithLibStyle(ksSpecificationStyleDifferenceTypeEnum * Result);

Возвращаемое значение:

0	- не отличается,
1	- отличается,
-1	- стиль не найден.

Delete – Удалить описание

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete([out, retval] BOOL* Result);

Примечание:

Метод удаляет описание спецификации.

GetPerformanceParam – Получить параметры и имя колонки исполнения по индексу

Интерфейс...

Синтаксис Automation:

LPCTSTR GetPerformanceParam(long DisplayPerformanceIndex, long * PerformanceIndex, long * BlockIndex);

Синтаксис COM:

HRESULT GetPerformanceParam(long DisplayPerformanceIndex, long * PerformanceIndex, long * BlockIndex, BSTR * PerformanceName);

Входные параметры:

DisplayPerformanceIndex	- индекс отображаемого исполнения (начинается с единицы).

Выходные параметры:

PerformanceIndex	- номер колонки исполнения,
BlockIndex	- указатель на структуру параметров библиотечной панели свойств.

Возвращаемое значение:

- Имя исполнения.

Метод позволяет получить имя исполнения, номер исполнения и номер блока по индексу исполнения.

Примечание:

1. Индекс исполнения начинается с единицы.
2. В групповой спецификации по варианту Б, созданной по сборке, у которой есть исполнения с непоследовательными номерами исполнений, часть колонок исполнений может быть отключена и отображаться будут только те исполнения, которые пришли из сборки. В такой спецификации в одной строке объекта спецификации могут отображаться исполнения и количество из нескольких блоков.
   В групповой спецификации по варианту А также могут быть непоследовательные номера исполнений, если спецификация создана по сборке.
3. Метод рекомендуется использовать вместо свойства ISpecificationDescription::PerformanceName.PerformanceIndex и BlockIndex позволяют сопоставить имя исполнения и значение количества в конкретной колонке количество объектов спецификации.

Update – Изменить описание

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update ([out, retval] BOOL* Result);

Примечание:

Метод устанавливает новые параметры для описания спецификации.

Интерфейс ISpecificationObject

Интерфейс объекта спецификации.

Иерархия:

IKompasAPIObject

ISpecificationObject

Описание:

Различаются несколько типов объектов спецификации (см. ksSpecificationObjectTypeEnum). Данный интерфейс позволяет получить общие параметры для объектов спецификации и изменить общие параметры для базовых и вспомогательных объектов спецификации.

Примечание:

1. После изменения параметров объекта спецификации требуется вызвать метод ISpecificationObject::Update.
2. Данный интерфейс можно получить у базового ISpecificationBaseObject и вспомогательного ISpecificationCommentObject объектов спецификации и через безопасный массив ISpecificationDescription::Objects для остальных типов объектов спецификации.

ISpecificationObject – свойства

AdditionalBlock – Номер блока дополнительных разделов

Интерфейс...

Тип данных: long

Синтаксис Automation:

AdditionalBlock = iObject.AdditionalBlock	=	Получить свойство (*)
iObject.AdditionalBlock = AdditionalBlock	=	Установить свойство (*)
AdditionalBlock = iObject.GetAdditionalBlock()	=	Получить свойство (**)
iObject.SetAdditionalBlock (AdditionalBlock)		Установить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlock (&AdditionalBlock)	Получить свойство
iObject->put_AdditionalBlock (AdditionalBlock)	Установить свойство

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

AdditionalColumns – Дополнительные колонки

Тип данных: указатель на интерфейс ISpecificationColumns коллекции колонок объекта спецификации

Синтаксис Automation:

AdditionalColumns = iObject.AdditionalColumns	=	Получить свойство (*)
AdditionalColumns = iObject.GetAdditionalColumns()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalColumns (&AdditionalColumns)	Получить свойство

Примечание:

Свойство доступно только для чтения.

AdditionalSection – Номер дополнительного раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

AdditionalSection = iObject.AdditionalSection	=	Получить свойство (*)
iObject.AdditionalSection = AdditionalSection	=	Установить свойство (*)
AdditionalSection = iObject.GetAdditionalSection()	=	Получить свойство (**)
iObject.SetAdditionalSection (AdditionalSection)		Установить свойство (**)

Синтаксис COM:

iObject->get_AdditionalSection (&AdditionalSection)	Получить свойство
iObject->put_AdditionalSection (AdditionalSection)	Установить свойство

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

AttachedDocuments – Присоединенные документы

Интерфейс...

Тип данных: указатель на интерфейс IAttachedDocuments коллекции присоединенных документов

Синтаксис Automation:

AttachedDocuments = iObject.AttachedDocuments		Получить свойство (*)
AttachedDocuments = iObject.GetAttachedDocuments()		Получить свойство (**)

Синтаксис COM:

iObject->get_AttachedDocuments (&AttachedDocuments)	Получить свойство

Примечание:

1. Получить коллекцию можно только для базовых и вспомогательных объектов спецификации, если в стиле для заданного раздела установлен признак ручное заполнение или чтение из основной надписи.
2. Свойство доступно только для чтения.

BlockNumberByIndex – Номер блока

Интерфейс...

Тип данных: long

Синтаксис Automation:

BlockNumber = iObject.BlockNumberByIndex (index)		Получить свойство (*)
BlockNumber = iObject.GetBlockNumberByIndex (index)		Получить свойство (**)

Синтаксис COM:

iObject->get_BlockNumberByIndex (index, &BlockNumber)	Получить свойство

Входные параметры:

index	- индекс объекта в общем массиве объектов спецификации.

Примечание:

1. Свойство позволяет получить номер блока объекта спецификации.
2. Один и тот же базовый объект спецификации может быть вставлен в спецификации несколько раз в нескольких блоках.
3. Свойство доступно только для чтения.

Columns – Колонки объекта спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumns коллекции колонок объекта спецификации

Синтаксис Automation:

Columns = iObject.Columns	Получить свойство (*)
Columns = iObject.GetColumns()	Получить свойство (**)

Синтаксис COM:

iObject->get_Columns(&Columns)	Получить свойство

Примечание:

Свойство доступно только для чтения.

FirstOnSheet – Размещать объект в начале нового листа спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FirstOnSheet = iObject.FirstOnSheet	=	Получить свойство (*)
iObject.FirstOnSheet = FirstOnSheet	=	Установить свойство (*)
FirstOnSheet = iObject.GetFirstOnSheet()	=	Получить свойство (**)
iObject.SetFirstOnSheet (FirstOnSheet)		Установить свойство (**)

Синтаксис COM:

iObject->get_FirstOnSheet (&FirstOnSheet)	Получить свойство
iObject->put_FirstOnSheet (FirstOnSheet)	Установить свойство

Значения свойства:

TRUE	- объект спецификации разместить в начале новой страницы спецификации,
FALSE	- объект спецификации разместить после предыдущего.

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

IncrementPosition – Позиция объекта возрастает

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IncrementPosition = iObject.IncrementPosition	=	Получить свойство (*)
iObject.IncrementPosition = IncrementPosition	=	Установить свойство (*)
IncrementPosition = iObject.GetIncrementPosition()	=	Получить свойство (**)
iObject.SetIncrementPosition (IncrementPosition)		Установить свойство (**)

Синтаксис COM:

iObject->get_IncrementPosition (&IncrementPosition)	Получить свойство
iObject->put_IncrementPosition (IncrementPosition)	Установить свойство

Значения свойства:

TRUE	- номер позиции объекта должен быть на единицу больше номера позиции предшествующего ему объекта,
FALSE	- номер позиции объекта совпадает с номером позиции предыдущего объекта.

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

NestedBlock – Номер блока вложенных разделов

Интерфейс...

Тип данных: long

Синтаксис Automation:

NestedBlock = iObject.NestedBlock	=	Получить свойство (*)
iObject.NestedBlock = NestedBlock	=	Установить свойство (*)
NestedBlock = iObject.GetNestedBlock()	=	Получить свойство (**)
iObject.SetNestedBlock (NestedBlock)		Установить свойство (**)

Синтаксис COM:

iObject->get_NestedBlock (&NestedBlock)	Получить свойство
Object->put_NestedBlock (NestedBlock)	Установить свойство

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

NestedSection – Номер вложенного раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

NestedSection = iObject.NestedSection	=	Получить свойство (*)
iObject.NestedSection = NestedSection	=	Установить свойство (*)
iObject.GetNestedSection()		Получить свойство (**)

iObject.SetNestedSection (NestedSection)	Установить свойство (**)

Синтаксис COM:

iObject->get_NestedSection (&NestedSection)	Получить свойство
iObject->put_NestedSection (NestedSection)	Установить свойство

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

ObjectType – Тип объекта

Интерфейс...

Тип данных: из перечисления ksSpecificationObjectTypeEnum

Синтаксис Automation:

ObjectType = iObject.ObjectType	Получить свойство (*)
ObjectType = iObject.GetObjectType()	Получить свойство (**)

Синтаксис COM:

iObject->get_ObjectType (&ObjectType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Section – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

Section = iObject.Section	Получить свойство (*)
Section = iObject.GetSection()	Получить свойство (**)

Синтаксис COM:

iObject->get_Section (&Section)	Получить свойство

Примечание:

1. При создании базового или вспомогательного объекта спецификации номер раздела передается в метод Add. Переместить уже созданные объекты из одного раздела в другой нельзя.
2. Свойство доступно только для чтения.

State – Состояние объекта

Интерфейс...

Тип данных: из перечисления ksSpecificationObjectStateEnum

Синтаксис Automation:

State = iObject.State	Получить свойство (*)
State = iObject.GetState()	Получить свойство (**)

Синтаксис COM:

iObject->get_State (&State)	Получить свойство

Значения свойства:

ssIndependent	- объект создан в данном документе,
ssFromInsert	- объект пришел из вставки другого документа (фрагмента, детали или подсборки),
ssEdit	- объект из вставки другого документа, но редактировался в данном документе.

Примечание:

Свойство доступно только для чтения.

Subsection – Номер подраздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

Subsection = iObject.Subsection	Получить свойство (*)
iObject.Subsection = Subsection	Установить свойство (*)
Subsection = iObject.GetSubsection()	Получить свойство (**)
iObject.SetSubsection (Subsection)	Установить свойство (**)

Синтаксис COM:

iObject->get_Subsection (&Subsection)	Получить свойство
iObject->put_Subsection (Subsection)	Установить свойство

Примечание:

1. Свойство доступно только для базового и вспомогательного объекта спецификации.
2. Изменение этого свойства вступит в силу после вызова метода ISpecificationObject::Update.

UniqueNumber – Уникальный номер, присваивается при создании

Интерфейс...

Тип данных: double

Синтаксис Automation:

UniqueNumber iObject.UniqueNumber	=	Получить свойство (*)
Objects iObject.GetBaseObjects()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_UniqueNumber (&UniqueNumber)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ISpecificationObject – методы

Delete – Удалить объект

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete([out, retval] BOOL* Result);

Примечание:

Метод позволяет удалить базовый или вспомогательный объект спецификации из документа и из коллекции.

Edit – Редактировать объект

Интерфейс...

Синтаксис Automation:

BOOL Edit();

Синтаксис COM:

HRESULT Edit ([out, retval] BOOL* Result);

Примечание:

Метод позволяет запустить процесс визуального редактирования для базового или вспомогательного объекта спецификации.

Update – Обновить данные

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update ([out, retval] BOOL* Result);

Примечание:

После изменения параметров объекта спецификации требуется вызвать данный метод, чтобы новые данные попали в модельный объект спецификации. Если объект спецификации еще не создан, метод Update создает объект спецификации в документе.

Для ускорения работы функции при массовом изменении спецификации рекомендуется воспользоваться методом ISpecificationDescription::NeedRebuild.

Интерфейс ISpecificationSectionStyle

Интерфейс стиля раздела спецификации.

Иерархия:

IKompasAPIObject

ISpecificationSectionStyle

Описание:

В спецификации все объекты располагаются в своем разделе, если включено разбиение на разделы ISpecificationStyle::SectionOn. Разделы располагаются в порядке возрастания или убывания их номеров ISpecificationSectionStyle::Number, в зависимости от значения свойства ISpecificationStyle::SortSectionDown. Раздел автоматически добавляется в спецификацию, если в нем создается базовый объект спецификации или комментарий, и автоматически удаляется из спецификации при удалении последнего объекта спецификации.

Примечание:

Данный интерфейс можно получить от интерфейса стиля раздела спецификации ISpecificationSectionStyles, ISpecificationSectionStyles::Item.

ISpecificationSectionStyle – свойства

AdditionalBlocks – Блоки вложенных разделов

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockStyles коллекции стилей блоков дополнительных разделов.

Синтаксис Automation:

AdditionalBlocks = iObject.AdditionalBlocks	=	Получить свойство (*)
AdditionalBlocks = iObject.GetAdditionalBlocks()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlocks (&AdditionalBlocks)	Получить свойство

Примечание:

Свойство доступно только для чтения.

AdditionalColumns – Дополнительные колонки спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnStyles коллекции стилей колонок спецификации

Синтаксис Automation:

AdditionalColumns = iObject.AdditionalColumns	=	Получить свойство (*)
AdditionalColumns = iObject.GetAdditionalColumns()	=	Получить свойство (**)

Синтаксис COM:

iObject- >get_AdditionalColumns(&AdditionalColumns)	Получить свойство

Примечание:

1. Настройки стиля колонок для данного раздела (могут отличаться от умолчательных).
2. Свойство доступно только для чтения.

Columns – Колонки спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnStyles коллекции стилей колонок спецификации.

Синтаксис Automation:

Columns = iObject.Columns	Получить свойство (*)
Columns = iObject.GetColumns()	Получить свойство (**)

Синтаксис COM:

iObject->get_Columns (&Columns)	Получить свойство

Примечание:

1. Настройки стиля колонок для данного раздела (могут отличаться от умолчательных).
2. Свойство доступно только для чтения.

FillDataFromStamp – Чтение данных из основной надписи

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FillDataFromStamp = iObject.FillDataFromStamp		Получить свойство (*)
FillDataFromStamp = iObject.GetFillDataFromStamp()		Получить свойство (**)

Синтаксис COM:

iObject->get_FillDataFromStamp (&FillDataFromStamp)	Получить свойство

Значения свойства:

TRUE	- заполнение колонок смешанным способом - вручную с возможностью чтения информации из основной надписи связанных с объектами спецификации чертежей,
FALSE	- заполнение колонок - вручную.

Примечание:

Свойство доступно только для чтения.

Name – Имя раздела

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

Name = iObject.Name	Получить свойство (*)
Name = iObject.GetName()	Получить свойство (**)

Синтаксис COM:

iObject->get_Name (&Name)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Number – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер раздела в данном стиле спецификации.
2. Разделы в спецификации располагаются в порядке возрастания или убывания их номеров Number, в зависимости от значения свойства ISpecificationStyle::SortSectionDown.
3. Свойство доступно только для чтения.

SortColumnNumber – Номер колонки, в которой необходимо проводить сортировку

Интерфейс...

Тип данных: long

Синтаксис Automation:

SortColumnNumber = iObject.SortColumnNumber		Получить свойство (*)
SortColumnNumber = iObject.GetSortColumnNumber()		Получить свойство (**)

Синтаксис COM:

iObject->get_SortColumnNumber (&SortColumnNumber)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер колонки для колонок с типом SortColumnType, по которой будет производится сортировка объектов в разделе. Иногда однородная (но не одинаковая) информация должна храниться в нескольких колонках и по этой причине они имеют одинаковый тип ColumnType.

   Чтобы отличать такие колонки, вместе с типом задается еще и номер, отличающий колонку этого типа от других колонок такого же типа.
2. Свойство доступно только для чтения.

SortColumnNumberEx – Номер колонки, в которой проводить сортировку

Интерфейс...

Тип данных: long

Синтаксис Automation:

SortColumnNumberEx = Object.SortColumnNumberEx(Index)		Получить свойство (*)
SortColumnNumberEx = Object.GetSortColumnNumberEx (Index)		Получить свойство (**)

Синтаксис COM:

Object.get_SortColumnNumberEx(Index, &SortColumnNumberEx)	Получить свойство

Входные парметры:

Index	- индекс колонки.

Примечание:

Свойство доступно только для чтения.

SortColumnType – Общий тип колонки, в которой необходимо проводить сортировку

Интерфейс...

Тип данных: из перечисления ksSpecificationColumnTypeEnum

Синтаксис Automation:

SortColumnType = iObject.SortColumnType	=	Получить свойство (*)
SortColumnType = iObject.GetSortColumnType()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SortColumnType (&SortColumnType)	Получить свойство

Примечание:

1. Используется при включенной опции ISpecificationTuningSection::SortObjects.
2. Свойство доступно только для чтения.

SortColumnTypeEx – Общий тип колонки, в которой необходимо проводить сортировку

Интерфейс...

Тип данных: из перечисления ksSpecificationColumnTypeEnum

Синтаксис Automation:

SortColumnTypeEx = Object.SortColumnTypeEx(Index)	=	Получить свойство (*)
SortColumnTypeEx = Object.GetSortColumnTypeEx(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SortColumnTypeEx(Index, &SortColumnTypeEx)	Получить свойство

Входные параметры:

Index	- индекс колонки.

Примечание:

Свойство доступно только для чтения.

SortLevelsCount – Количество уровней сортировки

Интерфейс...

Тип данных:long

Синтаксис Automation:

SortLevelsCount = Object.SortLevelsCount	=	Получить свойство (*)
SortLevelsCount = Object.GetSortLevelsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SortLevelsCount(&SortLevelsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SortType – Тип сортировки объектов спецификации в разделе

Интерфейс...

Тип данных: из перечисления ksSortTypeEnum

Синтаксис Automation:

SortType = iObject.SortType	Получить свойство (*)
SortType = iObject.GetSortType()	Получить свойство (**)

Синтаксис COM:

iObject->get_SortType (&SortType)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SortTypeEx – Тип сортировки

Интерфейс...

Тип данных: из перечисления ksSortTypeEnum

Синтаксис Automation:

SortTypeEx = Object.SortTypeEx(Index)	Получить свойство (*)
SortTypeEx = Object.GetSortTypeEx(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_SortTypeEx(Index, &SortTypeEx)	Получить свойство

Входные параметры:

Index	- индекс колонки.

Примечание:

Свойство доступно только для чтения.

Интерфейс ISpecificationStyle

Интерфейс стиля спецификации.

Иерархия:

IKompasAPIObject

ISpecificationStyle

Описание:

Описывает стиль спецификации. Стили спецификаций описаны в библиотеках стилей спецификаций (файлы библиотек стилей спецификаций имеют расширение lyt).

Данный интерфейс отражает содержимое стиля в библиотеке стилей на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.

Примечание:

Данный интерфейс может быть получен от интерфейса ISpecificationDescription с помощью свойства ISpecificationDescription::SpecificationStyle.

ISpecificationStyle – свойства

AdditionalBlocks – Блоки дополнительных разделов

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockStyles коллекции стилей блоков дополнительных разделов

Синтаксис Automation:

AdditionalBlocks = iObject.AdditionalBlocks	=	Получить свойство (*)
AdditionalBlocks = iObject.GetAdditionalBlocks()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlocks (&AdditionalBlocks)	Получить свойство

Примечание:

Свойство доступно только для чтения.

AdditionalColumns – дополнительные колонки спецификации – умолчательные значения

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnStyles коллекции стилей колонок спецификации

Синтаксис Automation:

AdditionalColumns = iObject.AdditionalColumns	=	Получить свойство (*)
AdditionalColumns = iObject.GetAdditionalColumns()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalColumns (&AdditionalColumns)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Columns – Колонки спецификации – умолчательные значения

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnStyles коллекции стилей колонок спецификации

Синтаксис Automation:

Columns = iObject.Columns	Получить свойство (*)
Columns = iObject.GetColumns()	Получить свойство (**)

Синтаксис COM:

iObject->get_Columns (&Columns)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Format – Формат листа

Интерфейс...

Тип данных: указатель на интерфейс ISheetFormat формата листа спецификации

Синтаксис Automation:

Format = iObject.Format	Получить свойство (*)
Format = iObject.GetFormat()	Получить свойство (**)

Синтаксис COM:

iObject->get_Format(&Format)	Получить свойство

Примечание:

1. Данное свойство определяет формат листа бумаги, на котором будет размещаться спецификация.
2. Свойство доступно только для чтения.

LayoutName1 – Имя файла библиотеки оформлений для первого листа

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

LayoutName1 iObject.LayoutName1	=	Получить свойство (*)
LayoutName1 iObject.GetLayoutName1()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LayoutName1 (&LayoutName1)	Получить свойство

Примечание:

1. Позволяет получить имя файла библиотеки оформлений для первого листа.
2. Свойство доступно только для чтения.

LayoutName2 – Имя файла библиотеки оформлений для последующих листов

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

LayoutName2 iObject.LayoutName2	=	Получить свойство (*)
LayoutName2 iObject.GetLayoutName2()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_LayoutName2 (&LayoutName2)	Получить свойство

Примечание:

1. Позволяет получить имя файла библиотеки оформлений для последующих листов.
2. Свойство доступно только для чтения.

PerformanceCountInBlock – Количество исполнений в блоке

Интерфейс...

Тип данных: long

Синтаксис Automation:

PerformanceCountInBlock Object.PerformanceCountInBlock(PVal)	=	Получить свойство (*)
PerformanceCountInBlock Object.GetPerformanceCountInBlock(PVal)	=	Получить свойство (**)

Синтаксис COM:

Object.get_PerformanceCountIn Block(PVal, &PerformanceCountInBlock)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Sections – Разделы

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationSectionStyles коллекции стилей разделов спецификации

Синтаксис Automation:

Sections = iObject.Sections	Получить свойство (*)
Sections = iObject.GetSections()	Получить свойство (**)

Синтаксис COM:

iObject->get_Sections (&Sections)	Получить свойство

Примечание:

Свойство доступно только для чтения.

SectionOn – Деление на разделы

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SectionOn = iObject.SectionOn	Получить свойство (*)
SectionOn = iObject.GetSectionOn()	Получить свойство (**)

Синтаксис COM:

iObject->get_SectionOn (&SectionOn)	Получить свойство

Значения свойства:

TRUE	- разбиение на разделы включено, все объекты будут располагаться в своем разделе,
FALSE	- разбиение на разделы выключено.

Примечание:

1. Позволяет получить признак разбиения спецификации на разделы.
2. Свойство доступно только для чтения.

SortSectionDown – Сортировка разделов по убыванию

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SortSectionDown = iObject.SortSectionDown	=	Получить свойство (*)
SortSectionDown = iObject.GetSortSectionDown()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_SortSectionDown (&SortSectionDown)	Получить свойство

Значения свойства:

TRUE	- сортировка разделов по убыванию включена, разделы будут располагаться внутри спецификации в порядке убывания их порядковых номеров,
FALSE	- сортировка разделов по убыванию выключена.

Примечание:

1. Позволяет получить признак сортировки разделов по убыванию.
2. Свойство доступно только для чтения.

SpecificationTuning – Настройки спецификации

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationTuning настроек спецификации

Синтаксис Automation:

SpecificationTuning = iObject.SpecificationTuning	=	Получить свойство (*)

SpecificationTuning = iObject.GetSpecificationTuning()	Получить свойство (**)

Синтаксис COM:

iObject->get_SpecificationTuning (&SpecificationTuning)	Получить свойство

Примечание:

Свойство доступно только для чтения. Полученные настройки будут соответствовать настройкам, описанным для данного стиля в библиотеке стилей. Для получения настроек описания спецификации в документе надо брать настройки непосредственно у описания ISpecificationDescription::SpecificationTuning, т.к. они могут отличатся от настроек стиля.

StyleID1 – Номер стиля в библиотеке оформлений для первого листа

Интерфейс...

Тип данных: long

Синтаксис Automation:

StyleID1 = iObject.StyleID1	Получить свойство (*)
StyleID1 = iObject.GetStyleID1()	Получить свойство (**)

Синтаксис COM:

iObject->get_StyleID1 (&StyleID1)	Получить свойство

Примечание:

1. Позволяет получить номер стиля в библиотеке оформлений для первого листа.
2. Свойство доступно только для чтения.

StyleID2 – Номер стиля в библиотеке оформлений для последующих листов

Интерфейс...

Тип данных: long

Синтаксис Automation:

StyleID2 = iObject.StyleID2	Получить свойство (*)
StyleID2 = iObject.GetStyleID2()	Получить свойство (**)

Синтаксис COM:

iObject->get_StyleID2 (&StyleID2)	Получить свойство

Примечание:

1. Позволяет получить номер стиля в библиотеке оформлений для последующих листов.
2. Свойство доступно только для чтения.

Variant – Вариант оформления спецификации

Интерфейс...

Тип данных: из перечисления ksSpecificationVariantEnum

Синтаксис Automation:

Variant = iObject.Variant	Получить свойство (*)
Variant = iObject.GetVariant()	Получить свойство (**)

Синтаксис COM:

iObject->get_Variant (&Variant)	Получить свойство

Примечание:

1. Позволяет получить способ представления информации об исполнениях изделия.
2. Доступны только варианты "Простая", "Вариант А" или "Вариант Б."
3. Свойство доступно только для чтения.

Интерфейс ISpecificationSubsection

Интерфейс подраздела спецификации.

Иерархия:

IKompasAPIObject

ISpecificationSubsection

Описание:

Данный интерфейс позволяет получить доступ к подразделу спецификации. Группы объектов внутри разделов называются подразделами. Подразделы, как и разделы, являются компонентом стиля спецификации. Их количество, названия и порядок внутри каждого раздела формируются при настройке стиля. Названия подразделов не отображаются в бланке спецификации. Они служат лишь для удобства выбора подраздела. Подразделы располагаются в разделе в порядке возрастания номеров.

Примечание:

Данный интерфейс можно получить у коллекции ISpecificationSubsections.

ISpecificationSubsection – свойства

Name – Имя подраздела

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

```
Name = iObject.Name                    Получить свойство (* )
Name = iObject.GetName()               Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Name (&Name)              Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Number – Номер подраздела

Интерфейс...

Тип данных: long.

Синтаксис Automation:

```
Number = iObject.Number                Получить свойство (* )
Number = iObject.GetNumber()           Получить свойство (**)
```

Синтаксис COM:

```
iObject->get_Number                    Получить свойство
(&Number)
```

Примечание:

1. Данное свойство определяет уникальный номер подраздела в данном разделе спецификации.
2. Свойство доступно только для чтения.

ISpecificationSubsection – методы

Change – Изменить параметры подраздела

Интерфейс...

Синтаксис Automation:

BOOL Change (BSTR Name, short Number);

Синтаксис COM:

HRESULT Change ([in] BSTR Name,
[in] short Number,
[out, retval] VARIANT_BOOL* pVal);

Входные параметры:

Name - новое имя подраздела,
Number - новый номер подраздела.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Delete – Удалить подраздел

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Интерфейс ISpecificationTuning

Интерфейс настроек спецификации.

Иерархия:

IKompasAPIObject

ISpecificationTuning

Описание:

Данный интерфейс позволяет получить доступ к настройкам стиля спецификации ISpecificationStyle или настройкам описания спецификации ISpecificationDescription.

Если настройки взяты у стиля спецификации, то они являются умолчательными для данного стиля и изменять их нельзя. В этом случае данный интерфейс отражает содержимое настроек стиля в библиотеке стилей на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.

Если настройки взяты у описания спецификации, то они отображают и отслеживают текущее состояние описания спецификации. Эти настройки можно изменять и они могут отличаться от умолчательных настроек стиля спецификации, по которому создано данное описание спецификации.

После изменения настроек (для того чтобы эти изменения реально отобразились в документе) необходимо вызвать метод ISpecificationTuning::Update.

Примечание:

Данный интерфейс можно получить от интерфейса описания спецификации документа ISpecificationDescription, используя свойство ISpecificationDescription::SpecificationTuning или от интерфейса стиля спецификации ISpecificationStyle с помощью свойства ISpecificationStyle::SpecificationTuning.

ISpecificationTuning – свойства

AdditionalBlocks – Блоки дополнительных разделов

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockTunings коллекции настроек блоков дополнительных разделов спецификации

Синтаксис Automation:

AdditionalBlocks = iObject.AdditionalBlocks	=	Получить свойство (*)
AdditionalBlocks = iObject.GetAdditionalBlocks()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlocks (&AdditionalBlocks)	Получить свойство

Примечание:

1. Позволяет получить параметры настроек блоков дополнительных разделов спецификации.
2. Свойство доступно только для чтения.

AdditionalBlockTextStyleFirst – Стиль текста заголовка дополнительных блоков – первая строка

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста.

Синтаксис Automation:

AdditionalBlockTextStyleFirst = iObject.AdditionalBlockTextStyleFirst	=	Получить свойство (*)
AdditionalBlockTextStyleFirst = iObject.GetAdditionalBlockTextStyleFirst()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlockTextStyleFirst (&AdditionalBlockTextStyleFirst)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения первой строки в названиях дополнительных блоков разделов спецификации.
2. Свойство доступно только для чтения.

AdditionalBlockTextStyleNext – Стиль текста заголовка дополнительных блоков – последующие строки

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста.

Синтаксис Automation:

AdditionalBlockTextStyleNext = iObject.AdditionalBlockTextStyleNext	Получить свойство (*)
AdditionalBlockTextStyleNext = iObject.GetAdditionalBlockTextStyleNext()	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlockTextStyleNext (&AdditionalBlockTextStyleNext)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения второй и последующих строк в многострочных названиях дополнительных блоков разделов спецификации.
2. Свойство доступно только для чтения.

BlockCount – Количество блоков

Интерфейс...

Тип данных: long

Синтаксис Automation:

BlockCount = iObject.BlockCount	Получить свойство (*)

BlockCount	=	Получить свойство
iObject.GetBlockCount()		(**)

Синтаксис COM:

iObject->get_BlockCount (&BlockCount)	Получить свойство

Примечание:

1. Позволяет получить количество блоков исполнений в спецификации. Имеет смысл только при настройке групповых спецификаций.
2. Свойство доступно только для чтения.

BlockOnNewPage – Располагать блок на новой странице

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

BlockOnNewPage	=	Получить свойство
iObject.BlockOnNewPage		(*)
iObject.BlockOnNewPage	=	Установить свойство
BlockOnNewPage		(*)
BlockOnNewPage	=	Получить свойство
iObject.GetBlockOnNewPage()		(**)
iObject.SetBlockOnNewPage		Установить свойство
(BlockOnNewPage)		(**)

Синтаксис COM:

iObject->get_BlockOnNewPage (&BlockOnNewPage)	Получить свойство
iObject->put_BlockOnNewPage (BlockOnNewPage)	Установить свойство

Значения свойства:

TRUE	- располагать блок на новой странице,
FALSE	- располагать блок сразу за предыдущим.

Примечание:

Позволяет указать, требуется ли располагать начала всех блоков на новых листах спецификации.

CalculatePosition – Режим расчета позиций

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CalculatePosition = iObject.CalculatePosition	=	Получить свойство (*)
iObject.CalculatePosition = CalculatePosition	=	Установить свойство (*)
CalculatePosition = iObject.GetCalculatePosition()	=	Получить свойство (**)
iObject.SetCalculatePosition (CalculatePosition)		Установить свойство (**)

Синтаксис COM:

iObject->get_CalculatePosition (&CalculatePosition)	Получить свойство
iObject->put_CalculatePosition (CalculatePosition)	Установить свойство

Значения свойства:

TRUE	- расчет позиций автоматически (после каждой сортировки) не происходит,
FALSE	- расчет позиций происходит автоматически.

Примечание:

Позволяет включить или отключить расчет позиций (то есть присвоение объектам новых номеров позиций после того, как нумерация нарушилась в результате автоматической сортировки или удаления объектов).

CalculateZone – Режим расчета зон

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CalculateZone = iObject.CalculateZone	=	Получить свойство (*)
iObject.CalculateZone = CalculateZone	=	Установить свойство (*)
CalculateZone = iObject.GetCalculateZone()	=	Получить свойство (**)
iObject.SetCalculateZone (CalculateZone)		Установить свойство (**)

Синтаксис COM:

iObject->get_CalculateZone (&CalculateZone)	Получить свойство
iObject->put_CalculateZone (CalculateZone)	Установить свойство

Значения свойства:

TRUE	- при автоматической простановке номеров позиций происходит обновление номеров зон чертежа,
FALSE	- зоны не обновляются.

Примечание:

Позволяет включить или отключить передачу в спецификацию обозначений зон сборочного чертежа, в которых находятся позиционные линии-выноски объектов. Если передача разрешена, она происходит при синхронизации чертежа со спецификацией.

CopySpcObjectOnCopyGeometry – Копировать объекты спецификации при копировании геометрии

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CopySpcObjectOnCopyGeometry = Object.CopySpcObjectOnCopyGeometry	Получить свойство (*)
Object.CopySpcObjectOnCopyGeometry = CopySpcObjectOnCopyGeometry	Установить свойство (*)
CopySpcObjectOnCopyGeometry = Object.GetCopySpcObjectOnCopyGeometry()	Получить свойство (**)
Object.SetCopySpcObjectOnCopyGeometry(CopySpcObjectOnCopyGeometry)	Установить свойство (**)

Синтаксис COM:

Object.get_CopySpcObjectOnCopyGeometry(&CopySpcObjectOnCopyGeometry)	Получить свойство
Object.put_CopySpcObjectOnCopyGeometry(CopySpcObjectOnCopyGeometry)	Установить свойство

DeleteGeometry – Режим удаления геометрии при удалении объекта спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DeleteGeometry = iObject.DeleteGeometry	=	Получить свойство (*)
iObject.DeleteGeometry = DeleteGeometry	=	Установить свойство (*)
DeleteGeometry = iObject.GetDeleteGeometry()	=	Получить свойство (**)
iObject.SetDeleteGeometry (DeleteGeometry)		Установить свойство (**)

Синтаксис COM:

iObject->get_DeleteGeometry (&DeleteGeometry)	Получить свойство
iObject->put_DeleteGeometry (DeleteGeometry)	Установить свойство

Значения свойства:

TRUE	- в результате удаления объекта спецификации из чертежа должны удаляться входящие в состав этого объекта графические объекты,
FALSE	- в результате удаления объекта спецификации из чертежа входящие в состав этого объекта графические объекты удаляться не будут.

Примечание:

Управляет удалением графических объектов, входящих в состав удаляемых объектов спецификации.

DeleteSpcObjectOnDeleteGeometry – Режим удаления спецификации при удалении геометрии

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- в результате удаления графического объекта из чертежа должны удаляться соответствующие объекты спецификации,
FALSE	- в результате удаления графического объекта из чертежа соответствующие объекты спецификации удаляться не будут.

Синтаксис Automation:

DeleteSpcObjectOnDeleteGeometry = iObject.DeleteSpcObjectOnDeleteGeometry	Получить свойство (*)
iObject.DeleteSpcObjectOnDeleteGeometry = DeleteSpcObjectOnDeleteGeometry	Установить свойство (*)
DeleteSpcObjectOnDeleteGeometry = iObject.GetDeleteSpcObjectOnDeleteGeometry()	Получить свойство (**)
iObject.SetDeleteSpcObjectOnDeleteGeometry (DeleteSpcObjectOnDeleteGeometry)	Установить свойство (**)

Синтаксис COM:

iObject->get_DeleteSpcObjectOnDeleteGeometry (&DeleteSpcObjectOnDeleteGeometry)	Получить свойство
iObject->put_DeleteSpcObjectOnDeleteGeometry (DeleteSpcObjectOnDeleteGeometry)	Установить свойство

Примечание:

Управляет удалением объектов спецификации при удалении графических объектов.

DisableEmptyString – Режим вывода пустых строк вокруг заголовка раздела

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableEmptyString = iObject.DisableEmptyString		Получить свойство (*)
iObject.DisableEmptyString = DisableEmptyString		Установить свойство (*)
DisableEmptyString = iObject.GetDisableEmptyString()		Получить свойство (**)
iObject.SetDisableEmptyString (DisableEmptyString)		Установить свойство (**)

Синтаксис COM:

iObject->get_DisableEmptyString (&DisableEmptyString)	Получить свойство
iObject->put_DisableEmptyString (DisableEmptyString)	Установить свойство

Значения свойства:

TRUE	- не показывать пустые строки, обрамляющие название разделов спецификации,
FALSE	- показывать пустые строки, обрамляющие название разделов спецификации.

Примечание:

Позволяет включать и отключать показ пустых строк, обрамляющих название разделов спецификации.

DisableEmptyBlockString – Режим вывода пустых строк вокруг начала блока

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableEmptyBlockString = iObject.DisableEmptyBlockString	Получить свойство (*)
iObject.DisableEmptyBlockString = DisableEmptyBlockString	Установить свойство (*)
DisableEmptyBlockString = iObject.GetDisableEmptyBlockString()	Получить свойство (**)
iObject.SetDisableEmptyBlockString (DisableEmptyBlockString)	Установить свойство (**)

Синтаксис COM:

iObject->get_DisableEmptyBlockString (&DisableEmptyBlockString)	Получить свойство
iObject->put_DisableEmptyBlockString (DisableEmptyBlockString)	Установить свойство

Значения свойства:

TRUE	- не показывать пустые строки, обрамляющие название начала блоков спецификации,
FALSE	- показывать пустые строки, обрамляющие название начала блоков спецификации.

Примечание:

Позволяет включать и отключать показ пустых строк, обрамляющих название начала блоков спецификации.

DisableAdditionalBlockEmptyStrings – Запретить вывод пустых строк вокруг начала дополнительного блока

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableAdditionalBlockEmptyStrings = iObject.DisableAdditionalBlockEmptyStrings	Получить свойство (*)
iObject.DisableAdditionalBlockEmptyStrings = DisableAdditionalBlockEmptyStrings	Установить свойство (*)

DisableAdditionalBlockEmptyStrings = iObject.GetDisableAdditionalBlockEmptyStrings()	Получить свойство (**)
iObject.SetDisableAdditionalBlockEmptyStrings (DisableAdditionalBlockEmptyStrings)	Установить свойство (**)

Синтаксис COM:

iObject->get_DisableAdditionalBlockEmptyStrings (&DisableAdditionalBlockEmptyStrings)	Получить свойство
iObject->put_DisableAdditionalBlockEmptyStrings (DisableAdditionalBlockEmptyStrings)	Установить свойство

Значения свойства:

TRUE	- не показывать пустые строки, обрамляющие название начала дополнительных блоков спецификации,
FALSE	- показывать пустые строки, обрамляющие название начала дополнительных блоков спецификации.

Примечание:

Позволяет включать и отключать показ пустых строк, обрамляющих название начала дополнительных блоков спецификации.

DisableNestingBlockEmptyStrings – Запретить вывод пустых строк вокруг начала вложенного блока

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DisableNestingBlockEmptyStrings = iObject.DisableNestingBlockEmptyStrings	Получить свойство (*)
iObject.DisableNestingBlockEmptyStrings = DisableNestingBlockEmptyStrings	Установить свойство (*)

DisableNestingBlockEmptyStrings = iObject.GetDisableNestingBlockEmptyStrings()	Получить свойство (**)
iObject.SetDisableNestingBlockEmptyStrings (DisableNestingBlockEmptyStrings)	Установить свойство (**)

Синтаксис COM:

iObject->get_DisableNestingBlockEmptyStrings (&DisableNestingBlockEmptyStrings)	Получить свойство
iObject->put_DisableNestingBlockEmptyStrings (DisableNestingBlockEmptyStrings)	Установить свойство

Значения свойства:

TRUE	- не показывать пустые строки, обрамляющие название начала вложенных блоков спецификации,
FALSE	- показывать пустые строки, обрамляющие название начала вложенных блоков спецификации.

Примечание:

Позволяет включать и отключать показ пустых строк, обрамляющих название начала вложенных блоков спецификации.

DrawBottomUp – Строить снизу вверх

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

DrawBottomUp = iObject.DrawBottomUp	=	Получить свойство (*)
iObject.DrawBottomUp = DrawBottomUp	=	Установить свойство (*)
DrawBottomUp = iObject.GetDrawBottomUp()	=	Получить свойство (**)
iObject.SetDrawBottomUp (DrawBottomUp)		Установить свойство (**)

Синтаксис COM:

iObject->get_DrawBottomUp (&DrawBottomUp)	Получить свойство
iObject->put_DrawBottomUp (DrawBottomUp)	Установить свойство

Значения свойства:

TRUE	- строить снизу вверх,
FALSE	- строить сверху вниз.

Примечание:

Позволяет управлять порядком следования разделов и объектов в них. Если значение свойства - TRUE, разделы располагаются в порядке, обратном указанному в стиле спецификации, и порядок сортировки объектов также становится «обратным». Такой порядок расположения предписывается некоторыми СТП при заполнении спецификации, размещаемой на чертеже.

InitialPosition – Начальная позиция

Интерфейс...

Тип данных: long

Синтаксис Automation:

InitialPosition = iObject.InitialPosition	=	Получить свойство (*)
iObject.InitialPosition = InitialPosition	=	Установить свойство (*)
InitialPosition = iObject.GetInitialPosition()	=	Получить свойство (**)
iObject.SetInitialPosition (InitialPosition)		Установить свойство (**)

Синтаксис COM:

iObject->get_InitialPosition (&InitialPosition)	Получить свойство
iObject->put_InitialPosition (InitialPosition)	Установить свойство

Примечание:

Позволяет получить или указать номер позиции, который должен иметь первый объект спецификации.

InsertDash – Настройка начала блока: вставлять тире

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InsertDash = iObject.InsertDash	Получить свойство (*)
iObject.InsertDash = InsertDash	Установить свойство (*)
InsertDash = iObject.GetInsertDash()	Получить свойство (**)
iObject.SetInsertDash(InsertDash)	Установить свойство (**)

Синтаксис COM:

iObject->get_InsertDash(&InsertDash)	Получить свойство
iObject->put_InsertDash(InsertDash)	Установить свойство

Значения свойства:

TRUE	- вставлять тире перед числом,
FALSE	- не вставлять тире перед числом.

Примечание:

Позволяет настроить формат номеров исполнений специфицируемого изделия (номеров исполнений, располагающихся в «шапке» спецификации).

Настройка номеров исполнений имеет смысл только для групповых спецификаций.

InsertNull – Настройка начала блока: вставлять нули перед числом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

InsertNull = iObject.InsertNull	Получить свойство (*)
iObject.InsertNull = InsertNull	Установить свойство (*)
InsertNull = iObject.GetInsertNull()	Получить свойство (**)
iObject.SetInsertNull (InsertNull)	Установить свойство (**)

Синтаксис COM:

iObject->get_InsertNull (&InsertNull)	Получить свойство
iObject->put_InsertNull (InsertNull)	Установить свойство

Значения свойства:

TRUE	- вставлять нули перед числом,
FALSE	- не вставлять нули перед числом.

Примечание:

Позволяет настроить формат номеров исполнений специфицируемого изделия (номеров исполнений, располагающихся в «шапке» спецификации).

Настройка номеров исполнений имеет смысл только для групповых спецификаций.

LinkType – Режим связки сборки или чертежа со спецификацией

Интерфейс...

Тип данных: из перечисления ksSpecificationLinkTypeEnum

Синтаксис Automation:

LinkType = iObject.LinkType	Получить свойство (*)
iObject.LinkType = LinkType	Установить свойство (*)
LinkType = iObject.GetLinkType()	Получить свойство (**)
iObject.SetLinkType (LinkType)	Установить свойство (**)

Синтаксис COM:

iObject->get_LinkType (&LinkType)	Получить свойство
iObject->put_LinkType (LinkType)	Установить свойство

Значения свойства:

slNone	- информация из подключенного сборочного чертежа не будет передаваться в спецификацию, однако чертеж будет по-прежнему подключен, и связь можно будет включить в любой момент,

slOnlyObjects	- при сохранении подключенного чертежа, в который внесены изменения, касающиеся спецификации (созданы или отредактированы объекты спецификации), эти изменения будут переданы в спецификацию.
slWithPosition Calculate	- после передачи изменений из чертежа в спецификацию в ней будет автоматически произведен расчет номеров позиций (в соответствии с последовательностью сортировки объектов), и новые номера позиций будут переданы в чертеж.

Примечание:

Позволяет включать и блокировать связь, подключенных друг к другу документов - спецификации и модели (сборочного чертежа), и устанавливать тип этой связи.

NestingBlockTextStyleFirst – Стиль текста заголовка вложенных блоков – первая строка

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

NestingBlockTextStyleFirst = iObject.NestingBlockTextStyleFirst	Получить свойство (*)
NestingBlockTextStyleFirst = iObject.GetNestingBlockTextStyleFirst()	Получить свойство (**)

Синтаксис COM:

iObject->get_NestingBlockTextStyleFirst (&NestingBlockTextStyleFirst)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения первой строки в названиях вложенных блоков разделов спецификации.
2. Свойство доступно только для чтения.

NestingBlockTextStyleNext – Стиль текста заголовка вложенных блоков – последующие строки

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

NestingBlockTextStyleNext = iObject.NestingBlockTextStyleNext	Получить свойство (*)
NestingBlockTextStyleNext = iObject.GetNestingBlockTextStyleNext()	Получить свойство (**)

Синтаксис COM:

iObject->get_NestingBlockTextStyleNext (&NestingBlockTextStyleNext)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения второй и последующих строк в многострочных названиях вложенных блоков разделов спецификации.
2. Свойство доступно только для чтения.

ObjectTextStyle – Стиль текста объекта спецификации

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

ObjectTextStyle = iObject.ObjectTextStyle	Получить свойство (*)
ObjectTextStyle = iObject.GetObjectTextStyle()	Получить свойство (**)

Синтаксис COM:

iObject->get_ObjectTextStyle (&ObjectTextStyle)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта текстовой части объектов спецификации, отличающиеся от параметров шрифта в оформлении. Имеет смысл, если свойство UserTextStyle имеет значение TRUE.
2. Свойство доступно только для чтения.

PerformanceBlockTextStyleFirst – Стиль текста заголовка блока исполнений – первая строка

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

PerformanceBlockTextStyleFirst = iObject.PerformanceBlockTextStyleFirst	Получить свойство (*)
PerformanceBlockTextStyleFirst = iObject.GetPerformanceBlockTextStyleFirst()	Получить свойство (**)

Синтаксис COM:

iObject->get_PerformanceBlockTextStyleFirst (&PerformanceBlockTextStyleFirst)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения первой строки в названиях блоков исполнений спецификации.
2. Свойство доступно только для чтения.

PerformanceBlockTextStyleNext – Стиль текста заголовка блока исполнений – последующие строки

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

PerformanceBlockTextStyleNext = iObject.PerformanceBlockTextStyleNext	Получить свойство (*)
PerformanceBlockTextStyleNext = iObject.GetPerformanceBlockTextStyleNext()	Получить свойство (**)

Синтаксис COM:

iObject->get_PerformanceBlockTextStyleNext (&PerformanceBlockTextStyleNext)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения второй и последующих строк в многострочных названиях блоков исполнений спецификации.
2. Свойство доступно только для чтения.

PerformanceCount – Количество исполнений

Интерфейс...

Тип данных: long

Синтаксис Automation:

PerformanceCount = iObject.PerformanceCount		Получить свойство (*)
iObject.PerformanceCount = PerformanceCount		Установить свойство (*)
PerformanceCount = iObject.GetPerformanceCount()		Получить свойство (**)
iObject.SetPerformanceCount (PerformanceCount)		Установить свойство (**)

Синтаксис COM:

iObject->get_PerformanceCount (&PerformanceCount)	Получить свойство
iObject->put_PerformanceCount (PerformanceCount)	Установить свойство

Примечание:

Позволяет указать количество исполнений специфицируемого изделия. Введенное число будет влиять на доступность ячеек "Количество", на исполнение и количество самих этих ячеек.Имеет смысл только при настройке групповых спецификаций.

PerformanceCountInBlock – Количество исполнений в блоке

Интерфейс...

Тип данных: long

Синтаксис Automation:

PerformanceCountInBlock	=	Получить свойство
Object.PerformanceCountInBlock(PVal)		(*)
PerformanceCountInBlock	=	Получить свойство
Object.GetPerformanceCountInBlock(PVal)		(**)

Синтаксис COM:

Object.get_PerformanceCountInBlock(&PerformanceCountInBlock)	PVal,	Получить свойство

Примечание:

Свойство доступно только для чтения.

PositionUp – Настройка исполнений объектов: позиции возрастают

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PositionUp = iObject.PositionUp		Получить свойство (*)
iObject.PositionUp = PositionUp		Установить свойство (*)
PositionUp	=	Получить свойство
iObject.GetPositionUp()		(**)
iObject.SetPositionUp		Установить свойство
(PositionUp)		(**)

Синтаксис COM:

iObject->get_PositionUp (&PositionUp)	Получить свойство
iObject->put_PositionUp (PositionUp)	Установить свойство

Значения свойства:

TRUE	- позиции объектов-исполнений возрастают,
FALSE	- позиции объектов-исполнений не возрастают.

Примечание:

Позволяет указать, требуется ли в спецификации отображать возрастающие номера объектов, являющихся исполнениями.

PredefinedTextFileName – Имя файла для вставки текстовых шаблонов

Интерфейс...

Тип данных: BSTR (строка)

Синтаксис Automation:

PredefinedTextFileName = iObject.PredefinedTextFileName	Получить свойство (*)
iObject.PredefinedTextFileName = PredefinedTextFileName	Установить свойство (*)
PredefinedTextFileName = iObject.GetPredefinedTextFileName()	Получить свойство (**)
iObject.SetPredefinedTextFileName (PredefinedTextFileName)	Установить свойство (**)

Синтаксис COM:

iObject->get_PredefinedTextFileName (&PredefinedTextFileName)	Получить свойство
iObject->put_PredefinedTextFileName (PredefinedTextFileName)	Установить свойство

Примечание:

Позволяет получить\изменить полное имя файла текстовых шаблонов (*.tdp), тексты из которого требуется вставлять при вводе и редактировании текстовой части объектов спецификации.

Sections – Настройки разделов

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationTuningSections коллекции настроек разделов спецификации

Синтаксис Automation:

Sections = iObject.Sections	Получить свойство (*)
Sections = iObject.GetSections()	Получить свойство (**)

Синтаксис COM:

iObject->get_Sections (&Sections)	Получить свойство

Примечание:

1. Позволяет получить параметры настроек разделов спецификации.
2. Свойство доступно только для чтения.

SectionTextStyleFirst – Стиль текста заголовка раздела – первая строка

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

SectionTextStyleFirst = iObject.SectionTextStyleFirst		Получить свойство (*)
SectionTextStyleFirst = iObject.GetSectionTextStyleFirst( )		Получить свойство (**)

Синтаксис COM:

iObject->get_SectionTextStyleFirst (&SectionTextStyleFirst) — Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения первой строки в названиях разделов спецификации.
2. Свойство доступно только для чтения.

SectionTextStyleNext – Стиль текста заголовка раздела – последующие строки

Интерфейс...

Тип данных: указатель на интерфейс ITextStyle параметров стиля текста

Синтаксис Automation:

SectionTextStyleNext = iObject.SectionTextStyleNext		Получить свойство (*)

SectionTextStyleNext = iObject.GetSectionTextStyleNext( )	Получить свойство (**)

Синтаксис COM:

iObject->get_SectionTextStyleNext (&SectionTextStyleNext)	Получить свойство

Примечание:

1. Позволяет получить параметры шрифта для отображения второй и последующих строк в многострочных названиях разделов спецификации.
2. Свойство доступно только для чтения.

ShowAdditionalBlockName – Показывать имя дополнительного блока

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowAdditionalBlockName = iObject.ShowAdditionalBlockName	Получить свойство (*)
iObject.ShowAdditionalBlockName = ShowAdditionalBlockName	Установить свойство (*)
ShowAdditionalBlockName = iObject.GetShowAdditionalBlockName()	Получить свойство (**)
iObject.SetShowAdditionalBlockName (ShowAdditionalBlockName)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowAdditionalBlockName (&ShowAdditionalBlockName)	Получить свойство
iObject->put_ShowAdditionalBlockName (ShowAdditionalBlockName)	Установить свойство

Значения свойства:

TRUE	- имена дополнительных блоков показываются в таблице спецификации,
FALSE	- имена дополнительных блоков не показываются в таблице спецификации.

Примечание:

Позволяет включать и отключать показ в таблице спецификации заголовков блоков дополнительных разделов и пустых строк вокруг них.

ShowEmbodimentWithoutVariables – Показывать исполнения, не содержащие переменных данных

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowEmbodimentWithoutVariables = Object.ShowEmbodimentWithoutVariables	=	Получить свойство (*)
Object.ShowEmbodimentWithoutVariables = ShowEmbodimentWithoutVariables	=	Установить свойство (*)
ShowEmbodimentWithoutVariables = Object.GetShowEmbodimentWithoutVariables()	=	Получить свойство (**)
Object.SetShowEmbodimentWithoutVariables(ShowEmbodimentWithoutVariables)		Установить свойство (**)

Синтаксис COM:

Object.get_ShowEmbodimentWithoutVariables(&ShowEmbodimentWithoutVariables)	Получить свойство
Object.put_ShowEmbodimentWithoutVariables(ShowEmbodimentWithoutVariables)	Установить свойство

ShowInfoByObjects – Выдавать информацию по объектам (для исполнений более 10)

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowInfoByObjects = iObject.ShowInfoByObjects	=	Получить свойство (*)
iObject.ShowInfoByObjects = ShowInfoByObjects	=	Установить свойство (*)

ShowInfoByObjects = iObject.GetShowInfoByObjects()	Получить свойство (**)
iObject.SetShowInfoByObjects (ShowInfoByObjects)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowInfoByObjects (&ShowInfoByObjects)	Получить свойство
iObject->put_ShowInfoByObjects (ShowInfoByObjects)	Установить свойство

Значения свойства:

TRUE	- выдавать информацию по объектам,
FALSE	- выдавать информацию блоками.

Примечание:

Позволяет выбрать способ отображения информации.

Свойство имеет смысл, если число исполнений, введенное в общих настройках спецификации, больше, чем количество колонок бланка спецификации, предназначенное для ввода количества на исполнение, становятся доступны выбора способа отображения информации.

ShowListCountIsSameFormat – Отображать количество листов одинакового формата

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowListCountIsSameFormat = Object.ShowListCountIsSameFormat	Получить свойство (*)
Object.ShowListCountIsSameFormat = ShowListCountIsSameFormat	Установить свойство (*)
ShowListCountIsSameFormat = Object.GetShowListCountIsSameFormat()	Получить свойство (**)
Object.SetShowListCountIsSameFormat(ShowListCountIsSameFormat)	Установить свойство (**)

Синтаксис COM:

Object.get_ShowListCountIsSameFormat(&ShowListCountIsSameFormat)	Получить свойство
Object.put_ShowListCountIsSameFormat(ShowListCountIsSameFormat)	Установить свойство

ShowNestingBlockName – Показывать имя вложенного блока

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowNestingBlockName = iObject.ShowNestingBlockName	Получить свойство (*)
iObject.ShowNestingBlockName = ShowNestingBlockName	Установить свойство (*)
ShowNestingBlockName = iObject.GetShowNestingBlockName()	Получить свойство (**)
iObject.SetShowNestingBlockName (ShowNestingBlockName)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowNestingBlockName (&ShowNestingBlockName)	Получить свойство
iObject->put_ShowNestingBlockName (ShowNestingBlockName)	Установить свойство

Значения свойства:

TRUE	- имена вложенных блоков показываются в таблице спецификации,
FALSE	- имена вложенных блоков не показываются в таблице спецификации.

Примечание:

Позволяет включать и отключать показ в таблице спецификации заголовков блоков вложенных разделов и пустых строк вокруг них.

ShowPerformanceBlockName – Показывать имя блока исполнений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowPerformanceBlockName = iObject.ShowPerformanceBlockName	Получить свойство (*)
iObject.ShowPerformanceBlockName = ShowPerformanceBlockName	Установить свойство (*)
ShowPerformanceBlockName = iObject.GetShowPerformanceBlockName()	Получить свойство (**)
iObject.SetShowPerformanceBlockName (ShowPerformanceBlockName)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowPerformanceBlockName (&ShowPerformanceBlockName)	Получить свойство
iObject->put_ShowPerformanceBlockName (ShowPerformanceBlockName)	Установить свойство

Значения свойства:

TRUE	- имена блоков исполнений показываются в таблице спецификации,
FALSE	- имена блоков исполнений не показываются в таблице спецификации.

Примечание:

Позволяет включать и отключать показ в таблице спецификации заголовков блоков исполнений и пустых строк вокруг них.

ShowPerformanceFull – Обозначение исполнения показывать полностью

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowPerformanceFull = iObject.ShowPerformanceFull		Получить свойство (*)
iObject.ShowPerformanceFull = ShowPerformanceFull		Установить свойство (*)
ShowPerformanceFull = iObject.GetShowPerformanceFull ()		Получить свойство (**)
iObject.SetShowPerformanceFull (ShowPerformanceFull)		Установить свойство (**)

Синтаксис COM:

iObject->get_ShowPerformanceFull (&ShowPerformanceFull)	Получить свойство
iObject->put_ShowPerformanceFull (ShowPerformanceFull)	Установить свойство

Значения свойства:

TRUE	- обозначение исполнения показывать полностью,
FALSE	- показывать только номер.

Примечание:

Позволяет определить вариант отображения обозначений объектов-исполнений.

ShowSectionName – Режим отображения заголовков разделов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowSectionName = iObject.ShowSectionName		Получить свойство (*)
iObject.ShowSectionName = ShowSectionName		Установить свойство (*)

ShowSectionName = iObject.GetShowSectionName()	Получить свойство (**)
iObject.SetShowSectionName (ShowSectionName)	Установить свойство (**)

Синтаксис COM:

iObject->get_ShowSectionName (&ShowSectionName)	Получить свойство
iObject->put_ShowSectionName (ShowSectionName)	Установить свойство

Значения свойства:

TRUE	- заголовки разделов показываются в таблице спецификации,
FALSE	- заголовки разделов не показываются в таблице спецификации.

Примечание:

Позволяет включить или отключить показ в таблице спецификации заголовков разделов и пустых строк. Если значение свойства - FALSE, объекты показываются в таблице спецификации непрерывным списком. При этом порядок их сортировки сохраняется (в том числе учитывается принадлежность объектов к разным разделам). На отображение резервных строк значение этого свойства не влияет.

SupportPerformance – Поддержка исполнений в спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SupportPerformance = iObject.SupportPerformance	Получить свойство (*)
iObject.SupportPerformance = SupportPerformance	Установить свойство (*)
SupportPerformance = iObject.GetSupportPerformance( )	Получить свойство (**)
iObject.SetSupportPerformance (SupportPerformance)	Установить свойство (**)

Синтаксис COM:

iObject->get_SupportPerformance (&SupportPerformance)	Получить свойство
iObject->put_SupportPerformance (SupportPerformance)	Установить свойство

Значения свойства:

TRUE	- исполнения поддерживаются в спецификации,
FALSE	- исполнения не поддерживаются в спецификации.

Примечание:

Позволяет определить требуется ли в спецификации создавать объекты-исполнения.

UseAdditionalBlocks – Использовать блоки дополнительных разделов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseAdditionalBlocks = iObject.UseAdditionalBlocks		Получить свойство (*)
iObject.UseAdditionalBlocks = UseAdditionalBlocks		Установить свойство (*)
UseAdditionalBlocks = iObject.GetUseAdditionalBlocks()		Получить свойство (**)
iObject.SetUseAdditionalBlocks (UseAdditionalBlocks)		Установить свойство (**)

Синтаксис COM:

iObject->get_UseAdditionalBlocks (&UseAdditionalBlocks)	Получить свойство
iObject->put_UseAdditionalBlocks (UseAdditionalBlocks)	Установить свойство

Значения свойства:

TRUE	- использовать блоки дополнительных разделов,
FALSE	- блоки дополнительных разделов использовать-ся не будут.

Примечание:

Позволяет получить\изменить возможность использования блоков дополнительных разделов. Если значение свойства - TRUE, при создании объектов в спецификации доступны блоки дополнительных разделов.

UserTextStyle – Стиль текста объектов спецификации пользовательский

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UserTextStyle = iObject.UserTextStyle	=	Получить свойство (*)
iObject.UserTextStyle = UserTextStyle	=	Установить свойство (*)
UserTextStyle = iObject.GetUserTextStyle()	=	Получить свойство (**)
iObject.SetUserTextStyle (UserTextStyle)		Установить свойство (**)

Синтаксис COM:

iObject->get_UserTextStyle (&UserTextStyle)	Получить свойство
iObject->put_UserTextStyle (UserTextStyle)	Установить свойство

Значения свойства:

TRUE	- шрифт текстовой части объектов спецификации задается свойством ObjectTextStyle,
FALSE	- шрифт текстовой части объектов спецификации будет таким, каким он был установлен по умолчанию в таблице спецификации.

Примечание:

Позволяет определить, какой шрифт будет использоваться для отображения текстовой части объектов спецификации.

ISpecificationTuning – методы

Update – Обновить описание спецификации в соответствии с внесенными в настройки изменениями

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update ([out, retval] VARIANT_BOOL* pVal);

Возвращаемое значение:

TRUE	- обновление прошло успешно,
FALSE	- обновить настройки не удалось.

Примечание:

Все изменения, сделанные в интерфейсах настроек спецификации, вступят в силу только после вызова этого метода.

Интерфейс ISpecificationTuningSection

Настройка раздела спецификации.

Иерархия:

IKompasAPIObject

ISpecificationTuningSection

Описание:

Данный интерфейс позволяет получить доступ к настройкам раздела спецификации. Если настройки взяты у стиля спецификации, то они являются умолчательными для данного стиля и изменять их нельзя. В этом случае данный интерфейс отражает содержимое настроек стиля в библиотеке стилей на момент получения и в дальнейшем не отслеживает изменений в библиотеке стилей.

Если настройки взяты у описания спецификации, то они отображают и отслеживают текущее состояние описания спецификации. Эти настройки можно изменять и они могут отличаться от умолчательных настроек стиля спецификации, по которому создано данное описание спецификации.

После изменения настроек для того, чтобы эти изменения реально отобразились в документе, необходимо вызвать метод ISpecificationTuning::Update.

Примечание:

Данный интерфейс можно получить у коллекции ISpecificationTuningSections.

ISpecificationTuningSection – свойства

AdditionalBlocks – Блоки вложенных разделов

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockTunings коллекции настроек блоков вложенных разделов спецификации

Синтаксис Automation:

AdditionalBlocks iObject.AdditionalBlocks	=	Получить свойство (*)
AdditionalBlocks iObject.GetAdditionalBlocks()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AdditionalBlocks (&AdditionalBlocks)	Получить свойство

Примечание:

1. Позволяет получить параметры настроек блоков вложенных разделов спецификации.
2. Свойство доступно только для чтения.

AttachGeometry – Подключать геометрию

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AttachGeometry iObject.AttachGeometry	=	Получить свойство (*)
iObject.AttachGeometry AttachGeometry	=	Установить свойство (*)
AttachGeometry iObject.GetAttachGeometry()	=	Получить свойство (**)
iObject.SetAttachGeometry (AttachGeometry)		Установить свойство (**)

Синтаксис COM:

iObject->get_AttachGeometry (&AttachGeometry)	Получить свойство
iObject->put_AttachGeometry (AttachGeometry)	Установить свойство

Значения свойства:

TRUE	- объектам раздела можно поставить в соответствие геометрические объекты графического документа,
FALSE	- объектам раздела нельзя поставить в соответствие геометрические объекты графического документа.

Примечание:

Позволяет включать и отключать подключение геометрии к объектам раздела.

FirstOnSheet – Размещать на новом листе

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

FirstOnSheet = iObject.FirstOnSheet	=	Получить свойство (*)
FirstOnSheet = iObject.FirstOnSheet	=	Установить свойство (*)
FirstOnSheet = iObject.GetFirstOnSheet()	=	Получить свойство (**)
iObject.SetFirstOnSheet (FirstOnSheet)		Установить свойство (**)

Синтаксис COM:

iObject->get_FirstOnSheet (&FirstOnSheet)	Получить свойство
iObject->put_FirstOnSheet (FirstOnSheet)	Установить свойство

Значения свойства:

TRUE	- располагать раздел на новой странице,
FALSE	- располагать раздел сразу за предыдущим.

Примечание:

Позволяет включать и отключать размещение раздела на новом листе спецификации.

IndependentPosition – Независимая нумерация позиций

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IndependentPosition = Object.IndependentPosition		Получить свойство (*)
Object.IndependentPosition = IndependentPosition		Установить свойство (*)
IndependentPosition = Object.GetIndependentPosition()		Получить свойство (**)
Object.SetIndependentPosition(IndependentPosition)		Установить свойство (**)

Синтаксис COM:

Object.get_IndependentPosition(&IndependentPosition)	Получить свойство
Object.put_IndependentPosition(IndependentPosition)	Установить свойство

MarkOn – Марка. Включена простановка позиций с префиксом

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

MarkOn = Object.MarkOn	Получить свойство (*)
Object.MarkOn = MarkOn	Установить свойство (*)
MarkOn = Object.GetMarkOn()	Получить свойство (**)
Object.SetMarkOn(MarkOn)	Установить свойство (**)

Синтаксис COM:

Object.get_MarkOn(&MarkOn)	Получить свойство
Object.put_MarkOn(MarkOn)	Установить свойство

Mark – Марка. Текст префикса

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Mark = Object.Mark	Получить свойство (*)
Object.Mark = Mark	Установить свойство (*)
Mark = Object.GetMark()	Получить свойство (**)
Object.SetMark(Mark)	Установить свойство (**)

Синтаксис COM:

Object.get_Mark(&Mark)	Получить свойство
Object.put_Mark(Mark)	Установить свойство

Number – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = iObject.Number	Получить свойство (*)
Number = iObject.GetNumber()	Получить свойство (**)

Синтаксис COM:

iObject->get_Number (&Number)	Получить свойство

Примечание:

1. Данное свойство определяет уникальный номер раздела в данном стиле спецификации.
2. Свойство доступно только для чтения.

PutPosition – Проставлять позиции

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PutPosition = iObject.PutPosition	Получить свойство (*)
iObject.PutPosition = PutPosition	Установить свойство (*)
PutPosition = iObject.GetPutPosition()	Получить свойство (**)
iObject.SetPutPosition (PutPosition)	Установить свойство (**)

Синтаксис COM:

iObject->get_PutPosition (&PutPosition)	Получить свойство
iObject->put_PutPosition (PutPosition)	Установить свойство

Значения свойства:

TRUE	- в разделе возможен автоматический расчет номеров позиций,
FALSE	- в разделе невозможен автоматический расчет номеров позиций.

Примечание:

Позволяет включать и отключать простановку позиций в разделе. Следует понимать, что отключение простановки позиций отличается от запрета ввода данных в колонку "Позиция". Если в разделе включена простановка позиций, но запрещен ввод данных в колонку "Позиция", то объектам этого раздела при автоматической простановке позиций будут присвоены номера позиций. И хотя эти номера не будут видны в таблице (из-за запрета на ввод данных в колонку "Позиция"), нумерация объектов в следующем разделе будет производиться с их учетом.

Если в разделе отключена простановка позиций и разрешен ввод данных в колонку "Позиция", то после автоматической простановки позиций номера позиций в этом разделе не изменятся, а номера позиций в следующем разделе не будут учитывать введенные вручную номера позиций в данном разделе. Обычно запрет ввода данных в колонку "Позиция" какого-либо раздела сочетают с отключением простановки позиций в этом разделе (пример такого раздела — Документация). И наоборот, если данные в колонку "Позиция" раздела вводить можно, то в этом разделе включают автоматическую простановку позиций.

ReserveStringCount – Число резервных строк и позиций

Интерфейс...

Тип данных: long

Синтаксис Automation:

ReserveStringCount = iObject.ReserveStringCount		Получить свойство (*)
iObject.ReserveStringCount = ReserveStringCount		Установить свойство (*)
ReserveStringCount = iObject.GetReserveStringCount()		Получить свойство (**)
iObject.SetReserveStringCount (ReserveStringCount)		Установить свойство (**)

Синтаксис COM:

iObject->get_ReserveStringCount (&ReserveStringCount)	Получить свойство
iObject->put_ReserveStringCount (ReserveStringCount)	Установить свойство

Примечание:

Позволяет получить или указать количество резервных строк в разделе.

ShowDocumentCode – Показывать код документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowDocumentCode = Object.ShowDocumentCode	=	Получить свойство (*)
Object.ShowDocumentCode = ShowDocumentCode	=	Установить свойство (*)
ShowShowDocumentCode = Object.GetShowDocumentCode( )	=	Получить свойство (**)
Object.SetShowDocumentCode(ShowDocumentCode)		Установить свойство (**)

Синтаксис COM:

Object.get_ShowDocumentCode (&ShowDocumentCode)	Получить свойство
Object.put_ShowDocumentCode (ShowDocumentCode)	Установить свойство

SortObjects – Сортировать объекты

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SortObjects = iObject.SortObjects	=	Получить свойство (*)
iObject.SortObjects = SortObjects	=	Установить свойство (*)
SortObjects = iObject.GetSortObjects()	=	Получить свойство (**)
iObject.SetSortObjects (SortObjects)		Установить свойство (**)

Синтаксис COM:

iObject->get_SortObjects (&SortObjects)	Получить свойство

iObject->put_SortObjects (SortObjects)	Установить свойство

Значения свойства:

TRUE	- сортировка объектов внутри раздела производится по предусмотренным стандартом правилам,
FALSE	- в разделе сортировка объектов не производится.

Примечание:

Позволяет включать и отключать автоматическую сортировку объектов в разделе. Если сортировка выключена, объекты будут добавляться в конец раздела и их можно будет перемещать вверх или вниз.

SubsectionOn – Деление на подразделы включено

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

SubsectionOn = iObject.SubsectionOn	=	Получить свойство (*)
iObject.SubsectionOn = SubsectionOn	=	Установить свойство (*)
SubsectionOn = iObject.GetSubsectionOn()	=	Получить свойство (**)
iObject.SetSubsectionOn (SubsectionOn)		Установить свойство (**)

Синтаксис COM:

iObject->get_SubsectionOn (&SubsectionOn)	Получить свойство
iObject->put_SubsectionOn (SubsectionOn)	Установить свойство

Значения свойства:

TRUE	- деление на подразделы включено,
FALSE	- деление на подразделы выключено.

Примечание:

Позволяет включать и отключать деление раздела на подразделы. Если деление на подразделы включено, объекты внутри раздела будут располагаться с учетом порядка подраздела. Подразделы располагаются в разделе в порядке возрастания номеров.

Subsections – Подразделы

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationSubsections подразделов спецификации

Синтаксис Automation:

Subsections = iObject.Subsections		Получить свойство (*)
Subsections = iObject.GetSubsections()		Получить свойство (**)

Синтаксис COM:

iObject->get_Subsections (&Subsections)	Получить свойство

Примечание:

1. Позволяет получить коллекцию подразделов данного раздела спецификации.
2. Свойство доступно только для чтения.

UseAdditionalBlocks – Использовать блоки вложенных разделов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseAdditionalBlocks = iObject.UseAdditionalBlocks		Получить свойство (*)
iObject.UseAdditionalBlocks = UseAdditionalBlocks		Установить свойство (*)
UseAdditionalBlocks = iObject.GetUseAdditionalBlocks( )		Получить свойство (**)
iObject.SetUseAdditionalBlocks (UseAdditionalBlocks)		Установить свойство (**)

Синтаксис COM:

iObject->get_UseAdditionalBlocks (&UseAdditionalBlocks)	Получить свойство
iObject->put_UseAdditionalBlocks (UseAdditionalBlocks)	Установить свойство

Значения свойства:

TRUE	- использовать блоки вложенных разделов,
FALSE	- блоки вложенных разделов использоваться не будут.

Примечание:

Позволяет получить или указать количество резервных строк в разделе.

Интерфейс IAdditionalBlockSectionTunings

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_ADDITION_TUNING_BLOCK.htm

Интерфейс коллекции настроек разделов блока дополнительных (вложенных) разделов.

Иерархия:

IKompasAPIObject

IKompasCollection

IAdditionalBlockSectionTunings

Описание:

Управляет массивом интерфейсов IAdditionalBlockSectionTuning. В коллекции разделов доступны все разделы, определенные в стиле спецификации. Настройки разделов, использующихся в блоках, полностью совпадают с настройками соответствующих обычных разделов.

Примечание:

Данный интерфейс может быть получен от интерфейса настройки блока дополнительных (вложенных) разделов IAdditionalBlockTuning с помощью свойства IAdditionalBlockTuning::Sections

IAdditionalBlockSectionTunings – свойства

Item – Настройки раздела блока дополнительных (вложенных) разделов, заданные по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockSectionTuning настроек раздела блока дополнительных (вложенных) разделов

Синтаксис Automation:

BlockTuning = iObject.Item (Index)	Получить свойство (*)
BlockTuning = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &BlockTuning)	(Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IAdditionalBlockStyles

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_ADDITION_TUNING_BLOCK.htm

Интерфейс коллекции стилей блоков дополнительных (вложенных) разделов.

Иерархия:

IKompasAPIObject

IKompasCollection

IAdditionalBlockStyles

Описание:

Управляет массивом интерфейсов IAdditionalBlockStyle.

Блоки разделов бывают вложенными и дополнительными. Вложенные блоки разделов располагаются внутри раздела, после всех объектов этого раздела.

Дополнительные блоки разделов располагаются в конце спецификации, после всех ее разделов.

Примечание:

Данный интерфейс может быть получен от интерфейса стиля спецификации ISpecificationStyle с помощью свойства ISpecificationStyle::AdditionalBlocks.

IAdditionalBlockStyles – свойства

Item – Стиль блока дополнительных (вложенных) разделов спецификации, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockStyle стиля блока дополнительных (вложенных) разделов спецификации

Синтаксис Automation:

Block = iObject.Item (Index)	Получить свойство (*)
Block = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (Index, &Block)		Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IAdditionalBlockTunings

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_ADDITION_BLOCK.htm

Интерфейс коллекции настроек блоков дополнительных (вложенных) разделов.

Иерархия:

IKompasAPIObject

IKompasCollection

IAdditionalBlockTunings

Описание:

Управляет массивом интерфейсов IAdditionalBlockTuning.

Примечание:

1. Интерфейс настроек блоков дополнительных разделов можно получить от интерфейса настроек спецификации ISpecificationTuning с помощью свойства ISpecificationTuning::AdditionalBlocks.
2. Интерфейс коллекции настроек блоков вложенных разделов можно получить от интерфейса настроек раздела спецификации ISpecificationTuningSection с помощью свойства ISpecificationTuningSection::AdditionalBlocks.

IAdditionalBlockTunings – свойства

Item – Настройки блока дополнительных (вложенных) разделов, заданные по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAdditionalBlockTuning настроек блока дополнительных (вложенных) разделов

Синтаксис Automation:

BlockTuning = iObject.Item (Index)		Получить свойство (*)
BlockTuning = iObject.GetItem(Index)		Получить свойство (**)

Синтаксис COM:

iObject->get_Item(&BlockTuning)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения

Интерфейс IAttachedDocuments

Интерфейс коллекции документов, присоединенных к объекту спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

IAttachedDocuments

Описание:

Объект спецификации можно связать с документами КОМПАС-3D. Эта связь является двусторонней и ассоциативной и позволяет передавать данные об объекте спецификации в подключенный документ или наоборот, данные из документа в соответствующий ему объект спецификации. Если к объекту спецификации подключено несколько документов, то в объект передаются данные из первого подключенного документа. Если отредактировать подключенную к объекту модель или данные в основной надписи подключенного документа, а также изменить его формат, то при сохранении этого документа будет выдан запрос на подтверждение передачи данных в спецификацию. После изменения наименования или обозначения объекта в документе-спецификации новые данные могут быть переданы в основную надпись подключенного документа.

Примечание:

1. Управляет массивом интерфейсов IAttachedDocument.
2. Данный интерфейс можно получить у интерфейса объекта спецификации ISpecificationObject с помощью свойства ISpecificationObject::AttachedDocuments.

IAttachedDocuments – свойства

Item – Присоединенный документ, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс IAttachedDocument присоединенного документа

Синтаксис Automation:

ColumnItem = iObject.Item(Index)		Получить свойство (*)
ColumnItem = iObject.GetItem(Index)		Получить свойство (**)

Синтаксис COM:

iObject->get_Item(&ColumnItem)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

IAttachedDocuments – методы

Add – Добавить новый документ к объекту спецификации и в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add(BSTR Name, BOOL transmit);

Синтаксис COM:

HRESULT Add ([in] BSTR Name,
[in] BOOL transmit,
[out, retval] IAttachedDocument** Result);

Входные параметры:

Name	- полное имя файла присоединяемого документа,
transmit	- передавать изменения в документ.

Возвращаемое значение:

- указатель на интерфейс IAttachedDocument присоединенного документа.

Примечание:

1. После вызова метода Add документ будет добавлен в коллекцию.
2. Для удаления документа из коллекции используется метод IAttachedDocument::Delete
3. После добавления нужных документов вызвать метод ISpecificationObject::Update.
4. В результате подключения документа в соответствующие колонки объекта спецификации автоматически будут переданы наименование и обозначение из модели или основной надписи чертежа, а также обозначение формата, на котором выполнен подключенный чертеж.
5. Информация из подключенного документа может передаваться не только в колонки бланка спецификации, но и в дополнительные колонки. Например, в дополнительную колонку объекта может быть автоматически передана масса детали из модели или из соответствующей графы основной надписи.

AddDocument – Добавляет новый документ к объекту спецификации и в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddDocument(BSTR Name, BOOL Transmit, BOOL FillText, BSTR

Синтаксис COM:

HRESULT AddDocument(BSTR Name, BOOL Transmit, BOOL FillText, BSTR

** Result);

Возвращаемое значение:

- указатель на интерфейс IAttachedDocument присоединенного документа.

Входные параметры:

Name	- полное имя присоединяемого документа,
Transmit	- передавать изменения в документ,
FillText	- взять данные из документа в объект спецификации,
EmbodimentMarking	- имя подключаемого исполнения.

Интерфейс ISpecificationBaseObjects

Интерфейс коллекции базовых объектов спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationBaseObjects

Описание:

Позволяет получить список базовых объектов спецификации.

Примечание:

1. Управляет массивом интерфейсов ISpecificationBaseObject.
2. Коллекция позволяет создать новые базовые объекты спецификации. Для создания нового базового объекта спецификации используется метод Add. После вызова метода Add объект будет добавлен в коллекцию, но в документе не появится.
3. Для отображения объекта в документе нужно задать его параметры и вызвать метод ISpecificationObject::Update.
4. Данный интерфейс можно получить от интерфейса описания спецификации ISpecificationDescription с помощью свойства ISpecificationDescription::BaseObjects.

ISpecificationBaseObjects – свойства

Item – Базовый объект спецификации, заданный по индексу, по уникальному идентификатору или по уникальному номеру

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationBaseObject объекта спецификации

Синтаксис Automation:

spcObj = iObject.Item (Index)	Получить свойство (*)
spcObj = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &spcObj)	(Index,	Получить свойство

Примечание:

1. С помощью данного свойства можно получить базовый объект спецификации, заданный по индексу, по уникальному идентификатору IKompasAPIObject::Reference или уникальному номеру ISpecificationObject::UniqueNumber.
2. Свойство доступно только для чтения.

ISpecificationBaseObjects – методы

Add – Добавить новый базовый объект спецификации в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (long Section, double AttrNumb);

Синтаксис COM:

HRESULT Add ([in] long Section,
[in] double AttrNumb,
[out, retval] ISpecificationBaseObject** Result);

Входные параметры:

Section	- номер раздела,
AttrNumb	- номер атрибута.

Возвращаемое значение:

- Указатель на интерфейс ISpecificationBaseObject базового объекта спецификации.

Примечание:

1. После вызова метода Add объект будет добавлен в коллекцию, но в документе не появится. Для отображения объекта в документе нужно задать его параметры и вызвать метод ISpecificationObject::Update.
2. Объект с номером раздела 0.0. добавить нельзя. Нужно добавлять объект спецификации в существующий раздел. Затем с помощью ISpecificationObject::AdditionalSection переносить в дополнительный раздел.

CopySpecificationObject – Копирование объекта спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH CopySpecificationObject(ISpecificationBaseObject * SpcObj, VARIANT Geometry);

Синтаксис COM:

HRESULT CopySpecificationObject(ISpecificationBaseObject * SpcObj, VARIANT Geometry, ISpecificationBaseObject ** Result);

Возвращаемое значение:

- Указатель на интерфейс скопированного объекта.

Входные параметры:

SpcObj	- копируемый объект,
Geometry	- геометрия объекта спецификации (Объект VT_DISPATCH или список объектов VT_ARRAY | VT_DISPATCH).

GetSpecificationObjectsForGeom – По геометрии получить массив объектов спецификации в виде SAFEARRAY'я DISPATCH – VT_ARRAY | VT_DISPATCH

Интерфейс...

Синтаксис Automation:

VARIANT GetSpecificationObjectsForGeom(VARIANT PGeom,
VARIANT_BOOL Equal,

long SectionNumb,
double AttrNumb);

Синтаксис COM:

HRESULT GetSpecificationObjectsForGeom([in]VARIANT PGeom,
[in]VARIANT_BOOL Equal,
[in] long SectionNumb,
[in] double AttrNumb,
[out, retval] VARIANT* PVal);

Входные параметры:

PGeom	- Массив SafeArray типа VT_ARRAY | VT_DISPATCH объектов геометрии,
Equal	- флаг типа BOOL TRUE - полное совпадение геометрии, FALSE - частичное совпадение геометрии,
SectionNumb	- номер раздела объектов СП,
AttrNumb	- номер типа атрибута объектов СП или 0.

Возвращаемое значение:

- Массив SafeArray типа VT_ARRAY | VT_DISPATCH объектов спецификации.

Примечание:

1. По присланной геометрии получить массив базовых объектов спецификации, содержащих эту геометрию. У базового объекта СП может быть установлена геометрия, которая ассоциируется с данным объектом.

   Для 2D документа это массив IDrawingObject.

   Для 3D документа это массив объектов:

- IPart7(может быть один компонент),
- IBody7 (тела верхнего уровня сборки),
- IPositionLeader (обозначения позиций верхнего уровня сборки).

   Получить массив объектов можно всегда.

2. Если геометрия должна совпадать полностью, нужно передать параметр Equal=TRUE.
3. Поиск можно ограничить объектами определенного раздела, если указать номер раздела объектов.
4. Поиск можно ограничить объектами определенного типа атрибута.

Интерфейс ISpecificationColumnItems

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс коллекции элементов колонки объекта спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationColumnItems

Описание:

Количество элементов колонки зависит от типа колонки.

Колонки типа Целое, Вещественное или Строка имеют только один элемент соответствующего типа.

Колонка типа запись (ksValueTypeRecord см. ISpecificationColumn::ValueType) может иметь более одного элемента.

Если базовый объект спецификации создан на основе атрибута, то количество элементов в колонке будет равно количеству колонок атрибута. Если объекту не указан тип атрибута, то в колонке типа запись будет один элемент с типом значения - строка.

Примечание:

1. Управляет массивом интерфейсов ISpecificationColumnItem.
2. Данный интерфейс можно получить от интерфейса колонки объекта спецификации ISpecificationColumn с помощью свойства ISpecificationColumn::ColumnItems.

ISpecificationColumnItems – свойства

Item – Элемент колонки объекта спецификации, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnItem элемента колонки объекта спецификации

Синтаксис Automation:

ColumnItem = iObject.Item(Index)	Получить свойство (*)
ColumnItem = iObject.GetItem(Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item(Index, &ColumnItem)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ISpecificationColumns

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс коллекции колонок объекта спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationColumns

Описание:

Различаются основные и дополнительные колонки объектов спецификации. Основные колонки отображаются в таблице спецификации и выводятся на печать. Дополнительные колонки содержат сведения, дополняющие информацию, включаемую в стандартный бланк. В таблице спецификации такие колонки не видны и на печать не выводятся. Количество основных колонок может не соответствовать количеству колонок в таблице спецификации. Это вызвано наличием исполнений для групповых спецификаций.

Примечание:

1. Управляет массивом интерфейсов ISpecificationColumn.
2. Интерфейс коллекции основных колонок объекта спецификации может быть получен от интерфейса ISpecificationObject с помощью свойства ISpecificationObject::Columns.
3. Интерфейс коллекции дополнительных колонок может быть получен от интерфейса ISpecificationObject с помощью свойства ISpecificationObject::AdditionalColumns.

ISpecificationColumns – свойства

Item – Колонка объекта спецификации, заданная по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationColumn колонки объекта спецификации

Синтаксис Automation:

Column = iObject.Item(Index)	Получить свойство (*)
Column = iObject.GetItem(Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item(&Column)	Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

Column – Колонка объекта спецификации, заданная по типу, номеру и блоку

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumn колонки объекта спецификации

Синтаксис Automation:

Column = iObject.Column(type, ColumnNumber, BlockNumber)	Получить свойство (*)
Column = iObject.GetColumn(type, ColumnNumber, BlockNumber)	Получить свойство (**)

Синтаксис COM:

iObject->get_Column(type, ColumnNumber, BlockNumber, &Column)	Получить свойство

Входные параметры:

type	- тип колонки объекта спецификации из ksSpecificationColumnTypeEnum,
ColumnNumber	- номер колонки данного типа,
BlockNumber	- номер блока.

Примечание:

Свойство доступно только для чтения.

Интерфейс ISpecificationColumnStyles

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm`

Интерфейс коллекции стилей колонок спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationColumnStyles

Описание:

Управляет массивом интерфейсов ISpecificationColumnStyle. Различаются основные и дополнительные колонки объектов спецификации.

Примечание:

1. Чтобы получить коллекцию стилей основных колонок спецификации, используется свойство интерфейса стиля спецификации ISpecificationStyle ISpecificationStyle::Columns или свойство интерфейса стиля раздела спецификации ISpecificationSectionStyle. ISpecificationSectionStyle::Columns.
2. Чтобы получить коллекцию стилей дополнительных колонок спецификации, используется свойство интерфейса стиля спецификации ISpecificationStyle ISpecificationStyle::AdditionalColumns или свойство интерфейса стиля раздела спецификации ISpecificationSectionStyle. ISpecificationSectionStyle::AdditionalColumns.

ISpecificationColumnStyles – свойства

Item – Стиль колонки спецификации, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationColumnStyle стиля колонки спецификации

Синтаксис Automation:

Column = iObject.Item (Index)	Получить свойство (*)
Column = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &Column)	(Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ISpecificationCommentObjects

Справка системы КОМПАС...

KOMPAS.chm::/Z_OBJECT_OF_SPEC.htm

Интерфейс коллекции вспомогательных объектов спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationCommentObjects

Описание:

Позволяет получить список вспомогательных объектов спецификации.

Примечание:

1. Управляет массивом интерфейсов ISpecificationCommentObject.
2. Коллекция позволяет создать новые вспомогательные объекты спецификации. Для создания нового вспомогательного объекта спецификации используется метод Add. После вызова метода Add объект будет добавлен в коллекцию, но в документе не появится.
3. Для отображения объекта в документе нужно задать его параметры и вызвать метод ISpecificationObject::Update.
4. Методы Attach и Detach работают только для коллекции, взятой у базового объекта СП.
5. Данный интерфейс можно получить от интерфейса описания спецификации ISpecificationDescription с помощью свойства ISpecificationDescription::CommentObjects и от интерфейса базового объекта спецификации ISpecificationBaseObject с помощью свойства ISpecificationBaseObject_CommentObjects.

ISpecificationCommentObjects – свойства

Item – Вспомогательный объект спецификации, заданный по индексу, по уникальному идентификатору или по уникальному номеру

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationCommentObject объекта спецификации.

Синтаксис Automation:

spcObj = iObject.Item (Index)	Получить свойство (*)
spcObj = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &spcObj)	(Index,	Получить свойство

Примечание:

1. С помощью данного свойства можно получить вспомогательный объект спецификации, заданный по индексу, по уникальному идентификатору IKompasAPIObject::Reference или уникальному номеру ISpecificationObject::UniqueNumber.
2. Свойство доступно только для чтения.

ISpecificationCommentObjects – методы

Add – Добавить новый вспомогательный объект спецификации в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (long Section);

Синтаксис COM:

HRESULT Add ([in] long Section,
[out, retval] ISpecificationCommentObject** Result);

Входные параметры:

Section	- номер раздела.

Возвращаемое значение:

	- Указатель на интерфейс ISpecificationCommentObject вспомогательного объекта спецификации.

Примечание:

После вызова метода Add объект будет добавлен в коллекцию, но в документе не появится. Для отображения объекта в документе нужно задать его параметры и вызвать метод ISpecificationObject::Update.

Attach – Добавить вспомогательный объект спецификации в коллекцию прикрепленных объектов (Для базового объекта)

Интерфейс...

Синтаксис Automation:

BOOL Attach(LPDISPATCH spcObj);

Синтаксис COM:

HRESULT Attach ([in]ISpecificationCommentObject* spcObj,
[out, retval] BOOL* PRes);

Входные параметры:

spcObj	- указатель на интерфейс прикрепляемого вспомогательного объекта спецификации.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Метод работает только для коллекции взятой у базового объекта СП.

CopySpecificationObject – Копирование объекта спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH CopySpecificationObject(ISpecificationCommentObject * SpcObj);

Синтаксис COM:

HRESULT CopySpecificationObject(ISpecificationCommentObject * SpcObj, ISpecificationCommentObject * * Result);

Возвращаемое значение:

- Указатель на интерфейс скопированного объекта.

Входные параметры:

spcObj - копируемый объект.

Detach – Удалить вспомогательный объект спецификации из коллекции прикрепленных объектов (Для базового объекта)

Интерфейс...

Синтаксис Automation:

BOOL Detach (LPDISPATCH spcObj);

Синтаксис COM:

HRESULT Detach ([in]ISpecificationCommentObject* spcObj,
[out, retval] BOOL* PRes);

Входные параметры:

spcObj - указатель на интерфейс прикрепляемого вспомогательного объекта спецификации.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Метод работает только для коллекции взятой у базового объекта СП.

Интерфейс ISpecificationDescriptions

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE_FRG.htm

Интерфейс коллекции описаний спецификации у документа.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationDescriptions

Примечание:

1. Для документа-спецификации возможно только одно описание, и оно есть всегда (т.е. добавлять и удалять нельзя).
2. В текстовом документе описаний нет.
3. Для остальных документов (чертеж, фрагмент, деталь и сборка) описаний может быть сколько угодно, в том числе и ни одного.
4. При добавлении нового описания оно становится текущим.
5. Данный интерфейс может быть получен следующими способами:

- от интерфейса документа IKompasDocument с помощью свойства IKompasDocument::SpecificationDescriptions,
- от интерфейса чертежа IDrawingDocument с помощью свойства IDrawingDocument::ChangeListDescriptions.

ISpecificationDescriptions – свойства

Active – Текущее описание

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationDescription описания спецификации

Синтаксис Automation:

Description = iObject.Active	Получить свойство (*)
Description = iObject.GetActive()	Получить свойство (**)

Синтаксис COM:

iObject->get_Active (&Description)	Получить свойство

Примечание:

1. Если в документе нет описаний спецификаций, возвращается NULL.
2. Свойство доступно только для чтения.

ActiveFromLibStyle – Текущее описание, соответствующее библиотечному стилю

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationDescription

Синтаксис Automation:

ActiveFromLibStyle = Object.ActiveFromLibStyle	=	Получить свойство (*)
ActiveFromLibStyle = Object.GetActiveFromLibStyle()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ActiveFromLibStyle(&ActiveFromLibStyle))	Получить свойство

Возвращаемое значение:

- если текущее описание соответствует своему описанию-источнику в библиотеке, то просто возвращается оно же,
- если текущее описание не соответствует своему описанию-источнику в библиотеке, то возвращается новое описание, созданное по описанию-источнику и сделанное текущим,
- если текущего описания нет или для текущего описания не найдено описание-источник (не найдена библиотека-источник), то возвращается NULL.

Примечание:

Свойство доступно только для чтения.

Description – Описание спецификации по имени файла библиотеки оформлений и номеру стиля в библиотеке

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationDescription описания спецификации

Синтаксис Automation:

Description = iObject.Description (LayoutName, StyleID)		Получить свойство (*)

Description = iObject.GetDescription (LayoutName, StyleID)		Получить свойство (**)

Синтаксис COM:

iObject->get_Description (LayoutName, StyleID, &Description)	Получить свойство

Входные параметры:

LayoutName	- имя файла библиотеки оформлений,
StyleID	- номер стиля в библиотеке оформлений.

Примечание:

Свойство доступно только для чтения.

Item – Описание спецификации, заданное по индексу

Интерфейс...

Тип данных: Указатель на интерфейс ISpecificationDescription описания спецификации

Синтаксис Automation:

Description = iObject.Item (Index)		Получить свойство (*)
Description = iObject.GetItem (Index)		Получить свойство (**)

Синтаксис COM:

iObject->get_Item (Index, &Description)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ISpecificationDescriptions – методы

Add – Добавить новое описание спецификации в документ и в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (BSTR LayoutName,
long StyleID,
BSTR SpcName,
ISpecificationDescription);

Синтаксис COM:

HRESULT Add ([in] BSTR LayoutName,
[in] long StyleID,
[in] BSTR SpcName,
[out, retval] ISpecificationDescription** Result);

Входные параметры:

LayoutName	- имя библиотеки стилей,
StyleID	- номер стиля в библиотеке стилей,
SpcName	- имя документа спецификации, связанного с описанием спецификации в документе.

Возвращаемое значение:

- Указатель на интерфейс ISpecificationDescription описания спецификации.

Примечание:

К одному документу могут быть добавлены описания спецификации только разных стилей. Если LayoutName и StyleID не заданы, то добавляется описание спецификации по умолчанию, если оно не было создано ранее.

Интерфейс ISpecificationSectionStyles

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_SECTION_STYLE_DIALOG.htm

Интерфейс коллекции стилей разделов спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationSectionStyles

Описание:

Управляет массивом интерфейсов ISpecificationSectionStyle.

Примечание:

Данный интерфейс может быть получен от интерфейса ISpecificationStyle с помощью свойства ISpecificationStyle::Sections.

ISpecificationSectionStyles – свойства

Item – Стиль раздела спецификации, заданный по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationSectionStyle стиля раздела спецификации

Синтаксис Automation:

Section = iObject.Item (Index)	Получить свойство (*)
Section = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &Section)	(Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс ISpecificationSubsections

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SECTION_STYLE_DIALOG.htm`

Интерфейс коллекции подразделов спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationSubsections

Описание:

Управляет массивом интерфейсов ISpecificationSubsection. Группы объектов внутри разделов называются подразделами. Подразделы, как и разделы, являются компонентом стиля спецификации. Их количество, названия и порядок внутри каждого раздела формируются при настройке стиля. Если деление на подразделы не запрещено в стиле текущей спецификации, то можно изменить список подразделов в каждом ее разделе. Подразделы располагаются в разделе в порядке возрастания номеров.

Примечание:

Данный интерфейс можно получить от интерфейса настройки раздела спецификации ISpecificationTuningSection с помощью свойства ISpecificationTuningSection::Subsections.

ISpecificationSubsections – свойства

Item – Настройки раздела спецификации, заданные по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationSubsection подраздела спецификации

Синтаксис Automation:

Subsection = iObject.Item (Index)	Получить свойство (*)
Subsection = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item (Index, &Subsection)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ISpecificationSubsections – методы

Add – Добавить новый подраздел

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add (BSTR Name, short Number);

Синтаксис COM:

HRESULT Add ([in] BSTR Name,
[in] short Number,
[out, retval] ISpecificationSubsections** Result);

Входные параметры:

Name	- имя подраздела,
Number	- номер подраздела.

Возвращаемое значение:

- Указатель на интерфейс подраздела спецификации ISpecificationSubsection.

Интерфейс ISpecificationTuningSections

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SECTION_STYLE_DIALOG.htm

Интерфейс коллекции настроек разделов спецификации.

Иерархия:

IKompasAPIObject

IKompasCollection

ISpecificationTuningSections

Описание:

Управляет массивом интерфейсов ISpecificationTuningSection .

Примечание:

Данный интерфейс может быть получен от интерфейса настроек спецификации ISpecificationTuning с помощью свойства ISpecificationTuning::Sections

ISpecificationTuningSections – свойства

Item – Настройки раздела спецификации, заданные по индексу

Интерфейс...

Тип данных: указатель на интерфейс ISpecificationTuningSection настроек раздела спецификации

Синтаксис Automation:

Section = iObject.Item (Index)	Получить свойство (*)
Section = iObject.GetItem (Index)	Получить свойство (**)

Синтаксис COM:

iObject->get_Item &Section)	(Index,	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейсы событий

Интерфейс IModelObjectNotifyResult

Интерфейс результатов редактирования 3d объекта.

Иерархия:

IDispatch

IModelObjectNotifyResult

Описание:

Позволяет контролировать редактирование объектов документа-модели. Источником событий для подписки является указатель на интерфейс IModelObject.

Примечание:

Получить интерфейс можно с помощью метода IUnknown::QueryInterface у интерфейса компонента IPart7.

IModelObjectNotifyResult – свойства

IsRedoMode – Признак работы команды Redo

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsRedoMode = Object.IsRedoMode;          Получить свойство (* )
IsRedoMode = Object.GetIsRedoMode();     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_IsRedoMode( &IsRedoMode )     Получить свойство
```

Примечание.

Свойство доступно только для чтения.

IsUndoMode – Признак работы команды Undo

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
IsUndoMode = Object.IsUndoMode;          Получить свойство (* )
IsUndoMode = Object.GetIsUndoMode();     Получить свойство (**)
```

Синтаксис COM:

Object.get_IsUndoMode(&IsUndoMode) Получить свойство

Примечание.

Свойство доступно только для чтения.

NotifyObjects – Массив SafeArray удаляемых объектов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

NotifyObjects = Object.NotifyObjects; Получить свойство (*)
NotifyObjects = Object.GetNotifyObjects(); Получить свойство (**)

Синтаксис COM:

iObject->get_NotifyObjects(&NotifyObjects) Получить свойство

Примечание.

Свойство доступно только для чтения.

NotifyType – Тип события

Интерфейс...

Тип данных: из перечисления KsObject3DNotifyEnum

Синтаксис Automation:

NotifyType = Object.NotifyType; Получить свойство (*)
NotifyType = Object.GetNotifyType(); Получить свойство (**)

Синтаксис COM:

iObject->get_NotifyType(&NotifyType) Получить свойство

Примечание.

Свойство доступно только для чтения.

ProcessType – Тип процесса

Интерфейс...

Тип данных: из перечисления ProcessTypeEnum

Синтаксис Automation:

ProcessType = Object.ProcessType; Получить свойство (*)
ProcessType = Object.GetProcessType(); Получить свойство (**)

Синтаксис COM:

Object.get_ProcessType(&ProcessType) Получить свойство

Примечание.

Свойство доступно только для чтения.

Интерфейс IKompasDocument3DNotifyResult

Интерфейс дополнительных параметров для событий документа-модели.

Примечание:

1. Интерфейс позволяет получить указатель на объект, для которого посылается данное событие. Например, выбор материала и обозначения доступны для верхнего компонента, вставки и тела.
2. Интерфейс является дополнительным для IKompasDocument3D.
3. Интерфейс можно получить с помощью метода IUnknown::QueryInterface у IKompasDocument3D.

IKompasDocument3DNotifyResul – свойства

NotifyType – Тип события

Интерфейс...

Тип данных: из перечисления ksDocument3DNotifyEnum

Синтаксис Automation:

NotifyType = Object.NotifyType Получить свойство (*)
NotifyType = Object.GetNotifyType() Получить свойство (**)

Синтаксис COM:

Object.get_NotifyType(&NotifyType) Получить свойство

Примечание.

Свойство доступно только для чтения.

NotifyObjectType – Тип объекта

Интерфейс...

Тип данных: из перечисления ksObj3dTypeEnum

Синтаксис Automation:

NotifyObjectType = Object.NotifyObjectType Получить свойство (*)
NotifyObjectType = Object.GetNotifyObjectType() Получить свойство (**)

Синтаксис COM:

Object.get_NotifyObjectType(&NotifyObjectType) Получить свойство

Возвращаемое значение:

- Тип объекта из перечисления ksObj3dTypeEnum.

Допустимыми значениями являются o3d_part, o3d_body или o3d_uknown.

Примечание.

Свойство доступно только для чтения.

NotifyObject – Тип объекта

Интерфейс...

Тип данных: указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

NotifyObject = Object.NotifyObject Получить свойство (*)
NotifyObject = Object.GetNotifyObject() Получить свойство (**)

Синтаксис COM:

Object.get_NotifyObject(&NotifyObject) Получить свойство

Возвращаемое значение:

В зависимости от типа объекта, возвращаемого методом NotifyObjectType возвращается:

тип объекта	Интерфейс
o3d_part	- IPart7
o3d_body	- IBody7
o3d_uknown	- NULL

Примечание.

Свойство доступно только для чтения.

RequestFilesType – Тип процесса, запрашивающего файл

Интерфейс...

Тип данных: из перечисления ksRequestFilesTypeEnum

Синтаксис Automation:

RequestFilesType = Получить свойство (*)
Object.RequestFilesType
RequestFilesType = Получить свойство (**)
Object.GetRequestFilesType()

Синтаксис COM:

Object.get_RequestFilesType(Получить
&RequestFilesType) свойство

Возвращаемое значение:

Возвращает идентификатор процесса, запрашивающего файл.

Примечание.

Свойство доступно только для чтения.

Интерфейс ksDocument3DNotify7

Интерфейс событий документа-модели в API7 (disp-интерфейс).

Позволяет контролировать редактирование документа-модели.

Источником для подписки является документ-модель.

ksDocument3DNotify7 – события

BeginChoiceMarking – Начало выбора обозначения

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL BeginChoiceMarking();

Возвращаемое значение:

TRUE - выбрать обозначение,
FALSE - выбор обозначения запрещен.

BeginChoiceMaterial – Начало выбора материала

Интерфейс...

Синтаксис:

VARIANT_BOOL BeginChoiceMaterial();

Возвращаемое значение:

TRUE - выбрать материал,

FALSE - выбор материала запрещен.

BeginCreatePartFromFile – Начало создания компонента в сборке (до диалога выбора имени)

Интерфейс...

Синтаксис:

VARIANT_BOOL BeginCreatePartFromFile (VARIANT_BOOL typeDoc, LPDISPATCH * plane);

Входные параметры:

typeDoc	TRUE - вставить деталь, FALSE - вставить сборку,
plane	- плоскость типа IModelObject, на которой создается компонент.

Возвращаемое значение:

TRUE	- установить компонент в сборку,
FALSE	- установка компонента в сборку запрещена.

BeginRebuild – Начало перестроения модели

Интерфейс...

Синтаксис:

VARIANT_BOOL BeginRebuild();

Возвращаемое значение:

TRUE	- перестроить модель,
FALSE	- перестройка модели запрещена.

BeginSetPartFromFile – Начало установки компонента в сборку (до диалога выбора имени)

Интерфейс...

Синтаксис:

VARIANT_BOOL BeginSetPartFromFile();

Возвращаемое значение:

TRUE	- установить компонент в сборку,
FALSE	- установка компонента в сборку запрещена.

ChoiceMarking – Закончен выбор обозначения

Интерфейс...

Синтаксис:

VARIANT_BOOL ChoiceMarking(BSTR marking);

Входные параметры:

marking - обозначение.

Возвращаемое значение:

TRUE - выбрано обозначение,
FALSE - выбор обозначения запрещен.

ChoiceMaterial – Закончен выбор материала

Интерфейс...

Синтаксис:

VARIANT_BOOL ChoiceMaterial (BSTR material, double density);

Возвращаемое значение:

TRUE - выбран материал,
FALSE - выбор материала запрещен.

Rebuild – Модель перестроена

Интерфейс...

Синтаксис:

VARIANT_BOOL Rebuild();

Возвращаемое значение:

TRUE - модель перестроена,
FALSE - перестройка модели запрещена.

Интерфейс ksDocumentFrameNotify/ IDocumentFrameNotify

ksDocumentFrameNotify - интерфейс Automation,
IdocumentFrameNotify - интерфейс COM

Интерфейс событий для окна документа (клавиатура, мышь, события по отрисовке документа)

Позволяют контролировать состояние окна документа.

Источником событий для подписки на данный интерфейс является объект IDocumentFrame - интерфейс окна документа.

В COM источником для подписки является указатель на документ. Дополнительный параметр при подписке задает интерфейс окна IDocumentFrame, для которого предназначен данный интерфейс событий.

ksDocumentFrameNotify, IDocumentFrameNotify – события

Activate – Активизация окна

Интерфейс...

BOOL Activate();

Синтаксис COM:

BOOL Activate();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

AddGabarit – Рассчитывается общий габарит окна

Интерфейс...

Синтаксис Automation:

BOOL AddGabarit (IGabaritObject * gabObj);

Синтаксис COM:

BOOL AddGabarit (IUnknown* gabObj);

Входные параметры:

gabObj - указатель на интерфейс IGabaritObject вспомогательных данных для изменения габарита документа.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.
2. Если на данное событие подписалось несколько библиотек, то заданные ими прямоугольники суммируются.

BeginPaintTmpObjects – Начало отрисовки временных объектов(фантомов)

Интерфейс...

Синтаксис Automation:

BOOL BeginPaintTmpObjects();

Синтаксис COM:

BOOL BeginPaintTmpObjects();

Возвращаемое значение:

FALSE	- запретить отрисовку временных объектов.

Примечание:

Событие можно получить в 2D документе и использовать для самостоятельной отрисовки временных объектов функциями OpenGL

BeginPaint – Начало отрисовки документа

Интерфейс...

Синтаксис Automation:

BOOL BeginPaint (IPaintObject* paintObj);

Синтаксис COM:

BOOL BeginPaint (IUnknown* paintObj);

Входные параметры:

paintObj	- указатель на интерфейс IPaintObject вспомогательных данных для отрисовки документа.

Возвращаемое значение:

FALSE	- запретить отрисовку документа.

Примечание:

1. Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.
2. Если событие вернет значение FALSE, то КОМПАС-3D не будет отрисовывать документ; библиотека в этом случае должна сама выполнить всю отрисовку, событие ClosePaint посылаться не будет.

BeginPaintGL – Начало создания листа в контексте OpenGL

Интерфейс...

Синтаксис Automation:

BOOL BeginCreateGLList (ksGLObject* glObj, long drawMode);

Синтаксис COM:

BOOL BeginCreateGLList (long drawMode);

Входные параметры:

glObj	- указатель на интерфейс ksGLObject вспомогательных данных для создания листа в контексте OpenGL; используется при работе под управлением внешнего контроллера,
drawMode	- режим отображения документа-модели KDocument3DDrawMode.

Возвращаемое значение:

TRUE	- модель продолжает отрисовываться,
FALSE	- запретить перерисовывать модель.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

CloseFrame – Закрытие окна

Интерфейс...

Синтаксис Automation:

BOOL CloseFrame();

Синтаксис COM:

BOOL CloseFrame();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

ClosePaint – Конец отрисовки документа

Интерфейс...

Синтаксис Automation:

BOOL ClosePaint (IPaintObject* paintObj);

Синтаксис COM:

BOOL ClosePaint (IUnknown* paintObj);

Входные параметры:

paintObj	- указатель на интерфейс IPaintObject вспомогательных данных для отрисовки документа.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

ClosePaintGL – Окончание создания листа в контексте OpenGL

Интерфейс...

Синтаксис Automation:

BOOL CloseCreateGLList (ksGLObject* glObj, long drawMode);

Синтаксис COM:

BOOL CloseCreateGLList (long drawMode);

Входные параметры:

glObj	- указатель на интерфейс ksGLObject вспомогательных данных для создания листа в контексте OpenGL; используется при работе под управлением внешнего контроллера,
drawMode	- режим отображения документа-модели KDocument3DDrawMode.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

ClosePaintTmpObjects – Конец отрисовки временных объектов(фантомов)

Интерфейс...

Синтаксис Automation:

BOOL ClosePaintTmpObjects();

Синтаксис COM:

BOOL ClosePaintTmpObjects();

Возвращаемое значение:

- не используется.

Примечание:

Событие можно получить в 2D документе и использовать для дорисовки временных объектов функциями OpenGL

Deactivate – Деактивация окна

Интерфейс...

Синтаксис Automation:

BOOL Deactivate();

Синтаксис COM:

BOOL Deactivate();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

MouseDblClick – Двойной щелчок кнопкой мыши

Интерфейс...

Синтаксис Automation:

BOOL MouseDblClick (short nButton, short nShiftState, long x, long y);

Синтаксис COM:

BOOL MouseDblClick (short nButton, short nShiftState, long x, long y);

Входные параметры:

nButton	- тип кнопки (LEFT_BUTTON, RIGHT_BUTTON или MIDDLE_BUTTON),
nShiftState	- комбинация нажатых системных кнопок клавиатуры (SHIFT_MASK, CTRL_MASK, ALT_MASK),
x,y	- координаты в экранных пикселах.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

Значения системных кнопок, допустимых в комбинации параметра nShiftState:

SHIFT_MASK	0x01
CTRL_MASK	0x02
ALT_MASK	0x04

Значения типов кнопок мыши для параметра nButton:

LEFT_BUTTON	0x01
RIGHT_BUTTON	0x02
MIDDLE_BUTTON	0x04

MouseDown – Нажатие кнопки мыши

Интерфейс...

Синтаксис Automation:

BOOL MouseDown (short nButton, short nShiftState, long x, long y);

Синтаксис COM:

BOOL MouseDown (short nButton, short nShiftState, long x, long y);

Входные параметры:

nButton	- тип кнопки (LEFT_BUTTON, RIGHT_BUTTON или MIDDLE_BUTTON),
nShiftState	- комбинация нажатых системных кнопок клавиатуры (SHIFT_MASK, CTRL_MASK, ALT_MASK),

x,y - координаты в экранных пикселах.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

Значения системных кнопок, допустимых в комбинации параметра nShiftState:

SHIFT_MASK	0x01
CTRL_MASK	0x02
ALT_MASK	0x04

Значения типов кнопок мыши для параметра nButton:

LEFT_BUTTON	0x01
RIGHT_BUTTON	0x02
MIDDLE_BUTTON	0x04

MouseMove – Перемещение мыши

Интерфейс...

Синтаксис Automation:

BOOL MouseDown (short nShiftState, long x, long y);

Синтаксис COM:

BOOL MouseMove (short nShiftState, long x, long y);

Входные параметры:

nShiftState - комбинация нажатых системных клавиш (SHIFT_MASK, CTRL_MASK, ALT_MASK),

x,y - координаты в экранных пикселах.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.
2. Значения системных кнопок допустимых в комбинации параметра nShiftState:

SHIFT_MASK	0x01
CTRL_MASK	0x02

ALT_MASK	0x04

MouseUp – Отпускание кнопки мыши

Интерфейс...

Синтаксис Automation:

BOOL MouseUp (short nButton, short nShiftState, long x, long y);

Синтаксис COM:

BOOL MouseUp (short nButton, short nShiftState, long x, long y);

Входные параметры:

nButton	- тип кнопки (LEFT_BUTTON, RIGHT_BUTTON или MIDDLE_BUTTON),
nShiftState	- комбинация нажатых системных кнопок клавиатуры (SHIFT_MASK, CTRL_MASK, ALT_MASK),
x,y	- координаты в экранных пикселах.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFrameNotifyEnum.

Значения системных кнопок, допустимых в комбинации параметра nShiftState:

SHIFT_MASK	0x01
CTRL_MASK	0x02
ALT_MASK	0x04

Значения типов кнопок мыши для параметра nButton:

LEFT_BUTTON	0x01
RIGHT_BUTTON	0x02
MIDDLE_BUTTON	0x04

ShowOcxTree – Активизация закладки дерева документа

Интерфейс...

Синтаксис COM:

BOOL ShowOcxTree([in]IUnknown * tree, boolean show);

Интерфейс ksDrawingObjectNotify/IDrawingObjectNotify

ksDrawingObjectNotify - интерфейс Automation

Интерфейс событий для графического объекта.

Позволяет контролировать состояние объектов графического документа.

В Automation источником событий для подписки на данный интерфейс являются объекты IDrawingObject - базового интерфейса для графических объектов (события генерируются только для этого объекта) или коллекции объектов 2D IDrawingObjects (события генерируются для всех объектов данной коллекции).

Для получения указателей на документ и объект можно использовать свойство IKompasAPIObject::Reference.

В API5 аналогом является интерфейс событий объектов графических документов ksObject2DNotify, поэтому перечисления событий совпадают с перечислением событий для объектов графических документов KsObject2DNotifyEnum.

В процессе обработки события можно получить дополнительную информацию о редактировании объекта при помощи интерфейса ksObject2DNotifyResult, который можно получить при помощи метода IKompasDocument2D::GetDrawingObjectNotifyResult.

ksDrawingObjectNotify – события

BeginCopy – Начало копирования объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginCopy (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- копировать объект,
FALSE	- запрещает копирование объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

BeginDelete – Начало удаления объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginDelete (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- удалить объект,
FALSE	- запрещает удаление объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.
3. Для обеспечения корректной обработки события удаления вида, нужно учитывать следующее:

- при удалении системного вида собственно сам системный вид не удаляется: удаляются только все объекты в нем;
- номер 0 может иметь служебный вид, создаваемый КОМПАС для режима визуального редактирования макроэлементов, это можно проверить по свойству IView1::EditMacroVisibleRegime.

BeginDestroyObject – Начало разрушения объекта

Интерфейс..

Синтаксис Automation:

VARIANT_BOOL BeginDestroyObject(VARIANT Objects);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- разрушить объект,
FALSE	- запрещает разрушение объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

BeginMove – Начало сдвига объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginMove (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- сдвинуть объект,
FALSE	- запрещает сдвиг объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

BeginProcess – Начало редактирования\создания объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginProcess (long pType, IDrawingObject* iObj);

Входные параметры:

pType - тип объекта из перечисления DrawingObjectTypeEnum,
iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE - редактировать\создать объект,
FALSE - запрещает редактирование\создание объекта объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

BeginPropertyChanged – Начало изменения свойств объекта

Интерфейс..

Синтаксис :

BOOL BeginPropertyChanged(VARIANT Objects);

Входной параметр:

Objects - указатель на объект VT_DISPATCH и/или массив объектов VT_ARRAY | VT_DISPATCH.

Возвращаемое значение:

TRUE - Разрешить изменение свойств объекта,
FALSE - Запретить изменение свойств объекта.

BeginRotate – Начало поворота объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginRotate (VARIANT Obj);

Входные параметры:

iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE - повернуть объект,
FALSE - запрещает поворот объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

BeginScale – Начало масштабирования объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginScale (VARIANT Obj);

Входные параметры:

iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE - масштабировать объект,
FALSE - запрещает масштабирование объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

BeginSymmetry – Начало симметрии объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginSymmetry (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- симметрия объекта,
FALSE	- запрещает симметрию объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

BeginTransform – Начало трансформации объекта

Интерфейс..

Синтаксис Automation:

BOOL BeginTransform (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- трансформировать объект,
FALSE	- запрещает трансформацию объекта.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

ChangeActive – Переключена активность объекта (вид, слой)

Интерфейс...

Синтаксис Automation:

BOOL ChangeActive (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

Copy – Объект скопирован

Интерфейс..

Синтаксис Automation:

BOOL Copy (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

CreateObject – Создание объекта

Интерфейс...

Синтаксис Automation:

BOOL CreateObject (VARIANT Obj);

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

Delete – Объект удален

Интерфейс..

Синтаксис Automation:

BOOL Delete (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

3. Для обеспечения корректной обработки события удаления вида, нужно учитывать следующее:

- при удалении системного вида собственно сам системный вид не удаляется: удаляются только все объекты в нем;
- номер 0 может иметь служебный вид, создаваемый КОМПАС для режима визуального редактирования макроэлементов, это можно проверить по свойству IView1::EditMacroVisibleRegime.

DestroyObject – Разрушение объекта

Интерфейс..

Синтаксис Automation:

VARIANT_BOOL DestroyObject(VARIANT Objects);

Входные параметры:

iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

EndProcess – Конец редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (long pType);

pType -тип объекта из перечисления DrawingObjectTypeEnum.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

Move – Объект сдвинут

Интерфейс..

Синтаксис Automation:

BOOL Move (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

PropertyChanged – Изменения свойств объекта

Интерфейс..

Синтаксис :

BOOL PropertyChanged(VARIANT Objects);

Входной параметр:

Objects	- указатель на объект VT_DISPATCH и/или массив объектов VT_ARRAY I VT_DISPATCH.

Возвращаемое значение:

- Не используется.

Rotate – Объект повернут

Интерфейс..

Синтаксис Automation:

BOOL Rotate (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects

Scale – Завершение масштабирования объекта

Интерфейс..

Синтаксис Automation:

BOOL Scale (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH если событие генерируется для одного объекта, - массив SafeArray объектов если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

Symmetry – Симметрия объекта

Интерфейс..

Синтаксис Automation:

BOOL Symmetry (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects/

Transform – Объект трансформирован

Интерфейс..

Синтаксис Automation:

BOOL Transform (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

UpdateObject – Редактирование объекта

Интерфейс..

Синтаксис Automation:

BOOL EndProcess (VARIANT Obj);

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Контейнером событий ksDrawingObjectNotify может использоваться объект графического документа IDrawingObject и коллекция объектов IDrawingObjects.

Интерфейс ksKompasObjectNotify/IKompasObjectNotify

ksKompasObjectNotify - интерфейс Automation
IKompasObjectNotify - интерфейс COM

Интерфейс событий приложения.

Позволяют контролировать состояние приложения.

В Automation источником событий для подписки являются:

- для API интерфейсов версии 5 - объект KompasObject.
- для API интерфейсов версии 7 - объект IApplication.

В COM источника нет.

ksKompasObjectNotify, IKompasObjectNotify – события приложения

ApplicationDestroy – Закрытие приложения

Интерфейс...

Синтаксис Automation:

BOOL ApplicationDestroy();

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL ApplicationDestroy();

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....
3. При обработке данного события необходимо отписаться от всех событий. Библиотека должна сама следить за отпиской событий.

BeginCloseAllDocument – Начало закрытия всех открытых документов

Интерфейс...

Синтаксис Automation:

BOOL BeginCloseAllDocument();

Синтаксис COM:

BOOL BeginCloseAllDocument();

Возвращаемое значение:

TRUE - закрытие всех открытых документов,
FALSE - закрытие всех открытых документов запрещено.

Примечание:

Индекс события задан в перечислении событий приложения....

BeginConvertToSavePrevious – Начало конвертации документа перед записью в предыдущую версию

Интерфейс...

Синтаксис Automation:

BOOL BeginConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPDISPATCH saveToPreviusParam);

Синтаксис COM:

BOOL BeginConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPUNKNOWN saveToPreviusParam);

Возвращаемое значение:

- Не используется.

Входные параметры:

pDoc	- документ,
docType	- тип документа.
saveVersion	- версия сохранения документа из перечисления ksSaveDocumentVersionEnum,
saveToPreviusParam	- параметры конвертации при сохранении в предыдущую версию ISaveToPreviusParam\ksSaveToPreviusParam.

BeginCreate – Начало создания документа (до диалога выбора типа)

Интерфейс...

Синтаксис Automation:

BOOL BeginCreate (long docType);

Синтаксис COM:

BOOL BeginCreate (long docType);

Входные параметры:

docName	- тип документа,
0	- если пользователь запросил диалог создания документа.

Возвращаемое значение:

TRUE	- создать документ,
FALSE	- создание документа запрещено.

Примечание:

Источником события является интерфейс KompasObject.

BeginOpenDocument – Начало открытия документа

Интерфейс...

Синтаксис Automation:

BOOL BeginOpenDocument (BSTR docName);

Входные параметры:

docName	- имя файла документа.

Возвращаемое значение:

TRUE - открыть документ,
FALSE - не открывать документ.

Синтаксис COM:

BOOL BeginOpenDocument (char * docName);

Входные параметры:

docName - имя файла документа.

Возвращаемое значение:

TRUE - открыть документ,
FALSE - не открывать документ.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

BeginOpenFile – Начало открытия документа (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginOpenFile();

Синтаксис COM:

BOOL BeginOpenFile();

Возвращаемое значение:

TRUE - вызвать диалог открытия документа,
FALSE - вызов диалога открытия документа запрещен.

Примечание:

Индекс события задан в перечислении событий приложения....

BeginRequestFiles – Запрос имен файлов

Интерфейс...

Синтаксис Automation:

BOOL BeginRequestFiles(long requestID, VARIANT * files);

Синтаксис COM:

BOOL BeginRequestFiles(long requestID, VARIANT * files);

Входные параметры:

requestID - тип запроса файлов из перечисления ksRequestFilesTypeEnum.

Выходные параметры:

files - имя файла или список SafeArray файлов.

Возвращаемое значение:

TRUE - использовать стандартный диалог выбора файлов,
FALSE - если список файлов задан - использовать файл или файлы из списка, если список файлов не задан - отмена выбора файлов.

Примечание:

1. Если выбран один файл, то в VARIANT его имя можно положить как строку, тип VARIANT-а - VT_BSTR. Если выбрано несколько файлов, то их имена в VARIANT нужно положить как массив строк тип VARIANT-а - VT_ARRAY | VT_BSTR.
2. Возможность множественного выбора и допустимые расширения файлов зависят от типа запроса файлов.
3. В данных событиях, в отличие от других событий начала какого либо действия, требуется обработка трех состояний:

- отказ пользователя или обработка события выполнена полностью на стороне библиотеки - событие возвращает FALSE;
- событие возвращает TRUE и имя файла для продолжения выполнения команды - процесс продолжается без запуска диалога выбора файла или файлов;
- событие возвращает TRUE, но имя файла не возвращается - запускается стандартный диалог выбора файла из системы КОМПАС.

ChangeActiveDocument – Переключение на другой активный документ

Интерфейс...

Синтаксис Automation:

BOOL ChangeActiveDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc - указатель на интерфейс IDispatch открытого документа,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL ChangeActiveDocument(long pDoc, int docType);

Входные параметры:

pDoc - указатель на открытый документ,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

CreateDocument – Документ создан

Интерфейс...

Синтаксис Automation:

BOOL CreateDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc - указатель на интерфейс IDispatch созданного документа,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL CreateDocument (long pDoc, int docType);

Входные параметры:

pDoc - указатель на созданный документ,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

ChangeTheme – Событие изменения темы

Версия: КОМПАС v18

Интерфейс...

Синтаксис Automation:

BOOL ChangeTheme(long newTheme);

Синтаксис COM:

BOOL ChangeTheme(long newTheme);

Возвращаемое значение:

- Не используется.

Входные параметры

newTheme - новая тема из перечисления ksThemeEnum.

EndConvertToSavePrevious – Завершение конвертации документа перед записью в предыдущую версию

Интерфейс...

Синтаксис Automation:

BOOL EndConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPDISPATCH saveToPreviusParam);

Синтаксис COM:

BOOL EndConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPUNKNOWN saveToPreviusParam);

Возвращаемое значение:

- Не используется.

Входные параметры

pDoc	- документ,
docType	- тип документа,
saveVersion	- версия сохранения документа из перечисления ksSaveDocumentVersionEnum,
saveToPreviusParam	- параметры конвертации при сохранении в предыдущую версию ISaveToPreviusParam\ksSaveToPreviusParam.

IsNeedConvertToSavePrevious – Начало сохранения документа в предыдущую версию

Интерфейс...

Синтаксис Automation:

BOOL IsNeedConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPDISPATCH saveToPreviusParam, BOOL * needConvert);

Синтаксис COM:

BOOL IsNeedConvertToSavePrevious(long pDoc, int docType, long saveVersion, LPUNKNOWN saveToPreviusParam, BOOL * needConvert);

Возвращаемое значение:

- Не используется.

Входные параметры

pDoc	- документ,
docType	- тип документа,
saveVersion	- версия сохранения документа из перечисления ksSaveDocumentVersionEnum,
saveToPreviusParam	-параметры конвертации при сохранении в предыдущую версию ISaveToPreviusParam\ksSaveToPreviusParam.

Выходной параметр:

needConvert	- требуется вернуть TRUE, если для сохранения в предыдущую версию требуется конвертация документа.

KeyDown – Клавиша нажата и удерживается нажатой

Интерфейс...

Синтаксис Automation:

BOOL KeyDown (long * key, long flags, BOOL sysKey);

Синтаксис COM:

BOOL KeyDown(long * key, long flags, BOOL sysKey);

Входные параметры:

key	- код нажатой клавиши,
flags	- флаг клавиатуры,
sysKey	- признак нажатия системной клавиши.

Выходные параметры:

key - измененный код нажатой клавиши.

Возвращаемое значение:

TRUE - разрешить обработку нажатой клавиши,
FALSE - запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения...
2. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события wm_keyup может выдаваться несколько событий wm_keydown. Значение разряда 30 параметра flags позволяет определить, было ли событие wm_keydown первым или является повторным во время удержания клавиши нажатой.
3. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

KeyPress – Одиночное нажатие клавиши

Интерфейс...

Синтаксис Automation:

BOOL KeyPress (long * key, BOOL sysKey);

Синтаксис COM:

BOOL KeyPress (long * key, BOOL sysKey);

Входные параметры:

key - код нажатой клавиши,
flags - флаг клавиатуры,
sysKey - признак нажатия системной клавиши.

Выходные параметры:

key - измененный код клавиши.

Возвращаемое значение:

TRUE - разрешить обработку нажатой клавиши,
FALSE - запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения....
2. Если syskey = false, была нажата обычная клавиша, что соответствует событию WM_KEYUP, если SYSKEY = true, была нажата системная клавиша, что соответствует событию WM_SYSKEYUP.
3. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

KeyUp – Клавиша отпущена

Интерфейс...

Синтаксис Automation:

BOOL KeyUp (long * key, long flags, BOOL sysKey);

Синтаксис COM:

BOOL KeyUp (long * key, long flags, BOOL sysKey);

Входные параметры:

key	- код нажатой клавиши,
flags	- флаг клавиатуры,
sysKey	- признак нажатия системной клавиши.

Выходные параметры:

key	- измененный код нажатой клавиши.

Возвращаемое значение:

TRUE	- разрешить обработку нажатой клавиши,
FALSE	- запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения....
2. Если syskey = false, была нажата обычная клавиша, что соответствует событию wm_keyup, если syskey = true, была нажата системная клавиша, что соответствует событию WM_KEYUP, если SYSKEY = true, была нажата системная клавиша, что соответствует событию WM_SYSKEYUP.
3. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

OpenDocument – Документ открыт

Интерфейс...

Синтаксис Automation:

BOOL OpenDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc - указатель на интерфейс IDispatch открытого документа,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL OpenDocument(long pDoc, int docType);

Входные параметры:

pDoc - указатель на открытый документ,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

Интерфейс ksLayoutSheetsNotify

Интерфейс событий для листов оформления.

Позволяют контролировать состояние листов оформления.

Источником событий для подписки на данный интерфейс являются объекты ILayoutSheets - Интерфейс коллекции листов оформления.

ksLayoutSheetsNotify – события

Add – Добавлен лист оформления

Интерфейс...

Синтаксис Automation:

BOOL Add (ILayoutSheet * PSheet);

Входные параметры:

PSheet - указатель на интерфейс ILayoutSheet добавленного листа оформления.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для листов оформления ksLayoutSheetsNotifyEnum.

Delete – Удален лист оформления

Интерфейс...

Синтаксис Automation:

BOOL Delete (ILayoutSheet * PSheet);

Входные параметры:

PSheet - указатель на интерфейс ILayoutSheet удаляемого листа оформления.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для листов оформления ksLayoutSheetsNotifyEnum.

Update – Изменены параметры листа оформления

Интерфейс...

Синтаксис Automation:

BOOL Update (ILayoutSheet * PSheet);

Входные параметры:

PSheet - указатель на интерфейс ILayoutSheet листа оформления, у которого изменились параметры.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для листов оформления ksLayoutSheetsNotifyEnum.

Интерфейс ksLibraryManagerNotify

Интерфейс событий Менеджера библиотек.

Позволяют следить и при необходимости управлять основными событиями менеджера библиотек.

Источником событий для подписки на данный интерфейс является объект ILibraryManager.

ksLibraryManagerNotify – события

AddInsert – Добавлен документ в библиотеку документов

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL AddInsert (IInsert * PInsert, BOOL create);

Синтаксис COM:

BOOL AddInsert (IUnknown * PInsert, BOOL create);

Входные параметры:

PInsert	- интерфейс вставки документа,
create	- TRUE - создали новый,
	- FALSE - добавили с диска.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

AddLibraryDescription – Добавлено описание библиотеки

Интерфейс...

Синтаксис Automation:

BOOL AddLibraryDescription (ILibrary * PLibrary);

Синтаксис COM:

BOOL AddLibraryDescription (IUnknown * PLibrary);

Входные параметры:

PLibrary	- интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

Attach – Библиотека подключена

Интерфейс...

Синтаксис Automation:

BOOL Attach (ILibrary * PLibrary);

Синтаксис COM:

BOOL Attach (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для Менеджера библиотек ksLibraryManagerNotifyEnum.

BeginAttach – Подключить библиотеку

Интерфейс...

Синтаксис Automation:

BOOL BeginAttach (ILibrary * PLibrary);

Синтаксис COM:

BOOL BeginAttach (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

TRUE - можно подключить библиотеку,
FALSE - библиотеку подключать нельзя.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

BeginDetach – Отключить библиотеку

Интерфейс...

Синтаксис Automation:

BOOL BeginDetach (ILibrary * PLibrary);

Синтаксис COM:

BOOL BeginDetach (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

TRUE - можно отключить библиотеку,
FALSE - библиотеку отключить нельзя.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

BeginExecute – Запуск выполнения команды библиотеки

Интерфейс...

Синтаксис Automation:

BOOL BeginExecute (ILibrary * PLibrary);

Синтаксис COM:

BOOL BeginExecute (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

TRUE - можно выполнять команду библиотеки,
FALSE - выполнять команду библиотеки нельзя.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

BeginInsertDocument – Запуск вставки документа из библиотеки

Интерфейс...

Синтаксис Automation:

BOOL BeginInsertDocument(LPDISPATCH PLibrary, long InsertionType, BSTR Insertion);

Синтаксис COM:

BOOL BeginInsertDocument(LPUNKNOWN PLibrary, long InsertionType, BSTR Insertion);

Возвращаемое значение:

TRUE - запретить запуск стандартного процесса вставки,
FALSE - разрешить запуск стандартного процесса вставки.

DeleteInsert – Удален документ из библиотеки документов

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL DeleteInsert (IInsert * PInsert);

Синтаксис COM:

BOOL DeleteInsert (IUnknown * PInsert);

Входные параметры:

PInsert - интерфейс вставки документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

DeleteLibraryDescription – Удалено описание библиотеки

Интерфейс...

Синтаксис Automation:

BOOL DeleteLibraryDescription (ILibrary * PLibrary);

Синтаксис COM:

BOOL DeleteLibraryDescription (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

Detach – Библиотека отключена

Интерфейс...

Синтаксис Automation:

BOOL Detach (ILibrary * PLibrary);

Синтаксис COM:

BOOL Detach (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

EditInsert – Редактирование документа из библиотеки документов

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

BOOL EditInsert (ILibrary * PLibrary, IKompasDocument * pDoc, BOOL newFrw);

Синтаксис COM:

BOOL DeleteInsert (IUnknown * PInsert);

Входные параметры:

PLibrary - интерфейс библиотеки,
pDoc - интерфейс редактируемого документа,

newFrw - TRUE - редактирование нового документа,
- FALSE - редактирование существующего документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

EndExecute – Завершение выполнения команды библиотеки

Интерфейс...

Синтаксис Automation:

BOOL EndExecute (ILibrary * PLibrary);

Синтаксис COM:

BOOL EndExecute (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

SystemControlStart – Передача управления системе

Интерфейс...

Синтаксис Automation:

BOOL SystemControlStart (ILibrary * PLibrary);

Синтаксис COM:

BOOL SystemControlStart (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

SystemControlStop – Передача управления библиотеке

Интерфейс...

Синтаксис Automation:

BOOL SystemControlStop (ILibrary * PLibrary);

Синтаксис COM:

BOOL SystemControlStop (IUnknown * PLibrary);

Входные параметры:

PLibrary - интерфейс библиотеки.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksLibraryManagerNotifyEnum.

TryExecute – Попытка выполнения команды библиотеки

Интерфейс...

Синтаксис Automation:

BOOL TryExecute(ILibrary * PLibrary, long CommandID);

Синтаксис COM

HRESULT TryExecute(ILibrary * PLibrary, long CommandID, BOOL * Result);

Возвращаемое значение:

TRUE - если нужно разрешить запуск команды.

Входные параметры:

PLibrary - указатель на интерфейс библиотеки,
CommandID - номер команды.

Интерфейс ksModelObjectNotify

Интерфейс событий для объектов документа-модели.

Описание:

Позволяют контролировать состояние объектов документа - модели.

В Automation источником событий для подписки на данный интерфейс являются объекты IModelObject (события генерируются только для этого объекта) или коллекции объектов 3D IModelObjects (события генерируются для всех объектов данной коллекции).

В процессе обработки события можно получить дополнительную информацию о редактировании объекта при помощи интерфейса IModelObjectNotifyResult.

В API 5 аналогом является Интерфейс событий объектов документа-модели ksObject3DNotify, поэтому перечисления событий совпадают с перечислением событий для объектов документа-модели KsObject3DNotifyEnum.

ksModelObjectNotify – события

BeginDelete – Начало удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginDelete (VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- удалить объект,
FALSE	- запрещает удаление объекта.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

BeginPlacementChanged – Начало изменения положения объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginPlacementChanged(VARIANT obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH если событие генерируется для одного объекта, - массив SafeArray объектов если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

BeginProcess – Начало редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess (long pType, VARIANT Obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта,
pType	-тип объекта из перечисления ksObj3dTypeEnum.

Возвращаемое значение:

TRUE	- редактировать\создать объект,
FALSE	- запрещает редактирование\создание объекта объекта.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

BeginPropertyChanged – Начало изменения свойств объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginPropertyChanged(VARIANT obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

CreateObject – Создание объекта

Интерфейс...

Синтаксис Automation:

BOOL CreateObject (VARIANT Obj);

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

Delete – Объект удален

Интерфейс...

Синтаксис Automation:

BOOL Delete (VARIANT Obj);

Входные параметры:

iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

EndProcess – Конец редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (long pType);

pType -тип объекта из перечисления ksObj3dTypeEnum.

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

Excluded – Объект исключен/включен в расчет

Интерфейс...

Синтаксис Automation:

BOOL Excluded(VARIANT obj, VARIANT_BOOL excluded);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- объект исключен из расчета,
FALSE	- объект включен в расчет.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

Hidden – Объект скрыт/показан

Интерфейс...

Синтаксис Automation:

BOOL Hidden(VARIANT obj, VARIANT_BOOL _hidden);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

TRUE	- объект скрыт,
FALSE	- объект показан.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

PlacementChanged – Изменено положения объекта

Интерфейс...

Синтаксис Automation:

BOOL PlacementChanged(VARIANT obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

PropertyChanged – Изменены свойства объекта

Интерфейс...

Синтаксис Automation:

BOOL PropertyChanged(VARIANT obj);

Входные параметры:

iObj	- VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта, - массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

UpdateObject– Редактирование объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (VARIANT Obj);

iObj - VARIANT с типом VT_DISPATCH, если событие генерируется для одного объекта,
- массив SafeArray объектов, если событие генерируется для более чем одного объекта.

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс события задан в перечислении событий для объектов документов-моделей KsObject3DNotifyEnum.
2. Контейнером событий ksModelObjectNotify может использоваться объект документа-модели IModelObject и коллекция объектов IModelObjects.

Интерфейс IProcess3DManipulatorsNotify\ksProcess3DManipulatorsNotify

Интерфейс событий манипуляторов.

Примечание:

Источником событий манипуляторов является коллекция манипуляторов IManipulators.

Версия: KOMPAS v19

ksProcess3DManipulatorsNotify – методы

RotateManipulator – Вращение манипулятора относительно оси на угол

Интерфейс...

Синтаксис Automation :

BOOL RotateManipulator(long ManipulatorId, double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle, BOOL FromEdit);

Синтаксис COM:

HRESULT RotateManipulator(long ManipulatorId, double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle, BOOL FromEdit, BOOL * Result);

Возвращаемое значение:

TRUE - если библиотека обработала событие,

FALSE - библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора
(при этом положение фантома процесса не изменится).

Входные параметры:

ManipulatorId	- идентификатор манипулятора,
X0	- первая координата точки,
Y0	- вторая координата точки,
Z0	- третья координата точки,
AxisZX	- ось X,
AxisZXY	- ось Y,
AxisZZ	- ось Z,
Angle	- угол поворота.
FromEdit	- признак поворота с помощью редактора.

MoveManipulator – Перемещение манипулятора

Интерфейс...

Синтаксис Automation:

BOOL MoveManipulator(long ManipulatorId, double VX, double VY, double VZ, double Delta, BOOL FromEdit);

Синтаксис COM:

HRESULT MoveManipulator(long ManipulatorId, double VX, double VY, double VZ, double Delta, BOOL FromEdit, BOOL * Result);

Возвращаемое значение:

TRUE - если библиотека обработала событие,
FALSE - библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора
(при этом положение фантома процесса не изменится).

Входные параметры:

ManipulatorId	- идентификатор манипулятора,
VX	- первая компонента вектора движения,
VY	- вторая компонента вектора движения,
VZ	- третья компонента вектора движения,
Delta	- значение сдвига по вектору движения,
FromEdit	- признак поворота с помощью редактора.

ClickManipulatorPrimitive – Клик или двойной клик по примитиву манипулятора

Интерфейс...

Синтаксис Automation :

BOOL ClickManipulatorPrimitive(long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType, BOOL DoubleClick);

Синтаксис COM:

BOOL ClickManipulatorPrimitive(long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType, [in] VARIANT_BOOL DoubleClick);

Возвращаемое значение:

TRUE	- если библиотека обработала событие,
FALSE	- библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId	- идентификатор манипулятора,
PrimitiveType	- примитив манипулятора из перечисления ksManipulatorPrimitiveEnum,
DoubleClick	- двойной клик.

BeginDragManipulator – Начало перемещения примитива манипулятором

Интерфейс...

Синтаксис Automation :

BOOL BeginDragManipulator (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Синтаксис COM :

BOOL BeginDragManipulator (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Возвращаемое значение:

TRUE	- если библиотека обработала событие,
FALSE	- библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId	- идентификатор манипулятора,
PrimitiveType	- примитив манипулятора из перечисления ksManipulatorPrimitiveEnum.

EndDragManipulator – Завершение перемещения примитива манипулятором

Интерфейс...

Синтаксис Automation :

BOOL EndDragManipulator (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Синтаксис COM :

BOOL EndDragManipulator (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Возвращаемое значение:

TRUE	- если библиотека обработала событие,
FALSE	- библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId	- идентификатор манипулятора,
PrimitiveType	- примитив манипулятора из перечисления ksManipulatorPrimitiveEnum,

CreateManipulatorEdit – Создание редактора для ввода значения, управляющего положением манипулятора

Интерфейс...

Синтаксис Automation:

BOOL CreateManipulatorEdit (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Синтаксис COM :

BOOL CreateManipulatorEdit (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType);

Возвращаемое значение:

TRUE	- если библиотека обработала событие,

FALSE - библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId - идентификатор манипулятора,
PrimitiveType - примитив манипулятора из перечисления ksManipulatorPrimitiveEnum,

DestroyManipulatorEdit – Удаление редактора для ввода значения, управляющего положением манипулятора

Интерфейс...

Синтаксис Automation:

BOOL DestroyManipulatorEdit (long ManipulatorId);

Синтаксис COM :

BOOL DestroyManipulatorEdit (long ManipulatorId);

Возвращаемое значение:

TRUE - если библиотека обработала событие,
FALSE - библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId - идентификатор манипулятора.

ChangeManipulatorValue – Завершение редактирования значения в редакторе манипулятора

Интерфейс...

Синтаксис Automation:

BOOL ChangeManipulatorValue (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType, double newValue);

Синтаксис COM:

BOOL ChangeManipulatorValue (long ManipulatorId, ksManipulatorPrimitiveEnum PrimitiveType, double newValue);

Возвращаемое значение:

TRUE - если библиотека обработала событие,
FALSE - библиотека не обработала событие, требуется использовать встроенную обработку событий манипулятора.

Входные параметры:

ManipulatorId - идентификатор манипулятора,
PrimitiveType - примитив манипулятора из перечисления ksManipulatorPrimitiveEnum,
newValue - новое значение.

Интерфейс ksPLMObjectNotify

Интерфейс событий объектов версионирования.

Иерархия:

IDispatch

ksPLMObjectNotify

Примечание:

Источником событий объектов версионирования является 3D документ IKompasDocument3D.

Версия: KOMPAS v19

ksPLMObjectNotify – методы

PLMChangeChanged – Изменение признака отличия

Интерфейс...

Синтаксис Automation:

BOOL PLMChangeChanged(IDispatch* Object, long newValue);

Синтаксис COM :

HRESULT PLMChangeChanged(IDispatch* Object, long newValue, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

Object - объект версионирования,
newValue - новое значение признака отличия из перечисления ksPLMChangesEnum.

PLMStatusChanged – Изменение статуса в системе версионирования

Интерфейс...

Синтаксис Automation:

BOOL PLMStatusChanged(IDispatch* Object, long newValue);

Синтаксис COM:

HRESULT PLMStatusChanged(IDispatch* Object, long newValue, BOOL * Result);

Возвращаемое значение:

TRUE - не используется.

Входные параметры:

Object - объект версионирования,
newValue - новое значение статуса из перечисления ksPLMStatusEnum.

Интерфейс ksProcess2DNotify\IProcess2DNotify

Интерфейс событий для процесса 2D.

Иерархия:

IDispatch

ksProcess2DNotify

События позволяют контролировать события процесса 2D.

Источником событий для подписки на данный интерфейс является:

IProcess2D - интерфейс процесса 2D.

События

Activate – Активизация процесса

Интерфейс...

Синтаксис Automation:

BOOL Activate();

Синтаксис COM:

BOOL Activate();

Возвращаемое значение:

Не используется.

Deactivate – Деактивизация процесса

Интерфейс...

Синтаксис Automation:

BOOL Deactivate();

Синтаксис COM

BOOL Deactivate();

Возвращаемое значение:

Не используется.

GetMouseEnterLeavePoint – Запрос параметров точек для визуального определения места применения параметра

Интерфейс...

Синтаксис Automation:

BOOL GetMouseEnterLeavePoint(LPDISPATCH Control, long BtnID, long PointIndex, LPDISPATCH Parameters);

Синтаксис COM :

BOOL GetMouseEnterLeavePoint(LPUNKNOWN control, long btnID, long pointIndex, LPUNKNOWN parameters);

Входные параметры:

index	- индекс стандартного или пользовательского набора загрузки.

Возвращаемое значение:

Не используется

Примечание

У сборки есть умолчательные стандартные наборы, индексы которых соответствуют перечислению ksLoadStateEnum. Кроме этого могут быть еще и пользовательские наборы.

EndProcess – Завершение процесса

Интерфейс...

Синтаксис Automation:

BOOL EndProcess();

Синтаксис COM:

BOOL EndProcess();

Возвращаемое значение:

Не используется.

ExecuteCommand – Выполнить команду меню процесса

Интерфейс...

Синтаксис Automation:

BOOL ExecuteCommand(long Command);

Синтаксис COM:

BOOL ExecuteCommand(long Command);

Возвращаемое значение:

TRUE - если параметры процесса изменились.

Входные параметры:

Command - номер команды меню процесса.

PlacementChange – Изменение положения

Интерфейс...

Синтаксис Automation:

BOOL PlacementChange(double X, double Y, double Angle, BOOL Dynamic);

Синтаксис COM:

BOOL PlacementChange(double X, double Y, double Angle, BOOL Dynamic);

Возвращаемое значение:

TRUE - если параметры процесса изменились.

Входные параметры:

X ,Y	- координаты курсора,
Angle	- угол,
Dynamic	TRUE - динамическое изменение положения,
	FALSE - фиксация положения.

Run – Запуск процесса

Интерфейс...

Синтаксис Automation:

BOOL Run();

Синтаксис COM:

BOOL Run();

Возвращаемое значение:

Не используется.

Stop – Остановка процесса

Интерфейс...

Синтаксис Automation:

BOOL Stop();

Синтаксис COM:

BOOL Stop();

Возвращаемое значение:

Не используется.

Интерфейс ksProcess3DNotify\IProcess3DNotify

Интерфейс событий для процесса 3D.

Иерархия:

IDispatch

ksProcess3DNotify

События позволяют контролировать события процесса 3D.

Источником событий для подписки на данный интерфейс является:

IProcess3D - интерфейс процесса 3D.

ksProcess3DNotify – методы

ProcessingGroupObjects – Обработать объекты, пришедшие из рамки

Интерфейс...

Синтаксис Automation:

BOOL ProcessingGroupObjects(VARIANT Objects, long selectionType);

Синтаксис COM:

HRESULT ProcessingGroupObjects(VARIANT Objects, long selectionType, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

Objects	- массив объектов,
selectionType	- тип выделения.

Примечание

Если процесс должен поддерживать групповое селектирование рамкой, требуется установить свойство IProcess3D::SelectionBandMode.

ksProcess3DNotify – события

Activate – Активизация процесса

Интерфейс...

Синтаксис Automation:

BOOL Activate();

Синтаксис COM:

BOOL Activate();

Возвращаемое значение:

Не используется.

CreateTakeObject – Создание объекта в подпроцессе

Интерфейс...

Синтаксис Automation:

BOOL CreateTakeObject(LPDISPATCH Object);

Синтаксис COM:

BOOL CreateTakeObject(LPUNKNOWN Object);

Возвращаемое значение:

TRUE	- если объект подходит.

Входные параметры:

Object	- указатель на на созданный объект при использовании автоматизации указатель IModelObject, IMateConstraint3D в зависимости от типа подпроцесса, при использовании COM указатель IEntity, IMateConstraint в зависимости от типа подпроцесса.

Deactivate – Деактивизация процесса

Интерфейс...

Синтаксис Automation:

BOOL Deactivate();

Синтаксис COM:

BOOL Deactivate();

Возвращаемое значение:

Не используется.

EndProcess – Завершение процесса

Интерфейс...

Синтаксис Automation:

BOOL EndProcess();

Синтаксис COM:

BOOL EndProcess();

Возвращаемое значение:

Не используется.

ExecuteCommand – Выполнить команду меню процесса

Интерфейс...

Синтаксис Automation:

BOOL ExecuteCommand(long Command);

Синтаксис COM:

BOOL ExecuteCommand(long Command);

Возвращаемое значение:

TRUE - если параметры процесса изменились.

Входные параметры:

Command - номер команды меню процесса.

FilterObject – Фильтрация объектов под курсором

Интерфейс...

Синтаксис Automation:

BOOL FilterObject(LPDISPATCH Object);

Синтаксис COM:

BOOL FilterObject(LPUNKNOWN Object);

Возвращаемое значение:

TRUE - если объект подходит.

Входные параметры:

Object - указатель на объект под курсором:
при использовании автоматизации указатель IModelObject, IPart7, IBody7, IMateConstraint3D в зависимости от типа выбранного объекта,
при использовании COM указатель IEntity, IPart, IBody, IMateConstraint в зависимости от типа выбранного объекта.

PlacementChange– Изменение положения

Интерфейс...

Синтаксис Automation:

BOOL PlacementChange(LPDISPATCH Object);

Синтаксис COM:

BOOL PlacementChange(LPUNKNOWN Object);

Возвращаемое значение:

TRUE - если параметры процесса изменились.

Входные параметры:

Object - указатель на выбранный объект:
при использовании автоматизации указатель IModelObject, IPart7, IBody7 в зависимости от типа выбранного объекта,
при использовании COM указатель IEntity, IPart, IBody в зависимости от типа выбранного объекта.

Run– Запуск процесса

Интерфейс...

Синтаксис Automation:

BOOL Run();

Синтаксис COM:

BOOL Run();

Возвращаемое значение:

Не используется.

Stop – Остановка процесса

Интерфейс...

Синтаксис Automation:

BOOL Stop();

Синтаксис COM:

BOOL Stop();

Возвращаемое значение:

Не используется.

Интерфейс ksObject2DNotifyResult IObject2DNotifyResult

Интерфейс результатов редактирования объекта.

ksObject2DNotifyResult - интерфейс Automation
IObject2DNotifyResult - интерфейс COM

В Automation получить интерфейс результата можно при помощи метода ksDocument2D::GetObject2DNotifyResult.

В COM получить интерфейс результата можно при помощи экспортной функции ksGetObject2DNotifyResult.

Примечание:

В процессе обработки события Object2DNotifyEnum можно получить информацию о редактировании объекта.

ksObject2DNotifyResult – методы

GetAngle – Получить угол поворота объекта

Интерфейс...

Синтаксис Automation:

double GetAngle();

Синтаксис COM:

double GetAngle();

Возвращаемое значение:

- угол поворота объекта.

GetCopyObject – Получить копию объекта, если выполнялась операция копирования

Интерфейс...

Синтаксис Automation:

long GetCopyObject();

Синтаксис COM:

long GetCopyObject();

Возвращаемое значение:

- указатель на объект.

GetNotifyType – Получить тип события

Интерфейс...

Синтаксис Automation:

long GetNotifyType();

Синтаксис COM:

long GetNotifyType();

Возвращаемое значение:

- тип события.

GetProcessType – Тип процесса

Интерфейс...

Синтаксис:

long GetProcessType();

Возвращаемое значение:

Тип процесса из перечисления ProcessTypeEnum - в случае успеха.

GetScale – Получить коэффициенты масштабирования по координатным осям

Интерфейс...

Синтаксис Automation:

BOOL GetScale (double *sx, double *sy);

Синтаксис COM:

BOOL GetScale (double *sx, double *sy);

Выходные параметры:

Sx - масштаб по оси X,
Sy - масштаб по оси Y.

Возвращаемое значение:

TRUE - в случае успеха,

FALSE - в случае неудачи.

GetSheetPoint – Получить координаты точки в системе координат листа

Интерфейс...

Синтаксис Automation:

BOOL GetSheetPoint (BOOL from, double *x, double *y);

Синтаксис COM:

BOOL GetSheetPoint (BOOL from, double *x, double *y);

Входные параметры:

from - TRUE - начало сдвига, центр поворота, центр масштабирования, первая точка оси симметрии, начало копирования,
- FALSE - конец сдвига, вторая точка оси симметрии, конец копирования.

Выходные параметры:

x, y, - значения координат точки в системе координат листа.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

IsCopy – Получить признак копирования исходных объектов

Интерфейс...

Синтаксис Automation:

BOOL IsCopy();

Синтаксис COM:

BOOL IsCopy();

Возвращаемое значение:

TRUE - процесс с копированием,
FALSE - процесс без копирования.

IsRedoMode – Признак работы команды Redo

Интерфейс...

Синтаксис Automation:

BOOL IsRedoMode();

Синтаксис COM:

BOOL IsRedoMode();

Возвращаемое значение:

TRUE - если работает команда Redo.

Примечание:

Свойство требуется проверять в событиях, если необходимо отличать событие, полученное от визуальной работы пользователя, от действий, вызванных командой повторить (т.е. вернуть отмененное действие).

См. также ksObject2DNotifyResult::IsUndoMode

IsUndoMode – Признак работы команды Undo

Интерфейс...

Синтаксис Automation:

BOOL IsUndoMode();

Синтаксис COM:

BOOL IsUndoMode();

Возвращаемое значение:

TRUE - если работает команда Undo.

Примечание:

Свойство требуется проверять в событиях, если необходимо отличать событие, полученное от визуальной работы пользователя, от действий, вызванных командой Отменить.

Например, при отмене создания объекта посылается событие удаления объекта.

При отмене удаления объекта посылается событие создания объекта.

См. также ksObject2DNotifyResult::IsRedoMode

Интерфейс ksPropertyManagerNotify/ IPropertyManagerNotify

ksPropertyManagerNotify - интерфейс Automation

IPropertyManagerNotify	- интерфейс COM

Интерфейс событий Панели свойств или процессных параметров.

Позволяют контролировать состояние Панели свойств.

Источником событий для подписки на данный интерфейс являются объекты:

- IPropertyManager - интерфейс панели свойств.
- IProcessParam - интерфейс параметров процесса.

ksPropertyManagerNotify, IPropertyManagerNotify – события

ButtonClick – Нажата кнопка спецпанели

Интерфейс..

Синтаксис Automation:

BOOL ButtonClick(long buttonID);

Синтаксис COM:

BOOL ButtonClick(long buttonID);

Входные параметры:

buttonID - идентификатор нажатой кнопки.

Возвращаемое значение:

Для кнопки **Справка** (pbHelp)	- TRUE - выводить стандартную справку КОМПАС, - FALSE - справку не выводить.
Для остальных	- не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ButtonUpdate – Задано состояние кнопки спецпанели

Интерфейс..

Синтаксис Automation:

BOOL ButtonUpdate (long buttonID, long *check, BOOL * enable);

Синтаксис COM:

BOOL ButtonUpdate (long buttonID, long *check, boolean * enable);

Входные параметры:

buttonID	- идентификатор кнопки,
check	- состояние кнопки (нажата/отжата),
enable	- доступность кнопки (доступна/недоступна).

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ChangeControlValue – Измененено значения элемента управления

Интерфейс..

Синтаксис Automation:

BOOL ChangeControlValue (IPropertyControl * control);

Синтаксис COM:

BOOL ChangeControlValue (IUnknown* control);

Входные параметры:

control	- указатель на интерфейс элемента панели свойств IPropertyControl.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ChangeTabExpanded – Изменение активности закладки панели свойств

Интерфейс..

Синтаксис Automation:

BOOL ChangeTabExpanded(long TabIndex);

Синтаксис COM:

BOOL ChangeTabExpanded(long TabIndex);

Возвращаемое значение:

- Не используется.

Входные параметры:

TabIndex - индекс закладки.

CommandHelp – Вызвана справка

Интерфейс...

Синтаксис Automation:

BOOL CommandHelp (long ID);

Синтаксис COM:

BOOL CommandHelp (long ID);

Входные параметры:

ID - идентификатор элемента Панели свойств, для которого вызывается справка.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ControlCommand – Нажата кнопка элемента управления

Интерфейс..

Синтаксис Automation:

BOOL ControlCommand (IPropertyControl * control, long buttonID);

Синтаксис COM:

BOOL ControlCommand (IUnknown* control , long buttonID);

Входные параметры:

control - указатель на интерфейс элемента панели свойств IPropertyControl,

buttonID - идентификатор нажатой кнопки.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

GetContextMenuType – CLLBACK для получения типа контекстного меню

Интерфейс..

Синтаксис Automation:

BOOL GetContextMenuType(long LX, long LY, long * ContextMenuType);

Синтаксис COM:

HRESULT GetContextMenuType(long LX, long LY, long * ContextMenuType, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Входные параметры:

LX - координаты курсора,
LY - координаты курсора.
ContextMenuType - тип контекстного меню.

FillContextPanel – CLLBACK для накачки контекстной панели

Интерфейс..

Синтаксис Automation

BOOL FillContextPanel(IProcessContextPanel * ContextPanel);

Синтаксис COM:

HRESULT FillContextPanel(IProcessContextPanel * ContextPanel, BOOL * Result);

Возвращаемое значение:

- Не используется.

Входные параметры:

ContextPanel - указатель на интерфейс контекстной панели,

FillContextIconMenu – CLLBACK для накачки иконок контекстной панели

Интерфейс..

Синтаксис Automation:

BOOL FillContextIconMenu(IProcessContextIconMenu * ContextMenu);

Синтаксис COM:

HRESULT FillContextIconMenu(IProcessContextIconMenu * ContextMenu, BOOL * Result);

Возвращаемое значение:

- Не используется.

Входные параметры:

ContextMenu - указатель на интерфейс контекстного меню с иконками.

EndEditItem – Завершение редактирования текста элемента списка

Интерфейс..

Синтаксис Automation:

BOOL EndEditItem(IPropertyControl * Control, long Index);

Синтаксис COM:

HRESULT EndEditItem(IPropertyControl * Control, long Index, BOOL * Result);

Возвращаемое значение:

- Не используется.

Входные параметры:

Control - указатель контрола,
Index - индекс значения.

ProcessActivate – Процесс активизирован

Интерфейс..

Синтаксис Automation:

BOOL ProcessActivate();

Синтаксис COM:

BOOL ProcessActivate();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ProcessDeactivate – Процесс деактивирован

Интерфейс..

Синтаксис Automation:

BOOL ProcessDeactivate();

Синтаксис COM:

BOOL ProcessDeactivate();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

SelectItem – Элемент списка выделен

Интерфейс..

Синтаксис Automation:

BOOL SelectItem (LPUNKNOWN Control, long Index, BOOL Select);

Синтаксис COM:

BOOL SelectItem (LPUNKNOWN Control, long Index, BOOL Select);

Входные параметры:

Control	- указатель на указатель на интерфейс элемента Панели свойств IPropertyControl,
Index	- индекс строки в раскрывающемся списке,
Select	- тип объекта: TRUE - комментарий, системная клавиша, FALSE - базовый объект.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

CheckItem – Элемент списка выбран

Интерфейс..

Синтаксис Automation:

BOOL CheckItem (LPUNKNOWN Control, long Index, BOOL Check);

Синтаксис COM:

BOOL CheckItem (LPUNKNOWN Control, long Index, BOOL Check);

Входные параметры:

Control	- указатель на указатель на интерфейс элемента Панели свойств IPropertyControl,
Index	- идентификатор кнопки,
Select	- тип объекта: TRUE - комментарий, системная клавиша, FALSE - базовый объект.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

ChangeActiveTab – Изменение активности закладки Панели свойств

Функция не поддерживается

Интерфейс..

Синтаксис Automation:

BOOL ChangeActiveTab(long TabIndex);

Синтаксис COM:

BOOL ChangeActiveTab(long TabIndex);

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

EditFocus – Установка/снятие фокуса на поле ввода

Интерфейс..

Синтаксис Automation:

BOOL EditFocus(IPropertyControl * Control, BOOL Set);

Синтаксис COM:

BOOL EditFocus(LPUNKNOWN ctrl, BOOL Set);

Входные параметры:

ctrl	- указатель на интерфейс элемента управления IPropertyControl,
Set	true - фокус установлен, false - фокус снят.

Возвращаемое значение:

Не используется.

Примечание:

Индекс события задан в перечислении событий Панели свойств ksPropertyManagerNotifyEnum.

LayoutChanged – Изменение размещения панели свойств

Функция не поддерживается

Интерфейс..

Синтаксис Automation:

BOOL LayoutChanged();

Синтаксис COM:

HRESULT LayoutChanged(BOOL * Result);

Входные параметры:

ctrl	- указатель на интерфейс IPropertyControl,
menuID	- идентификатор меню.

Возвращаемое значение:

Не используется.

UserMenuCommand – Вызов команды пользовательского меню

Интерфейс..

Синтаксис Automation:

BOOL UserMenuCommand (LPDISPATCH ctrl, long menuID);

Синтаксис COM:

BOOL UserMenuCommand (LPUNKNOWN ctrl, long menuID);

Входные параметры:

ctrl	- указатель на интерфейс IPropertyControl,
menuID	- идентификатор меню.

Возвращаемое значение:

Не используется.

Интерфейс ksPropertyUserControlNotify/ IPropertyUserControlNotify

ksPropertyUserControlNotify - интерфейс Automation
- аналога в COM нет

Интерфейс событий пользовательского элемента управления.

Позволяют контролировать состояние элемента управления OCX.

Источником событий для подписки на данный интерфейс является IPropertyUserControl - пользовательский элемент управления Панели свойств.

ksPropertyUserControlNotify – события

CreateOCX – Создан элемент управления OCX

Интерфейс...

Синтаксис Automation:

BOOL CreateOCX (LPDISPATCH iOcx);

Входные параметры:

iOcx - указатель на интерфейс IDispatch созданного элемента управления OCX.

Возвращаемое значение:

- Не используется.

Примечание:

1. При обработке этого события библиотека может подписаться на события элемента управления OCX для осуществления взаимодействия с ним. Если библиотека подписалась на события элемента управления OCX, то она должна обязательно от них отписаться при получении события DestroyOCX.
2. Индекс события задан в перечислении событий пользовательского элемента управления ksPropertyUserControlNotifyEnum.

DestroyOCX – Элемент управления OCX удален

Интерфейс...

Синтаксис Automation:

BOOL DestroyOCX();

Возвращаемое значение:

- Не используется.

Примечание:

1. При обработке этого события библиотека должна отписаться от событий элемента управления OCX, если она на них подписывалась в событии CreateOCX.
2. Индекс события задан в перечислении событий пользовательского элемента управления ksPropertyUserControlNotifyEnum.

Интерфейс ksContentDialogNotify

Интерфейс событий диалога с произвольным контентом IContentDialogParam.

Иерархия:

IDispatch

ksContentDialogNotify

Методы интерфейса позволяют обрабатывать события диалога с произвольным наполнением.

Источником событий для подписки является объект IContentDialogParam.

КОМПАС версия v18

ksContentDialogNotify – методы

ButtonUpdate – Установка состояния кнопки панели

Интерфейс...

Синтаксис Automation:

BOOL ButtonUpdate(long ButtonID, long * Check, BOOL * Enable);

Синтаксис COM:

HRESULT ButtonUpdate(long ButtonID, long * Check, BOOL * Enable, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

ButtonID - идентификатор команды панели диалога,
Check - состояние кнопки (Нажата\Отжата),
Enable - доступность команды.

CreateContentCallback – Создание контента

Интерфейс...

Синтаксис Automation:

BOOL CreateContentCallback(OLE_HANDLE ParentHwnd, OLE_HANDLE * NewContentHwnd);

Синтаксис COM:

HRESULT CreateContentCallback(OLE_HANDLE ParentHwnd, OLE_HANDLE * NewContentHwnd, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

ParentHwnd - родительское окно.

Выходные параметры:

NewContentHwnd - новое окно контента.

DestroyContent – Удаление контента

Интерфейс...

Синтаксис Automation:

BOOL DestroyContent();

Синтаксис COM:

HRESULT DestroyContent(BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

ExecuteCommand – Выполнить команду

Интерфейс...

Синтаксис Automation:

BOOL ExecuteCommand(long ButtonID);

Синтаксис COM:

HRESULT ExecuteCommand(long ButtonID, BOOL * Result);

Возвращаемое значение:

TRUE - в случае успешного завершения.

Входные параметры:

ButtonID - идентификатор команды панели диалога.

Интерфейс ksSpecificationDescriptionNotify

Интерфейс событий описания спецификации.

Позволяют контролировать редактирование описания спецификации.

Источником событий для подписки на данный интерфейс являются объекты:

- ISpecificationDescriptions - Интерфейс коллекции описаний спецификации,
- ISpecificationDescription - Интерфейс описания спецификации.

ksSpecificationDescriptionNotify – события

BeginCalcPositions – Начало расчета позиций

Интерфейс...

Синтаксис Automation:

BOOL BeginCalcPositions();

Возвращаемое значение:

TRUE	- расчет позиций,
FALSE	- запрещает расчет позиций.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

BeginCreateObject – Начало создания объекта СП (до диалога выбора раздела)

Интерфейс...

Синтаксис Automation:

BOOL BeginCreateObject (long typeObj);

Входные параметры:

typeObj	0 - базовый объект, 1 - объект вспомогательный, -1 - тип объекта не определен.

Возвращаемое значение:

TRUE	- создать объект спецификации,
FALSE	- запрещает создание объекта спецификации.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

CalcPositions – Проведен расчет позиций

Интерфейс...

Синтаксис Automation:

BOOL CalcPositions();

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

ChangeCurrentSpcDescription – Изменилось текущее описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL ChangeCurrentSpcDescription (ISpecificationDescription * Descr);

Входные параметры:

Descr - указатель на интерфейс ISpecificationDescription текущего описания спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

SpcDescriptionAdd – Добавилось описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionAdd (ISpecificationDescription * Descr);

Входные параметры:

Descr - указатель на интерфейс ISpecificationDescription добавленного описания спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

SpcDescriptionBeginEdit – Начало редактирования описания спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionBeginEdit (ISpecificationDescription * Descr);

Входные параметры:

Descr	- указатель на интерфейс ISpecificationDescription редактируемого описания спецификации.

Возвращаемое значение:

TRUE	- редактировать описание спецификации,
FALSE	- запрещает редактирование описания спецификации.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

SpcDescriptionEdit – Отредактировано описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionEdit (ISpecificationDescription * Descr, BOOL isOk);

Входные параметры:

Descr	- указатель на интерфейс ISpecificationDescription редактируемого описания спецификации,
isOk	- результат изменения описания спецификации.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

SpcDescriptionRemove – Удалилось описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionRemove (ISpecificationDescription * Descr);

Входные параметры:

Descr - указатель на интерфейс ISpecificationDescription удаляемого описания спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

Synchronization – Синхронизация проведена

Интерфейс...

Синтаксис Automation:

BOOL Synchronization();

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

SynchronizationBegin – Начало синхронизации

Интерфейс...

Синтаксис Automation:

BOOL SynchronizationBegin();

Возвращаемое значение:

TRUE - синхронизировать,

FALSE - запрещает синхронизацию.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

TuningSpcStyleBeginChange – Начало изменения настроек спецификации

Интерфейс...

Синтаксис Automation:

BOOL TuningSpcStyleBeginChange(ISpecificationDescription * Descr);

Входные параметры:

Descr - указатель на интерфейс ISpecificationDescription редактируемого описания спецификации.

Возвращаемое значение:

TRUE - изменить настройки спецификации,
FALSE - запрещает изменения настроек спецификации.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

TuningSpcStyleChange – Настройки спецификации изменились

Интерфейс...

Синтаксис Automation:

BOOL TuningSpcStyleChange (ISpecificationDescription * Descr, BOOL isOk);

Входные параметры:

Descr - указатель на интерфейс ISpecificationDescription редактируемого описания спецификации.
isOk - результат изменения настроек спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для для Менеджера библиотек ksSpecificationNotifyEnum.

Интерфейс ksSpecificationObjectNotify

Интерфейс событий объектов спецификации.

Предназначен для того, чтобы следить и при необходимости управлять редактированием объектов спецификации.

Источником событий для подписки на данный интерфейс являются объекты:

- ISpecificationObject - Интерфейс объекта спецификации (базового или вспомогательного),
- ISpecificationBaseObjects - Интерфейс коллекции базовых объектов спецификации,
- ISpecificationCommentObjects - Интерфейс коллекции вспомогательных объектов спецификации.

ksSpecificationObjectNotify – события

BeginDelete – Начало удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginDelete (ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject удаляемого объекта спецификации.

Возвращаемое значение:

TRUE - удалить объект,
FALSE - запрещает удаление объекта.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

BeginGeomChange – Начало изменения геометрии объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL BeginGeomChange(ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,

Возвращаемое значение:

TRUE - изменить геометрию объекта спецификации,
FALSE - запрещает изменение геометрии объекта спецификации.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

BeginProcess – Начало редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess(long pType, ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,
pType - тип объекта из перечисления ksSpecificationObjectTypeEnum(SPC_BASE_OBJECT, SPC_COMMENT).

Возвращаемое значение:

TRUE - редактировать\создать объект,
FALSE - запрещает редактирование\создание объекта объекта.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

CellBeginEdit – Начало редактирования в ячейке

Интерфейс...

Синтаксис Automation:

BOOL CellBeginEdit(ISpecificationObject * obj, long number);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,
number - номер редактируемой ячейки.

Возвращаемое значение:

TRUE - редактировать ячейку,
FALSE - запрещает редактировать ячейку.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

CellDblClick – Двойной щелчок в ячейке

Интерфейс...

Синтаксис Automation:

BOOL CellDblClick(ISpecificationObject * obj, long number);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,
number - номер редактируемой ячейки.

Возвращаемое значение:

TRUE - обработать двойной клик в ячейке,
FALSE - запрещает обработку двойного клика в ячейке.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

ChangeCurrent – Изменился текущий объект

Интерфейс...

Синтаксис Automation:

BOOL ChangeCurrent(ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject текущего объекта спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

CreateObject – Создание объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL CreateObject(ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject созданного объекта спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

Delete – Удаление объекта

Интерфейс...

Синтаксис Automation:

BOOL Delete (ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject удаляемого объекта спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

DocumentAdd – Добавление документа в объекте спецификации

Интерфейс...

Синтаксис Automation:

BOOL DocumentAdd(ISpecificationObject * obj, BSTR docName);

Входные параметры:

obj	- указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,
docName	- имя файла добавленного документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

DocumentBeginAdd – Начало добавления документа

Интерфейс...

Синтаксис Automation:

BOOL DocumentBeginAdd(ISpecificationObject * obj);

Входные параметры:

obj	- указатель на интерфейс ISpecificationObject редактируемого объекта спецификации.

Возвращаемое значение:

TRUE	- добавить документ сборочного чертежа,
FALSE	- запрещает добавление документа сборочного чертежа.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

DocumentRemove – Удаление документа из объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL DocumentRemove(ISpecificationObject * obj, BSTR docName);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации,
docName - имя файла удаляемого документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

EndProcess – Конец редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess(long pType);

Входные параметры:

pType - тип процесса (SPC_BASE_OBJECT, SPC_COMMENT).

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

GeomChange – Геометрия объекта спецификации изменилась

Интерфейс...

Синтаксис Automation:

BOOL GeomChange(ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

UpdateObject – Редактирование объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL UpdateObject(ISpecificationObject * obj);

Входные параметры:

obj - указатель на интерфейс ISpecificationObject редактируемого объекта спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для объектов спецификации ksSpcObjectNotifyEnum.

Интерфейс ksViewsAndLayersManagerNotify/ IViewsAndLayersManagerNotify

ksViewsAndLayersManagerNotify	- интерфейс Automation
IViewsAndLayersManagerNotify	- интерфейс COM

Интерфейс событий менеджера видов и слоев.

Позволяют контролировать начало и завершение работы с диалогом менеджера видов и слоев.

Источником событий для подписки на данный интерфейс являются объекты:

IViewsAndLayersManager - Менеджер слоев и видов графического документа.

ksViewsAndLayersManagerNotify/ IViewsAndLayersManagerNotify – события

BeginEdit – Начато редактирование

Интерфейс...

Синтаксис Automation:

BOOL BeginEdit();

Синтаксис COM:

BOOL BeginEdit();

Возвращаемое значение:

FALSE - запретить редактирование через диалог.

Примечание:

Индекс события задан в перечислении событий для менеджера видов и слоев ksViewsAndLayersManagerNotifyEnum.

EndEdit – Завершено редактирование

Интерфейс..

Синтаксис Automation:

BOOL EndEdit (BOOL isOk);

Синтаксис COM:

BOOL EndEdit (BOOL isOk);

Входные параметры:

isOk - TRUE диалог закрыт по кнопке **OK**,
- FALSE диалог закрыт по кнопке **Отмена**.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий для менеджера видов и слоев ksViewsAndLayersManagerNotifyEnum.

Интерфейс ksFindObjectParametersNotify

Интерфейс фильтрации объектов в функциях поиска объектов под курсором.

Иерархия:

IDispatch

ksFindObjectParametersNotify

Версия: KOMPAS v19

ksFindObjectParametersNotify – методы

FilterObject – Фильтрация объектов

Интерфейс..

Синтаксис Automation:

BOOL FilterObject(IDispatch* Object);

Синтаксис COM :

HRESULT FilterObject(IDispatch* Object, BOOL * Result);

Возвращаемое значение:

TRUE	- если объект подходит.

Входные параметры:

Object	- указатель на объект под курсором.

Конвертер файлов КОМПАС

Интерфейс IConverter

Конвертер файлов КОМПАС.

Иерархия:

IKompasAPIObject

IConverter

Примечание:

1. Интерфейс позволяет управлять конвертером системы КОМПАС, задавать параметры конвертирования, получать фильтр и выполнять конвертацию документа системы КОМПАС.
2. Данный интерфейс может быть получен от интерфейса приложения IApplication с помощью свойства IApplication::Converter.

IConverter – методы

Convert – Запустить процесс конвертации

Интерфейс...

Синтаксис Automation:

long * Convert (BSTR inputFile, BSTR outfile, long command, BOOL showParam);

Синтаксис COM:

HRESULT Convert ([in] BSTR inputFile,
[in] BSTR outfile,
[in] long command,
[in] VARIANT_BOOL showParam,
[out, retval] long* Result);

Входные параметры:

inputFile	- имя исходного файла документа,
outfile	- новое имя файла документа,
command	- номер команды,
showParam	- TRUE - выдавать диалог параметров конвертации,
	FALSE - не выдавать диалог параметров конвертации.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если имя исходного файла документа не задано, то берется текущий документ. Если новое имя файла документа не задано, то конвертация будет происходить в новый документ системы КОМПАС.

ConverterParameters – Получить интерфейс параметров конвертирования

Интерфейс...

Синтаксис Automation:

LPDISPATCH ConverterParameters (long command);

Синтаксис COM:

HRESULT ConverterParameters ([in] long command, [out, retval] IUnknown ** PVal);

Входные параметры:

command	- номер команды, для которой необходимо получить интерфейс параметров.

Возвращаемое значение:

указатель на интерфейс IUnknown	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Данный метод позволяет получить интерфейс параметров конвертирования, который объявлен, реализован и поддерживается в самом конвертере.

GetFilter – Получить фильтр и номер команды по типу документа

Интерфейс...

Синтаксис Automation:

BSTR * GetFilter (long docType, BOOL saveAs, long * command);

Синтаксис COM:

HRESULT GetFilter ([in] long docType,
[in] VARIANT_BOOL saveAs,
[out] long* command,
[out, retval] BSTR* Result);

Входные параметры:

docType - при импорте в КОМПАС-документ - тип документа из перечисления DocumentTypeEnum,
- при экспорте КОМПАС-документа - тип документа, в который производится экспорт, перечисление типов документов, должно быть задано в конверторе. Например, для конвертора в DXF и DWG используются типы:
#define FORMAT_DXF 1 // ID команды для работы с DXF
#define FORMAT_DWG 2 // ID команды для работы с DWG

saveAs - TRUE - экспорт документа КОМПАС,
- FALSE - импорт в документ КОМПАС.

Выходные параметры:

command - номер команды.

Возвращаемое значение:

фильтр - строка со справочным значением (например 'КОМПАС-Фрагменты (*.myfrw)|*.myfrw|'), которое используется в диалогах **Открыть файл** и **Сохранить файл**.

VisualEditConvertParam – Запустить визуальное редактирование параметров конвертации

Интерфейс...

Синтаксис Automation:

BOOL * VisualEditConvertParam(OLE_HANDLE parentHwnd, long command);

Синтаксис COM:

HRESULT VisualEditConvertParam ([in] OLE_HANDLE parentHwnd,
[in] long command,
[out, retval] VARIANT_BOOL * val);

Входные параметры:

parentHwnd - дескриптор окна родителя,
command - номер команды.

Возвращаемое значение:

TRUE - выход из диалога по кнопке ОК,
FALSE - выход из диалога по кнопке Отмена.

Печать

Интерфейс IPrintJob

Справка системы КОМПАС...

KOMPAS.chm::/638_76_10_Zadanie_na_pechatq.htm

Интерфейс задания на печать.

Иерархия:

IDispatch

IKompasAPIObject

IPrintJob

IPrintJob_OutputParameters

Описание:

Интерфейс позволяет устанавливать и получать параметры задания на печать.

Термины:

Лист - лист документа.

Страница устройства печати - область печати при заданном размере листов бумаги печатающего устройства.

Примечание:

1. Интерфейс можно получить у интерфейса приложения КОМПАС с помощью свойства IApplication::PrintJob.
2. Посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif) у данного интерфейса можно получить дополнительный интерфейс IPrintJob_OutputParameters.

IPrintJob – свойства

PagesCount – Количество страниц печати

Интерфейс...

Тип данных: long

Синтаксис Automation:

PagesCount Object.PagesCount	=	Получить свойство (*)
PagesCount Object.GetPagesCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_PagesCount(&PagesCount)	Получить свойство

Примечание:

1. Свойство позволяет получать текущее количество страниц устройства печати с учетом выключенных.
2. Свойство доступно только для чтения.

PagePrintableFlag – Признак печатаемости страницы по ее индексам

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PagePrintableFlag = Object.PagePrintableFlag(WPage, HPage)	=	Получить свойство (*)
Object.PagePrintableFlag(WPage, HPage) = PagePrintableFlag		Установить свойство (*)
PagePrintableFlag = Object.GetPagePrintableFlag(WPage, HPage)	=	Получить свойство (**)
Object.SetPagePrintableFlag(WPage, HPage, DimensionLineScale)		Установить свойство (**)

Синтаксис COM:

Object.get_PagePrintableFlag(WPage, HPage, &PagePrintableFlag)	Получить свойство
Object.put_PagePrintableFlag(WPage, HPage, PagePrintableFlag)	Установить свойство

Входные параметры:

long WPage	- индекс страницы по горизонтали,
long HPage	- индекс страницы по вертикали.

Примечание:

1. Свойство позволяет включить/отключить печать страницы по ее индексам (по горизонтали и вертикали). Если индексы заданы вне диапазона, возвращается FALSE.
2. Состояние страницы сбросится, если в результате перемещения листов какой-либо из ее индексов станет не действительным.

SheetsCoun – Количество листов

Интерфейс...

Тип данных: long

Синтаксис Automation:

SheetsCount = Object.SheetsCount	=	Получить свойство (*)
SheetsCount = Object.GetSheetsCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_SheetsCount(&SheetsCount)	Получить свойство

Примечание:

1. Свойство позволяет получать количество загруженных листов документов в задании на печать.
2. Свойство доступно только для чтения.

Sheet – Интерфейс листа документа

Интерфейс...

Тип данных: указатель на интерфейс листа документа IPrintJob_Sheet

Синтаксис Automation:

Sheet = Object.Sheet (Index)	Получить свойство (*)
Sheet = Object.GetSheet(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Sheet(Index, &Sheet)	Получить свойство

Входные параметры:

long Index - индекс листа, нумерация начинается с единицы.

Примечание:

1. Свойство позволяет получать интерфейс листа документа.
2. Свойство доступно только для чтения.

IPrintJob – методы

AddSheets – Добавить листы указанного документа

Интерфейс...

Синтаксис Automation:

BOOL AddSheets(BSTR FileName,
VARIANT Sheets,
long Range);

Синтаксис COM:

HRESULT AddSheets(BSTR FileName,
VARIANT Sheets,
ksSheetsRangeEnum Range,
BOOL * Result);

Входные параметры:

FileName	- имя файла, из которого надо добавить листы,
Sheets	- массив VT_ARRAY I VT_DISPATCH листов документа,
Range	- тип диапазона страниц из перечисления ksSheetsRangeEnum.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет добавить в задание на печать листы указанного документа.
2. Если вместо имени файла передается пустая строка, то добавляются листы активного документа.

Clear – Очистить здание на печать

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

HRESULT Clear(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE	- в случае неудачи.

Примечание:

Метод позволяет очистить задание на печать и привести все настройки в умолчательное состояние.

Load – Загрузить задание на печать из файла

Интерфейс...

Синтаксис Automation:

```
BOOL Load( BSTR FileName,
VARIANT * Errors );
```

Синтаксис COM:

```
HRESULT Load( BSTR FileName,
VARIANT * Errors,
BOOL  * Result );
```

Входные параметры:

FileName	- имя файла задания на печать.

Выходные параметры:

Errors	- массив VT_ARRAY I VT_BSTR сообщений об ошибках.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Метод позволяет загрузить задание на печать из файла *.pjd.
2. FALSE возвращается в следующих случаях:

- файл не найден,
- файл поврежден,
- файл имеет неправильный формат,
- загрузка файла была прервана из-за отсутствия данных,
- ни одного листа ни одного документа загрузить в задание не удалось.

3. Если задание загружено, возвращается TRUE, даже если при загрузке возникли некритические ошибки.

Execute – Отправить задание на печать на устройство печати

Интерфейс...

Синтаксис Automation:

BOOL Execute(BSTR OutputFileName);

Синтаксис COM:

HRESULT Execute(BSTR OutputFileName,
BOOL * Result);

Входные параметры:

OutputFileName	- имя устройства печати.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет отправить задание на печать на устройство печати.
2. Чтобы отправить задание на печать на текущий принтер, следует в качестве параметра OutputFileName прислать пустую строку.

GetPageGabarites – Получить габариты страницы устройства печати

Интерфейс...

Синтаксис Automation:

BOOL GetPageGabarites(double * Width
double * Height);

Синтаксис COM:

HRESULT GetPageGabarites(double * Width,
double * Height,
BOOL * Result);

Выходные параметры:

Width	- ширина страницы,
Height	- высота страницы.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE - в случае неудачи (если листа с таким номером не существует).

Примечание:

Метод позволяет получить габариты страницы устройства печати.

GetPagesMapGabarites – Получить размеры карты страниц

Интерфейс...

Синтаксис Automation:

BOOL GetPagesMapGabarites(long * WPageCount,
long * HPageCount);

Синтаксис COM:

HRESULT GetPagesMapGabarites(long * WPageCount,
long * HPageCount,
BOOL * Result);;

Выходные параметры:

WPageCount - количество страниц по горизонтали,
HPageCount - количество страниц по вертикали.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет получить размеры карты страниц - количество страниц по вертикали и горизонтали.

RemoveSheets – Удалить лист из задания на печать

Интерфейс...

Синтаксис Automation:

BOOL RemoveSheets(VARIANT Index);

Синтаксис COM:

HRESULT RemoveSheets(VARIANT Index,
BOOL * Result);

Входные параметры:

Index - индекс листа.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи (если листа с таким номером не существует).

Примечание:

Метод позволяет удалить лист из задания на печать.

Save – Сохранить задание на печать в файл

Интерфейс...

Синтаксис Automation:

```
BOOL Save( BSTR  FileName,
VARIANT * Errors );
```

Синтаксис COM:

```
HRESULT Save( BSTR FileName,
VARIANT * Errors,
BOOL  * Result );
```

Входные параметры:

FileName	- имя файла задания на печать.

Выходные параметры:

Errors	- массив VT_ARRAY | VT_BSTR, если есть ошибки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет сохранить задание на печать в файл *.pjd.

ShowPreviewWindow – Показать окно предварительного просмотра перед печатью

Интерфейс...

Синтаксис Automation:

```
BOOL ShowPreviewWindow();
```

Синтаксис COM:

```
HRESULT ShowPreviewWindow( BOOL * Result );
```

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Метод позволяет показать окно предварительного просмотра перед печатью с текущим состоянием задания на печать.
2. После закрытия окна предварительного просмотра задание на печать очищается, аналогично вызову метода IPrintJob::Clear.

SpecialExecute – Отправить задание на специальную печать

Интерфейс...

Синтаксис Automation:

BOOL SpecialExecute(BSTR OutputFileName);

Синтаксис COM:

HRESULT SpecialExecute(BSTR OutputFileName, BOOL * Result);

Входные параметры:

OutputFileName	- имя устройства печати.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Метод позволяет отправить задание на полистную печать на устройство печати.

Примечание:

Чтобы отправить задание на печать на текущий принтер, следует в качестве параметра OutputFileName прислать пустую строку.

Интерфейс IPrintJob_Sheet

Справка системы КОМПАС...

`KOMPAS.chm::/638_76_10_Zadanie_na_pechatq.htm`

Задание на печать: Интерфейс листа документа.

Иерархия:

IDispatch

IKompasAPIObject

IPrintJob_Sheet

Описание:

Интерфейс позволяет устанавливать и получать параметры листа документа в задании на печать.

Примечание:

Интерфейс можно получить у интерфейса задания на печать с помощью свойства IPrintJob::Sheet.

IPrintJob_Sheet – свойства

ClipFlag – Получить признак вывода части листа документа

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
ClipFlag = Object.ClipFlag          Получить свойство (* )
Object.ClipFlag = ClipFlag          Установить свойство (* )
ClipFlag = Object.GetClipFlag()     Получить свойство (**)
Object.SetClipFlag( ClipFlag )      Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ClipFlag( &ClipFlag )    Получить свойство
Object.put_ClipFlag(  ClipFlag )    Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать признак вывода части листа документа.

DocumentName – Имя документа-владельца листа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
DocumentName                =   Получить свойство (* )
Object.DocumentName
DocumentName                =   Получить свойство (**)
Object.GetDocumentName()
```

Синтаксис COM:

Object.get_DocumentName(&DocumentName)	Получить свойство

Примечание:

1. Свойство позволяет получать имя документа-владельца листа.
2. Свойство доступно только для чтения.

GetGabarites – Получить габариты листа

Интерфейс...

Синтаксис Automation:

BOOL GetGabarites(double * Width,
double * Height);

Синтаксис COM:

HRESULT GetGabarites(double * Width,
double * Height,
BOOL * Result);

Выходные параметры:

Width	- ширина листа,
Height	- высота листа.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить габариты листа.

Number – Номер листа документа

Интерфейс...

Тип данных: long

Синтаксис Automation:

Number = Object.Number	Получить свойство (*)
Number = Object.GetNumber()	Получить свойство (**)

Синтаксис COM:

Object.get_Number(&Number)	Получить свойство

Примечание:

1. Свойство позволяет получать номер листа документа.
2. Свойство доступно только для чтения.

Orientation – Получить текущую ориентацию листа

Интерфейс...

Тип данных: из перечисления ksAngleEnum

Синтаксис Automation:

```
Orientation = Object.Orientation      Получить свойство (* )
Object.Orientation = Orientation      Установить свойство (* )
Orientation                      =    Получить свойство (**)
Object.GetOrientation()
Object.SetOrientation(                Установить свойство (**)
Orientation )
```

Синтаксис COM:

```
Object.get_Orientation(               Получить свойство
&Orientation )
Object.put_Orientation(               Установить свойство
Orientation )
```

Примечание:

Свойство позволяет устанавливать и получать текущую ориентацию листа.

Scale – Задать масштаб листа

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Scale = Object.Scale                  Получить свойство (* )
Object.Scale = Scale                  Установить свойство (* )
Scale = Object.GetScale()             Получить свойство (**)
Object.SetScale( Scale )              Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Scale( &Scale )            Получить свойство
Object.put_Scale(  Scale )            Установить свойство
```

Примечание:

Свойство позволяет устанавливать и получать масштаб листа.

X – Координата X левого нижнего угла листа

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство (*)
Object.X = X	Установить свойство (*)
X = Object.GetX()	Получить свойство (**)
Object.SetX(X)	Установить свойство (**)

Синтаксис COM:

Object.get_X(&X)	Получить свойство
Object.put_X(X)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату X левого нижнего угла листа.

Y – Координата Y левого нижнего угла листа

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство (*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство (**)
Object.SetY(Y)	Установить свойство (**)

Синтаксис COM:

Object.get_Y(&Y)	Получить свойство
Object.put_Y(Y)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать координату Y левого нижнего угла листа.

IPrintJob_Sheet – методы

GetClipFrameGabarites – Получить выводимые на печать габариты листа

Интерфейс...

Синтаксис Automation:

BOOL GetClipFrameGabarites(double * Left,
double * Bottom,
double * Right,
double * Top);

Синтаксис COM:

HRESULT GetClipFrameGabarites(double * Left,
double * Bottom,
double * Right,
double * Top,
BOOL * Result);

Выходные параметры:

Left, Bottom, Right, Top - габариты рамки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод позволяет получить выводимые на печать габариты листа.

SetClipFrameGabarites – Установить выводимые на печать габариты листа

Интерфейс...

Синтаксис Automation:

BOOL SetClipFrameGabarites(double Left,
double Bottom,
double Right,
double Top);

Синтаксис COM:

HRESULT SetClipFrameGabarites(double Left,
double Bottom,
double Right,
double Top,
BOOL * Result);

Входные параметры:

Left, Bottom, Right, Top - габариты рамки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Метод позволяет установить выводимые на печать габариты листа.
2. Никакой из габаритов рамки не может быть больше соответствующего ему габарита листа.

Интерфейс IPrintJob_OutputParameters

Справка системы КОМПАС...

KOMPAS.chm::/638_76_10_Zadanie_na_pechatq.htm

Задание на печать:Интерфейс параметров вывода.

Иерархия:

IDispatch

IKompasAPIObject

IPrintJob_OutputParameters

Описание:

Интерфейс позволяет устанавливать и получать параметры вывода задания на печать.

Примечание:

Интерфейс является дополнительным к интерфейсу IPrintJob.

IPrintJob_OutputParameters – свойства

AccuracyModelOutput – Точность вывода моделей

Интерфейс...

Тип данных: long

Синтаксис Automation:

AccuracyModelOutput = Object.AccuracyModelOutput	=	Получить свойство (*)
Object.AccuracyModelOutput = AccuracyModelOutput	=	Установить свойство (*)
AccuracyModelOutput = Object.GetAccuracyModelOutput ()	=	Получить свойство (**)
Object.SetAlternativeFillingOutput(AlternativeFillingOutput)		Установить свойство (**)

Синтаксис COM:

Object.get_AccuracyModelOutput(&AccuracyModelOutput)	Получить свойство
Object.put_AccuracyModelOutput(AccuracyModelOutput)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать точность вывода моделей.
2. Если задаваемое значение находится вне диапазона, то его значение корректируется к ближайшему (минимальному или максимальному) значению.

AlternativeFillingOutput – Альтернативный способ вывода заливок

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AlternativeFillingOutput = Object.AlternativeFillingOutput	=	Получить свойство (*)
Object.AlternativeFillingOutput = AlternativeFillingOutput	=	Установить свойство (*)
AlternativeFillingOutput = Object.GetAlternativeFillingOutput()	=	Получить свойство (**)
Object.SetAlternativeFillingOutput(AlternativeFillingOutput)		Установить свойство (**)

Синтаксис COM:

Object.get_AlternativeFillingOutput(&AlternativeFillingOutput)	Получить свойство
Object.put_AlternativeFillingOutput(AlternativeFillingOutput)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак альтернативного способа вывода заливок.

AutoScale – Автоподгонка масштаба при добавлении листов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

AutoScale = Object.AutoScale	Получить свойство (*)
Object.AutoScale = AutoScale	Установить свойство (*)
AutoScale = Object.GetAutoScale()	Получить свойство (**)
Object.SetAutoScale(AutoScale)	Установить свойство (**)

Синтаксис COM:

Object.get_AutoScale(&AutoScale)	Получить свойство
Object.put_AutoScale(AutoScale)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак автоподгонки масштаба при добавлении листов.

CatchSpacing – Зазор между листами

Интерфейс...

Тип данных: long

Синтаксис Automation:

CatchSpacing = Object.CatchSpacing	=	Получить свойство (*)
Object.CatchSpacing = CatchSpacing	=	Установить свойство (*)
CatchSpacing = Object.GetCatchSpacing()	=	Получить свойство (**)
Object.SetCatchSpacing(CatchSpacing)		Установить свойство (**)

Синтаксис COM:

Object.get_CatchSpacing(&CatchSpacing)	Получить свойство
Object.put_CatchSpacing(CatchSpacing)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать зазор между листами при привязке к углам листов, мм.

CollateCopies – Копии в подбор

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

CollateCopies = Object.CollateCopies	=	Получить свойство (*)
Object.CollateCopies = CollateCopies	=	Установить свойство (*)
CollateCopies = Object.GetCollateCopies()	=	Получить свойство (**)
Object.SetCollateCopies(CollateCopies)		Установить свойство (**)

Синтаксис COM:

Object.get_CollateCopies(&CollateCopies)	Получить свойство
Object.put_CollateCopies(CollateCopies)	Установить свойство

Color – Цвет вывода

Интерфейс...

Тип данных: из перечисления ksOutputColorTypeEnum

Синтаксис Automation:

Color = Object.Color	Получить свойство (*)
Object.Color = Color	Установить свойство (*)
Color = Object.GetColor()	Получить свойство (**)
Object.SetColor(Color)	Установить свойство (**)

Синтаксис COM:

Object.get_Color(&Color)	Получить свойство
Object.put_Color(Color)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать цвет вывода.

DefaultScale – Масштаб листов по умолчанию

Интерфейс...

Тип данных: double

Синтаксис Automation:

DefaultScale = Получить свойство Object.DefaultScale (*)
Object.DefaultScale = DefaultScale Установить свойство (*)
DefaultScale = Object.GetDefaultScale() Получить свойство (**)
Object.SetDefaultScale(DefaultScale) Установить свойство (**)

Синтаксис COM:

Object.get_DefaultScale(&DefaultScale) Получить свойство
Object.put_DefaultScale(DefaultScale) Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать масштаб листов по умолчанию.
2. Если задаваемое значение находится вне диапазона (1e-006 : 1e+006), то его значение корректируется к ближайшему (минимальному или максимальному) значению.

Hooking – Диапазон привязки к узлам страниц

Интерфейс...

Тип данных: long

Синтаксис Automation:

Hooking = Object.Hooking Получить свойство (*)
Object.Hooking = Hooking Установить свойство (*)
Hooking = Object.GetHooking() Получить свойство (**)
Object.SetHooking(Hooking) Установить свойство (**)

Синтаксис COM:

Object.get_Hooking(&Hooking) Получить свойство
Object.put_Hooking(Hooking) Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать диапазон привязки к узлам страниц.
2. Если задаваемое значение находится вне диапазона (1: 50), то его значение корректируется к ближайшему (минимальному или максимальному) значению.

NumberOfCopies – Количество копий

Интерфейс...

Тип данных: long

Синтаксис Automation:

NumberOfCopies Object.NumberOfCopies	=	Получить свойство (*)
Object.NumberOfCopies NumberOfCopies	=	Установить свойство (*)
NumberOfCopies Object.GetNumberOfCopies()	=	Получить свойство (**)
Object.SetNumberOfCopies(NumberOfCopies)		Установить свойство (**)

Синтаксис COM:

Object.get_NumberOfCopies(&NumberOfCopies)	Получить свойство
Object.put_NumberOfCopies(NumberOfCopies)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать количество копий.

OnlyThinLines – Вывод тонкими линиями

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

OnlyThinLines Object.OnlyThinLines	=	Получить свойство (*)
Object.OnlyThinLines OnlyThinLines	=	Установить свойство (*)
OnlyThinLines Object.GetOnlyThinLines()	=	Получить свойство (**)
Object.SetOnlyThinLines(OnlyThinLines)		Установить свойство (**)

Синтаксис COM:

Object.get_OnlyThinLines(&OnlyThinLines)	Получить свойство
Object.put_OnlyThinLines(OnlyThinLines)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак вывода тонкими линиями.

PageOutputOrder – Порядок вывода страниц на печать

Интерфейс...

Тип данных: long

Синтаксис Automation:

PageOutputOrder = Object.PageOutputOrder		Получить свойство (*)
Object.PageOutputOrder = PageOutputOrder		Установить свойство (*)
PageOutputOrder = Object.GetPageOutputOrder()	=	Получить свойство (**)
Object.SetPageOutputOrder(PageOutputOrder)		Установить свойство (**)

Синтаксис COM:

Object.get_PageOutputOrder(&PageOutputOrder)	Получить свойство
Object.put_PageOutputOrder(PageOutputOrder)	Установить свойство

Примечание:

1. Свойство позволяет устанавливать и получать порядок вывода страниц на печать.
2. Если задаваемое значение вне диапазона (1:8), то его значение корректируется к ближайшему (минимальному или максимальному) значению.

PlotToFile – Вывести в файл

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PlotToFile = Object.PlotToFile		Получить свойство (*)
Object.PlotToFile = PlotToFile		Установить свойство (*)
PlotToFile = Object.GetPlotToFile()	=	Получить свойство (**)
Object.SetPlotToFile(PlotToFile)		Установить свойство (**)

Синтаксис COM:

Object.get_PlotToFile(&PlotToFile)	Получить свойство
Object.put_PlotToFile(PlotToFile)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак вывода в файл.

UseCatchSpacing – Привязка к углам листов

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseCatchSpacing Object.UseCatchSpacing	=	Получить свойство (*)
Object.UseCatchSpacing UseCatchSpacing	=	Установить свойство (*)
UseCatchSpacing Object.GetUseCatchSpacing()	=	Получить свойство (**)
Object.SetUseCatchSpacing(UseCatchSpacing)		Установить свойство (**)

Синтаксис COM:

Object.get_UseCatchSpacing(&UseCatchSpacing)	Получить свойство
Object.put_UseCatchSpacing(UseCatchSpacing)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак привязки к углам листов.

UseHooking – Привязка к узлам страниц

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

UseHooking Object.UseHooking	=	Получить свойство (*)
Object.UseHooking UseHooking	=	Установить свойство (*)
UseHooking Object.GetUseHooking()	=	Получить свойство (**)
Object.SetUseHooking(UseHooking)		Установить свойство (**)

Синтаксис COM:

Object.get_UseHooking(&UseHooking)	Получить свойство
Object.put_UseHooking(UseHooking)	Установить свойство

Примечание:

Свойство позволяет устанавливать и получать признак привязки к узлам страниц.

Интерфейс IPrintJob_PrinterSettings

Справка системы КОМПАС...

KOMPAS.chm::/638_76_10_Zadanie_na_pechatq.htm

Задание на печать: Интерфейс настроек принтера

Иерархия:

IDispatch

IKompasAPIObject

IPrintJob_PrinterSettings

Описание:

Позволяет настроить параметры принтера или плоттера.

Примечание:

Интерфейс можно получить у интерфейса IPrintJob с помощью метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

IPrintJob_PrinterSettings – свойства

DeviceName – Имя устройства вывода

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

DeviceName = Object.DeviceName	Получить свойство (*)
DeviceName = Object.GetDeviceName()	Получить свойство (**)

Синтаксис COM:

Object.get_DeviceName(&DeviceName)	Получить свойство

Свойство позволяет получить имя текущего принтера.

Примечание:

1. Свойство доступно только для чтения.
2. Установить другой принтер можно с помощью метода InitPrinterSettings

IsPortraitPage – Установить ориентацию бумаги

Интерфейс...

Синтаксис Automation:

IsPortraitPage = Object.IsPortraitPage	Получить свойство (*)
Object.IsPortraitPage = IsPortraitPage	Установить свойство (*)
IsPortraitPage = Object.GetIsPortraitPage()	Получить свойство (**)
Object.SetIsPortraitPage(IsPortraitPage)	Установить свойство (**)

Синтаксис COM:

Object.get_IsPortraitPage(&IsPortraitPage)	Получить свойство
Object.put_IsPortraitPage(IsPortraitPage)	Установить свойство

Значения свойства:

TRUE	- книжная (вертикальная),
FALSE	- альбомная (горизонтальная).

PaperLength – Длина бумаги

Интерфейс...

Тип данных: long

Синтаксис Automation:

PaperLength = Object.PaperLength	Получить свойство (*)
Object.PaperLength = PaperLength	Установить свойство (*)
PaperLength = Object.GetPaperLength()	Получить свойство (**)
Object.SetPaperLength(PaperLength)	Установить свойство (**)

Синтаксис COM:

Object.get_PaperLength(&PaperLength)	Получить свойство
Object.put_PaperLength(PaperLength)	Установить свойство

Примечание:

Используется, если не задан размер бумаги.

PaperSize – Размер бумаги

Интерфейс...

Тип данных: long

Синтаксис Automation:

PaperSize = Object.PaperSize	Получить свойство (*)

```
Object.PaperSize = PaperSize        Установить свойство (* )
PaperSize                     =     Получить свойство (**)
Object.GetPaperSize()
Object.SetPaperSize( PaperSize )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PaperSize( &PaperSize )    Получить свойство
Object.put_PaperSize(  PaperSize )    Установить свойство
```

Примечание:

1. Значение свойства задается из списка предопределенных констант (подробнее см. описание переменной dmPaperSize из структуры DEVMODE в MSDN).
2. Используется, если не заданы длина и ширина.

PaperSource – Подача бумаги

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PaperSource = Object.PaperSource        Получить свойство (* )
Object.PaperSource = PaperSource        Установить свойство (* )
PaperSource                       =     Получить свойство (**)
Object.GetPaperSource()
Object.SetPaperSource( PaperSource )    Установить свойство (**)
```

Синтаксис COM:

```
Object.get_PaperSource( &PaperSource )    Получить свойство
Object.put_PaperSource(  PaperSource )    Установить свойство
```

Примечание:

Значение свойства задается из списка предопределенных констант (подробнее см. описание переменной dmDefaultSource из структуры DEVMODE в MSDN).

PaperWidth – Ширина бумаги

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
PaperWidth = Object.PaperWidth          Получить свойство (* )
Object.PaperWidth = PaperWidth          Установить свойство (* )
PaperWidth = Object.GetPaperWidth()     Получить свойство (**)
Object.SetPaperWidth( PaperWidth )      Установить свойство (**)
```

Синтаксис COM:

Object.get_PaperWidth(&PaperWidth)	Получить свойство
Object.put_PaperWidth(PaperWidth)	Установить свойство

Примечание:

Используется, если не задан размер бумаги.

Port– Порт вывода

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Port = Object.Port	Получить свойство (*)
Port = Object.GetPort()	Получить свойство (**)

Синтаксис COM:

Object.get_Port(&Port)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PrinterType – Тип настраиваемого принтера

Интерфейс...

Тип данных: из перечисления ksPrinterTypeEnum

Синтаксис Automation:

PrinterType = Object.PrinterType	Получить свойство (*)
Object.PrinterType = PrinterType	Установить свойство (*)
PrinterType = Object.GetPrinterType()	Получить свойство (**)
Object.SetPrinterType(PrinterType)	Установить свойство (**)

Синтаксис COM:

Object.get_PrinterType(&PrinterType)	Получить свойство
Object.put_PrinterType(PrinterType)	Установить свойство

IPrintJob_PrinterSettings – методы

InitPrinterSettings – Установить настройки принтера

Интерфейс...

Синтаксис Automation:

BOOL InitPrinterSettings(BSTR DeviceName,
BOOL IsPortraitPage,
long PaperSize,
long PaperLength,
long PaperWidth,
long PaperSource);

Синтаксис COM:

HRESULT InitPrinterSettings(BSTR DeviceName,
BOOL IsPortraitPage,
long PaperSize,
long PaperLength,
long PaperWidth,
long PaperSource,
BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

DeviceName	- имя принтера,
IsPortraitPage	- ориентация бумаги,
PaperSize	- размер бумаги,
PaperLength	- длина бумаги,
PaperWidth	- ширина бумаги,
PaperSource	- источник бумаги.

Метод позволяет настроить параметры принтера или плоттера, на котором будут печататься документы.

Примечание:

Размер бумаги можно задать двумя способами:

1. Константой, переданной параметром PaperSize. В этом случае параметры PaperLength и PaperWidth должны равняться -1.
2. Задать произвольный размер с помощью параметров PaperLength и PaperWidth. В этом случае параметр PaperSize должен быть равен 0.

LoadPrinterConfig – Загрузить конфигурацию принтера

Интерфейс...

Синтаксис Automation:

BOOL LoadPrinterConfig(BSTR FileName);

Синтаксис COM:

HRESULT LoadPrinterConfig(BSTR FileName, BOOL * Result);

Входные параметры:

FileName - имя файла конфигурации.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Метод позволяет загрузить настройки печати из файла.

Примечание:

Если в качестве параметра передать пустую строку, то будет загружена конфигурация из настроек в зависимости от типа принтера

SavePrinterConfig – Сохранить конфигурацию принтера

Интерфейс...

Сохранить конфигурацию принтера

Синтаксис Automation:

BOOL SavePrinterConfig(BSTR FileName);

Синтаксис COM:

HRESULT SavePrinterConfig(BSTR FileName, BOOL * Result);

Входные параметры:

FileName - имя файла конфигурации.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Метод позволяет сохранить настройки печати в файл.

2D математика

Интерфейс IMath2D

Интерфейс 2D математики.

Иерархия:

IDispatch

IKompasAPIObject

IMath2D

Описание:

Интерфейс позволяет создавать математические кривые и осуществлять работу с точками.

Примечание:

Интерфейс можно получить у интерфейса приложения КОМПАС с помощью свойства IApplication::Math2D.

IMath2D – методы

Arc – Временная математическая дуга

Интерфейс...

Синтаксис Automation:

LPDISPATCH Arc(double Xc,
double Yc,
double Radius,
double Angle1,
double Angle2,);

Синтаксис COM:

HRESULT Arc(double X,
double Y,
double Angle,
ICurve2D ** Result);

Входные параметры:

X, Y	- координаты точки, через которую проходит прямая,
Angle	- угол между прямой и осью OX.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую дугу.

Bezier – Временная математическая кривая Безье

Интерфейс...

Синтаксис Automation:

LPDISPATCH Bezier(BOOL Closed,
BOOL AllPoints,
VARIANT Points);

Синтаксис COM:

HRESULT Bezier(BOOL Closed,
BOOL AllPoints,
VARIANT Points,
ICurve2D ** Result);

Входные параметры:

Closed	- признак замкнутости кривой,
AllPoints	- TRUE - в массиве присутствуют все точки узла, FALSE - в массиве присутствуют только опорные точки,
Points	- массив SAFEARRAY I VT_R8 координат точек кривой.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую кривую Безье.

Circle – Временная математическая окружность

Интерфейс...

Синтаксис Automation:

LPDISPATCH Circle(double Xc,
double Yc,
double Radius);

Синтаксис COM:

HRESULT Circle(double Xc,
double Yc,

double Radius,
ICurve2D ** Result);

Входные параметры:

Xc, Yc	- координаты центра окружности,
Radius	- радиус окружности.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую окружность.

Ellipse – Временный математический эллипс

Интерфейс...

Синтаксис Automation:

LPDISPATCH Ellipse(double Xc,
double Yc,
double A,
double B,
double Angle);

Синтаксис COM:

HRESULT Ellipse(double Xc,
double Yc,
double A,
double B,
double Angle,
ICurve2D ** Result);

Входные параметры:

Xc, Yc	- координаты центра эллипса,
A, B	- длина полуосей эллипса,
Angle	- угол наклона эллипса к оси OX.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временный математический эллипс.

EllipseArc – Временная математическая дуга эллипса

Интерфейс...

Синтаксис Automation:

```
LPDISPATCH EllipseArc( double Xc,
double Yc,
double A,
double B,
double Angle,
double Angle1,
double Angle2 );
```

Синтаксис COM:

```
HRESULT EllipseArc( double Xc,
double   Yc,
double   A,
double   B,
double   Angle,
double   Angle1,
double   Angle2,
ICurve2D ** Result );
```

Входные параметры:

Xc, Yc	- координаты центра дуги эллипса,
A, B	- длина полуосей дуги эллипса,
Angle	- угол наклона дуги эллипса к оси OX.
Angle 1	- начальный угол дуги к оси A,
Angle 2	- конечный угол дуги к оси A.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую дугу эллипса.

Line – Временная математическая прямая

Интерфейс...

Синтаксис Automation:

```
LPDISPATCH Line( double X,
double Y,
double Angle );
```

Синтаксис COM:

```
HRESULT Line( double X,
double Y,
double Angle,
ICurve2D ** Result );
```

Входные параметры:

X, Y	- координаты точки, через которую проходит прямая,
Angle	- угол наклона прямой к оси OX.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую прямую.

LineSeg – Временный математический отрезок

Интерфейс...

Синтаксис Automation:

```
LPDISPATCH LineSeg( double X1,
double Y1,
double X2,
double Y2 );
```

Синтаксис COM:

```
HRESULT LineSeg( double X1,
double Y1,
double X2,
double Y2,
ICurve2D ** Result );
```

Входные параметры:

X1, Y1	- координаты первой точки отрезка,
X2, Y2	- координаты второй точки отрезка.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временный математический отрезок.

MovePoint – Сместить точку

Интерфейс...

Синтаксис Automation:

BOOL MovePoint(double * X,
double * Y,
Angle,
double Len);

Синтаксис COM:

HRESULT MovePoint(double * X,
double * Y,
double Angle,
double Len,
BOOL * Result);

Входные параметры:

X, Y	- координаты точки,
Angle	- угол смещения,
Len	- расстояние смещения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Nurbs – Временная математическая Nurbs-кривая

Интерфейс...

Синтаксис Automation:

LPDISPATCH Nurbs(BOOL Closed,
long Degree,
VARIANT Points,
VARIANT Weights,
VARIANT Knots);

Синтаксис COM:

HRESULT Nurbs(BOOL Closed,
long Degree,
VARIANT Points,
VARIANT Weights,
VARIANT Knots,
ICurve2D ** Result);

Входные параметры:

Closed	- признак замкнутости кривой,
Degree	- порядок кривой,
Points	- массив SAFEARRAY I VT_R8 координат точек кривой,
Weight	- массив SAFEARRAY I VT_R8 весов точек,
Knots	- массив SAFEARRAY I VT_R8 узлов сплайна.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую Nurbs-кривую.

PolyLine – Временная математическая полилиния

Интерфейс...

Синтаксис Automation:

```
LPDISPATCH PolyLine( VARIANT Points,
BOOL  Closed );
```

Синтаксис COM:

```
HRESULT PolyLine( VARIANT Points,
BOOL Closed,
ICurve2D ** Result );
```

Входные параметры:

Points	- массив SAFEARRAY I VT_R8 координат точек полилинии,
Closed	- признак замкнутости кривой.

Возвращаемое значение:

- указатель на интерфейс математической кривой ICurve2D.

Примечание:

Метод позволяет создать временную математическую полилинию.

Rotate – Повернуть точку на угол

Интерфейс...

Синтаксис Automation:

```
BOOL Rotate( double * X,
double * Y,
double  Xc,
double  Yc,
double  Angle );
```

Синтаксис COM:

```
HRESULT Rotate( double  * X,
double  * Y,
double   Xc,
double   Yc,
double   Angle,
BOOL  *  Result );
```

Входные параметры:

X, Y	- координаты точки, которую нужно повернуть,
Xc, Yc	- координаты центра поворота,
Angle	- угол поворота.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет повернуть точку на угол.

Symmetry – Симметрия точки относительно оси

Интерфейс...

Синтаксис Automation:

```
BOOL Symmetry( double  * X,
double  * Y,
LPDISPATCH Curve );
```

Синтаксис COM:

```
HRESULT Symmetry( double  * X,
double  * Y,
ICurve2D * Curve,
BOOL   * Result );
```

Входные параметры:

X, Y	- координаты точки,
Curve	- ось (математическая прямая или отрезок).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить координаты точки, симметричной относительно заданной оси.

Сервисные функции

Измерения

Интерфейс IAreaMeasurements3D

Интерфейс коллекции измерений площади.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IAreaMeasurements3D

КОМПАС версия v18

IAreaMeasurements3D – свойства

AreaMeasurement3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IAreaMeasurement3D.

Синтаксис Automation:

AreaMeasurement3D Object.AreaMeasurement3D(Index)	=	Получить свойство (*)
AreaMeasurement3D Object.GetAreaMeasurement3D(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_AreaMeasurement3D (Index, &AreaMeasurement3D)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IAreaMeasurements3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IAreaMeasurement3D * * Result);

Возвращаемое значение:

- указатель на интерфейс IAreaMeasurement3D

Примечания:

1. Метод позволяет создать новый интерфейс измерения площади.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IAreaMeasurement3D

Интерфейс измерения площади.

Иерархия:

IDispatch
IKompasAPIObject
IModelObject
IAreaMeasurement3D

Данный интерфейс можно получить с помощью метода коллекции измерений площади IAreaMeasurements3D::Add или свойства IAreaMeasurements3D::AreaMeasurement3D.

IAreaMeasurement3D – свойства

Areas – Массив SAFEARRAY площадей граней

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Areas = Object.Areas	Получить свойство (*)
Areas = Object.GetAreas()	Получить свойство (**)

Синтаксис COM:

Object.get_Areas(&Areas)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Faces – Массив SAFEARRAY граней

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Faces = Object.Faces	Получить свойство (*)
Object.Faces = Faces	Установить свойство (*)
Faces = Object.GetFaces()	Получить свойство (**)
Object.SetFaces (Faces)	Установить свойство (**)

Синтаксис COM:

Object.get_Faces(&Faces)	Получить свойство
Object.put_Faces(Faces)	Установить свойство

Sum – Сумма площадей граней

Интерфейс...

Тип данных: double

Синтаксис Automation:

Sum = Object.Sum	Получить свойство (*)
Sum = Object.GetSum()	Получить свойство (**)

Синтаксис COM:

Object.get_Sum(&Sum)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Интерфейс IDistanceAngleMeasurements3D

Интерфейс коллекции измерений расстояния и угла.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IDistanceAngleMeasurements3D

Данный интерфейс можно получить, используя свойство контейнера объектов измерений и диагностики IMeasurementContainer::DistanceAngleMeasurements3D.

IDistanceAngleMeasurements3D – свойства

DistanceAngleMeasurement3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IDistanceAngleMeasurement3D.

Синтаксис Automation:

DistanceAngleMeasurement3D = Object.DistanceAngleMeasurement3D(Index)	Получить свойство (*)
DistanceAngleMeasurement3D = Object.GetDistanceAngleMeasurement3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_DistanceAngleMeasurement3D(Index, &DistanceAngleMeasurement3D)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IDistanceAngleMeasurements3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IDistanceAngleMeasurement3D * * Result);

Возвращаемое значение:

- указатель на интерфейс IDistanceAngleMeasurement3D.

Примечания:

1. Метод позволяет создать новый интерфейс измерения расстояния и угла.
2. После получения нового интерфейса нужно задать параметры операции и вызвать метод IModelObject::Update.

Интерфейс IDistanceAngleMeasurement3D

Интерфейс измерений расстояния и угла.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IDistanceAngleMeasurement3D

Данный интерфейс можно получить с помощью метода коллекции измерений расстояния и угла IDistanceAngleMeasurements3D::Add или свойства IDistanceAngleMeasurements3D::DistanceAngleMeasurement3D.

КОМПАС версия v18

IDistanceAngleMeasurement3D – свойства

Angle – Угол между объектами

Интерфейс...

Тип данных: double

Синтаксис Automation:

Angle = Object.Angle	Получить свойство (*)
Angle = Object.GetAngle()	Получить свойство (**)

Синтаксис COM:

Object.get_Angle(&Angle)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Briefly – Кратко

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Briefly = Object.Briefly	Получить свойство (*)
Object.Briefly = Briefly	Установить свойство (*)
Briefly = Object.GetBriefly()	Получить свойство (**)
Object.SetBriefly (Briefly)	Установить свойство (**)

Синтаксис COM:

Object.get_Briefly(&Briefly)	Получить свойство
Object.put_Briefly(Briefly)	Установить свойство

IsAngleValid – Применимость расчета угла. TRUE – если для данных объектов угол имеет смысл

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

IsAngleValid Object.IsAngleValid	=	Получить свойство (*)
IsAngleValid Object.GetIsAngleValid()	=	Получить свойство (**)

Синтаксис COM:

Object.get_IsAngleValid(&IsAngleValid)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Lmax – Максимальное расстояние

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lmax = Object.Lmax              Получить свойство (* )
Lmax = Object.GetLmax()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Lmax( &Lmax )        Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Lmin – Минимальное расстояние

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Lmin = Object.Lmin              Получить свойство (* )
Lmin = Object.GetLmin()         Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Lmin( &Lmin )        Получить свойство
```

Примечание:

Свойство доступно только для чтения.

LNormal – Расстояние по нормали

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
LNormal = Object.LNormalt       Получитьо свойство (* )
LNormal = Object.GetLNormalt()  Получить свойство (**)
```

Синтаксис COM:

```
Object.get_LNormal( &LNormal )  Получить свойство
```

Примечание:

Свойство доступно только для чтения.

MeasureResult – Результат измерения

Интерфейс...

Тип данных: Значение из перечисления ksMeasureResultEnum.

Синтаксис Automation:

```
MeasureResult = Получить свойство (* )
Object.MeasureResult
MeasureResult = Получить свойство (**)
Object.GetMeasureResult()
```

Синтаксис COM:

```
Object.get_MeasureResult( Получить свойство
&MeasureResult )
```

Примечание:

Свойство доступно только для чтения.

Object1 – Первый объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

```
Object1 = Object.Object1 Получить свойство (* )
Object.Object1 = Object1 Установить свойство (* )
Object1 = Object.GetObject1() Получить свойство (**)
Object.SetObject1 (Object1) Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Object1( &Object1 ) Получить свойство
Object.put_Object1( Object1 ) Установить свойство
```

Object2 – Второй объект

Интерфейс...

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

Object2 = Object.Object2	Получить свойство (*)
Object.Object2 = Object2	Установить свойство (*)
Object2 = Object.GetObject2()	Получить свойство (**)
Object.SetObject2 (Object2)	Установить свойство (**)

Синтаксис COM:

Object.get_Object2(&Object2)	Получить свойство
Object.put_Object2(Object2)	Установить свойство

IDistanceAngleMeasurement3D – методы

GetMaxPoint1 – Получить первую точку отрезка максимального расстояния

Интерфейс...

Синтаксис Automation:

BOOL GetMaxPoint1(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetMaxPoint1(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

GetMaxPoint2 – Получить вторую точку отрезка максимального расстояния

Интерфейс...

Синтаксис Automation:

BOOL GetMaxPoint2(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetMaxPoint2(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

GetMinPoint1 – Получить первую точку отрезка минимального расстояния

Интерфейс...

Синтаксис Automation:

BOOL GetMinPoint1(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetMinPoint1(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

GetMinPoint2 – Получить вторую точку отрезка минимального расстояния

Интерфейс...

Синтаксис Automation:

BOOL GetMinPoint2(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetMinPoint2(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

GetNormalPoint1 – Получить первую точку отрезка расстояния по нормали

Интерфейс...

Синтаксис Automation:

BOOL GetNormalPoint1(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetNormalPoint1(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

GetNormalPoint2 – Получить вторую точку отрезка расстояния по нормали

Интерфейс...

Синтаксис Automation:

BOOL GetNormalPoint2(double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetNormalPoint2(double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X, Y, Z	- координаты точки.

Интерфейс IEdgeLengthMeasurements3D

Интерфейс коллекции измерений длины ребра.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

IModelObjects

IEdgeLengthMeasurements3D

Данный интерфейс можно получить, используя свойство контейнера объектов измерений и диагностики IMeasurementContainer::EdgeLengthMeasurements3D.

КОМПАС версия v18

IEdgeLengthMeasurements3D – свойства

EdgeLengthMeasurement3D – Возвращает элемент, заданный по индексу или по имени

Интерфейс...

Тип данных: Указатель на интерфейс IEdgeLengthMeasurement3D.

Синтаксис Automation:

EdgeLengthMeasurement3D = Object.EdgeLengthMeasurement3D(Index)	Получить свойство (*)
EdgeLengthMeasurement3D = Object.GetEdgeLengthMeasurement3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_EdgeLengthMeasurement3D(Index, &EdgeLengthMeasurement3D)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

IEdgeLengthMeasurements3D – методы

Add – Создает новый элемент и добавляет его в коллекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(IEdgeLengthMeasurement3D * * Result);

Возвращаемое значение:

	- указатель на интерфейс IEdgeLengthMeasurement3D.

Примечание:

1. Метод позволяет создать новый интерфейс объекта измерение длины ребра.
2. После получения нового интерфейса нужно задать параметры измерения и вызвать метод IModelObject::Update.

Интерфейс IEdgeLengthMeasurement3D

Интерфейс измерений длины ребра.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IEdgeLengthMeasurement3D

Данный интерфейс можно получить с помощью метода коллекции измерений длины ребра IEdgeLengthMeasurements3D::Add или свойства IEdgeLengthMeasurements3D::EdgeLengthMeasurement3D.

КОМПАС версия v18

IEdgeLengthMeasurement3D – свойства

Edges – Массив SAFEARRAY ребер

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Edges = Object.Edges	Получить свойство (*)
Object.Edges = Edges	Установить свойство (*)
Edges = Object.GetEdges()	Получить свойство (**)
Object.SetEdges (Edges)	Установить свойство (**)

Синтаксис COM:

Object.get_Edges(&Edges)	Получить свойство
Object.put_Edges(Edges)	Установить свойство

Lengths – Массив SAFEARRAY длин ребер

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

```
Lengths = Object.Lengths          Получить свойство (* )
Lengths = Object.GetLengths()     Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Lengths( &Lengths )          Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Sum – Сумма длин ребер

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
Sum = Object.Sum                  Получить свойство (* )
Sum = Object.GetSum()             Получить свойство (**)
```

Синтаксис COM:

```
Object.get_Sum( &Sum )                  Получить свойство
```

Примечание:

Свойство доступно только для чтения.

Интерфейс IMeasurementContainer

Интерфейс контейнера объектов измерений и диагностики.

Иерархия:

IDispatch

IMeasurementContainer

Интерфейс является дополнительным для компонента IPart7. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

КОМПАС версия v18

IMeasurementContainer – свойства

AreaMeasurements3D – Коллекция измерений площади

Интерфейс...

Тип данных: Указатель на интерфейс IAreaMeasurements3D.

Синтаксис Automation:

AreaMeasurements3D = iObject.AreaMeasurements3D	=	Получить свойство (*)
AreaMeasurements3D = iObject.GetAreaMeasurements3D()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_AreaMeasurements3D (&AreaMeasurements3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DistanceAngleMeasurements3D – Коллекция измерений расстояния и угла

Интерфейс...

Тип данных: Указатель на интерфейс IDistanceAngleMeasurements3D.

Синтаксис Automation:

DistanceAngleMeasurements3D = iObject.DistanceAngleMeasurements3D	=	Получить свойство (*)
DistanceAngleMeasurements3D = iObject.GetDistanceAngleMeasurements3D()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_DistanceAngleMeasurements3D (&DistanceAngleMeasurements3D)	Получить свойство

Примечание:

Свойство доступно только для чтения.

EdgeLengthMeasurements3D – Коллекция измерений длины ребра

Интерфейс...

Тип данных: Указатель на интерфейс IEdgeLengthMeasurements3D.

Синтаксис Automation:

EdgeLengthMeasurements3D = iObject.EdgeLengthMeasurements3D	=	Получить свойство (*)
EdgeLengthMeasurements3D = iObject.GetEdgeLengthMeasurements3D()	=	Получить свойство (**)

Синтаксис COM:

iObject->get_EdgeLengthMeasurements3D (&EdgeLengthMeasurements3D)	(	Получить свойство

Примечание:

Свойство доступно только для чтения.

Слои

Интерфейс IDocument3DManager

Менеджер 3D документа.

Иерархия:

IDispatch

IKompasAPIObject

IDocument3DManager

Интерфейс можно получить с помощью свойства IKompasDocument3D1::Document3DManager.

IDocument3DManager – свойства

Layers3D – Коллекция слоев в 3D

Интерфейс...

Тип данных: Указатель на интерфейс ILayers3D

Синтаксис Automation:

Layers3D = Object.Layers3D		Получить свойство (*)

Layers3D = Object.GetLayers3D)		Получить свойство (**)

Синтаксис COM:

Object.get_Layers3D(&Layers3D)	Получить свойство

Примечание:

Свойство только для чтения.

LayersDynamicGroups3D – Коллекция динамических групп слоев в 3D

Интерфейс...

Тип данных: Указатель на интерфейс ILayerGroups3D

Синтаксис Automation:

LayersDynamicGroups3D = Object.LayersDynamicGroups3D		Получить свойство (*)
LayersDynamicGroups3D = Object.GetLayersDynamicGroups 3D		Получить свойство (**)

Синтаксис COM:

Object.get_LayersDynamicGroup s3D(&LayersDynamicGroups3D)	Получить свойство

Примечание:

Свойство только для чтения.

LayersGroups3D – Коллекция групп слоев в 3D

Интерфейс...

Тип данных: Указатель на интерфейс ILayerGroups3D

Синтаксис Automation:

LayersGroups3D Object.LayersGroups3D	=	Получить свойство (*)
LayersGroups3D Object.GetLayersGroups3D	=	Получить свойство (**)

Синтаксис COM:

Object.get_LayersGroups3D(&LayersGroups3D)	Получить свойство

Примечание:

Свойство только для чтения.

IDocument3DManager – методы

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс ILayers3D

Интерфейс коллекции слоев в 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

ILayers3D

Интерфейс коллекции слоев в 3D можно получить у Менеджера 3D документа с помощью свойства IDocument3DManager::Layers3D или у Группы слоев в 3D с помощью свойства ILayerGroup3D::Layers.

ILayers3D – свойства

Layer3D – Возвращает слой, заданный по индексу, имени или ссылке

Интерфейс...

Тип данных: Указатель на интерфейс ILayer3D

Синтаксис Automation:

Layer3D = Object.Layer3D(Index)	Получить свойство (*)
Layer3D = Object.GetLayer3D(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Layer3D(Index, &Layer3D)	Получить свойство

Примечание:

1. Свойство только для чтения.
2. В качестве индекса может использоваться индекс элемента в массиве, ссылка на слой, имя слоя.

Layer3DByNumber – Возвращает слой, заданный по номеру

Интерфейс...

Тип данных: Указатель на интерфейс ILayer3D

Синтаксис Automation:

Layer3DByNumber Object.Layer3DByNumber(Number)	=	Получить свойство (*)
Layer3DByNumber Object.GetLayer3DByNumber(Number)	=	Получить свойство (**)

Синтаксис COM:

Object.get_Layer3DByNumber(Number, &Layer3DByNumber)	Получить свойство

Входные параметры:

number	- номер слоя; тип long.

Примечание:

Свойство только для чтения.

ILayers3D – методы

Интерфейс...

Add – Создать слой (добавляет слой в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILayer3D ** Result);

Возвращаемое значение:

- указатель на интерфейс слоя ILayer3D.

Attach – Добавить существующий слой

Интерфейс...

Синтаксис Automation:

BOOL Attach(ILayer3D * PVal);

Синтаксис COM:

HRESULT Attach(ILayer3D * PVal, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Detach – Отсоединить слой

Интерфейс...

Синтаксис Automation:

BOOL Detach(ILayer3D * PVal);

Синтаксис COM:

HRESULT Detach(ILayer3D * PVal, BOOL * Result);

Возвращаемое значение:

TRUE	в случае успешного завершения,
FALSE	- в случае неудачи.

Входные параметры:

PVal - указатель на интерфейс слоя ILayer3D.

Возвращаемое значение:

- Полный путь к библиотеке на диске.

Примечание:

1. Интерфейс слоя отсоединяется от коллекции, слой при этом из документа не удаляется.
2. Метод работает только для коллекции слоев группы ILayerGroup3D, если она не является динамической.

Интерфейс ILayerGroups3D

Интерфейс коллекции групп слоев в 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

ILayerGroups3D

Интерфейс можно получить у Менеджера 3D документа с помощью свойств IDocument3DManager::LayersGroups3D и IDocument3DManager::LayersDynamicGroups3D и у группы слоев с помощью свойства ILayerGroup3D::LayerGroups.

ILayerGroups3D – свойства

LayerGroup3D – Возвращает группу по индексу, уникальному идентификатору или имени

Интерфейс...

Тип данных: Указатель на интерфейс ILayerGroup3D

Синтаксис Automation:

LayerGroup3D = Получить свойство (*)
Object.LayerGroup3D(Index)
LayerGroup3D = Получить свойство (**)
Object.GetLayerGroup3D(Index)

Синтаксис COM:

Object.get_LayerGroup3D(Index, &LayerGroup3D) Получить свойство

Примечание:

Свойство только для чтения.

ILayerGroups3D – методы

Add – Создать группу (добавляет группу в коллекцию)

Интерфейс...

Синтаксис Automation:

LPDISPATCH Add();

Синтаксис COM:

HRESULT Add(ILayerGroup3D ** Result);

Возвращаемое значение:

- указатель на интерфейс группы слоев ILayerGroup3D.

Интерфейс ILayer3D

Интерфейс слоя в 3D.

Иерархия:

IDispatch

IKompasAPIObject

ILayer3D

Интерфейс можно получить у коллекции слоев 3D с помощью свойства ILayers3D::Layer3D и метода ILayers3D::Add.

Имеет дополнительный интерфейс IColorParam7, который можно получить с помощью метода IUnknown::QueryInterface.

Примечание:

Чтобы новые параметры интерфейса IColorParam7 слоя ILayer3D обновились у объектов модели, нужно вызвать метод IDocument3DManager::Update.

ILayer3D – свойства

Color – Цвет слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

Color = Object.Color Получить свойство (*)

```
Object.Color = Color                    Установить свойство (* )
Color = Object.GetColor()               Получить свойство (**)
Object.SetColor (Color)                 Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Color( &Color )              Получить свойство
Object.put_Color(  Color )              Установить свойство
```

Comment – Комментарий

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Comment = Object.Comment                Получить свойство (* )
Object.Comment = Comment                Установить свойство (* )
Comment = Object.GetComment()           Получить свойство (**)
Object.SetComment (Comment)             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Comment( &Comment )          Получить свойство
Object.put_Comment(  Comment )          Установить свойство
```

Current – Состояние слоя – текущий

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Current = Object.Current                Получить свойство (* )
Object.Current = Current                Установить свойство (* )
Current = Object.GetCurrent()           Получить свойство (**)
Object.SetCurrent (Current)             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Current( &Current )          Получить свойство
Object.put_Current(  Current )          Установить свойство
```

Примечание:

Установить свойство можно только равным TRUE. Новый слой всегда создается текущим.

Editable – Признак редактирования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
Editable = Object.Editable                Получить свойство (* )
Object.Editable = Editable                Установить свойство (* )
Editable = Object.GetEditable()           Получить свойство (**)
Object.SetEditable (Editable)             Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Editable( &Editable )          Получить свойство
Object.put_Editable(  Editable )          Установить свойство
```

Name – Имя слоя

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

```
Name = Object.Name                        Получить свойство (* )
Object.Name = Name                        Установить свойство (* )
Name = Object.GetName()                   Получить свойство (**)
Object.SetName (Name)                     Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Name( &Name )                  Получить свойство
Object.put_Name(  Name )                  Установить свойство
```

Number – Номер слоя

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
Number = Object.Number                    Получить свойство (* )
Object.Number = Number                    Установить свойство (* )
Number = Object.GetNumber()               Получить свойство (**)
Object.SetNumber (Number)                 Установить свойство (**)
```

Синтаксис COM:

Object.get_Number(&Number)	Получить свойство
Object.put_Number(Number)	Установить свойство

Projected – Признак проецирования

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Projected = Object.Projected	Получить свойство (*)
Object.Projected = Projected	Установить свойство (*)
Projected = Object.GetProjected()	Получить свойство (**)
Object.SetProjected (Projected)	Установить свойство (**)

Синтаксис COM:

Object.get_Projected(&Projected)	Получить свойство
Object.put_Projected(Projected)	Установить свойство

Visible – Состояние слоя – видимый или погашенный

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

Visible = Object.Visible	Получить свойство (*)
Object.Visible = Visible	Установить свойство (*)
Visible = Object.GetVisible()	Получить свойство (**)
Object.SetVisible (Visible)	Установить свойство (**)

Синтаксис COM:

Object.get_Visible(&Visible)	Получить свойство
Object.put_Visible(Visible)	Установить свойство

ILayer3D – методы

Delete – Удалить слой

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Update – Обновление данных

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM :

HRESULT Update(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс ILayerGroup3D

Интерфейс группы слоев в 3D.

Иерархия:

IDispatch

IKompasAPIObject

IKompasCollection

ILayerGroup3D

Данный интерфейс можно получить у коллекции групп слоев ILayerGroups3D с помощью свойства ILayerGroups3D::LayerGroup3D, метода ILayerGroups3D::Add, либо от интерфейса другой группы слоев с помощью свойства ILayerGroup3D::OwnerGroup.

ILayerGroup3D – свойства

LayerFilterConditions – Коллекция условий фильтрации

Интерфейс...

Тип данных: Указатель на интерфейс ILayerFilterConditions

Синтаксис Automation:

LayerFilterConditions = Object.LayerFilterConditions	=	Получить свойство (*)
LayerFilterConditions = Object.GetLayerFilterConditions( )	=	Получить свойство (**)

Синтаксис COM:

Object.get_LayerFilterConditions (&LayerFilterConditions)	Получить свойство

Примечание:

Свойство только для чтения.

LayerGroups – Коллекция групп слоев

Интерфейс...

Тип данных: Указатель на интерфейс ILayerGroup3D

Синтаксис Automation:

LayerGroups = Object.LayerGroups	=	Получить свойство (*)
LayerGroups = Object.GetLayerGroups()	=	Получить свойство (**)

Синтаксис COM:

Object.get_LayerGroups(&LayerGroups)	Получить свойство

Примечание:

Свойство только для чтения.

Layers – Коллекция слоев

Интерфейс...

Тип данных: Указатель на интерфейс ILayers3D

Синтаксис Automation:

Layers = Object.Layers	Получить свойство (*)
Layers = Object.GetLayers()	Получить свойство (**)

Синтаксис COM:

Object.get_Layers(&Layers)	Получить свойство

Примечание:

Свойство только для чтения.

Name – Имя группы

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

Name = Object.Name	Получить свойство (*)
Object.Name = Name	Установить свойство (*)
Name = Object.GetName()	Получить свойство (**)
Object.SetName (Name)	Установить свойство (**)

Синтаксис COM:

Object.get_Name(&Name)	Получить свойство
Object.put_Name(Name)	Установить свойство

OwnerGroup – Владелец группы – другая группа

Интерфейс...

Тип данных: Указатель на интерфейс ILayerGroup3D

Синтаксис Automation:

OwnerGroup Object.OwnerGroup	=	Получить свойство (*)
OwnerGroup Object.GetOwnerGroup()	=	Получить свойство (**)

Синтаксис COM:

Object.get_OwnerGroup(&OwnerGroup)	Получить свойство

Примечание:

Свойство только для чтения.

UniqueId – Уникальный идентификатор группы слоев

Интерфейс...

Тип данных: double

Синтаксис Automation:

UniqueId = Object.UniqueId	Получить свойство (*)
UniqueId = Object.GetUniqueId()	Получить свойство (**)

Синтаксис COM:

Object.get_UniqueId(&UniqueId)	Получить свойство

Примечание:

1. Свойство только для чтения.
2. После создания группы свойство не изменяется и хранится в чертеже.

ILayerGroup3D – методы

Delete – Удалить группу

Интерфейс...

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

При удалении группы слоев слои, входящие в нее, из документа не удаляются.

Зоны

Интерфейс IZone

Интерфейс параметров зоны

Иерархия:

IDispatch
- IKompasAPIObject
  - IModelObject
    - IZone

Данный интерфейс можно получить с помощью метода IZonesManager::AddZone или свойств IZonesManager::Zone, IZonesManager::CurrentZone.

Версия Компас v18.1

IZone – свойства

ZoneType – Способ создания зоны

Интерфейс

Тип данных: из перечисления ksZoneTypeEnum

Синтаксис Automation:

ZoneType = Object.ZoneType	Получить свойство (*)
Object.ZoneType = ZoneType	Установить свойство (*)
ZoneType = Object.GetZoneType()	Получить свойство (**)
Object.SetZoneType(ZoneType)	Установить свойство (**)

Синтаксис COM:

Object.get_ZoneType(&ZoneType)	Получить свойство
Object.put_ZoneType(ZoneType)	Установить свойство

Parameters – Параметры зоны

Интерфейс

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Parameters = Object.Parameters		Получить свойство (*)
Parameters	=	Получить свойство (**)
Object.GetParameters()		

Синтаксис COM:

Object.get_Parameters(&Parameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

В зависимости от способа создания зоны IZone::ZoneType возвращается интерфейс параметров зоны.

- ksZoneFree - Нет параметров;
- ksZoneByPoints - Параметры зоны заданной габаритным параллелепипедом - IZoneParametersByBorderPoints;

▼ ksZoneByObjects - Параметры зоны по суммарному габариту объектов - IZoneParametersByObjects.

SelectObjects – Вернуть объекты, попадающие в зону или находящиеся вне зоны

Интерфейс

Тип данных: VARIANT

Синтаксис Automation:

SelectObjects = Object.SelectObjects(SelectType)	=	Получить свойство (*)
SelectObjects = Object.GetSelectObjects(SelectType)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SelectObjects(SelectType, &SelectObjects)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

SelectParts – Вернуть вставки моделей с редактируемого уровня, попадающие в зону или находящиеся вне зоны

Интерфейс

Тип данных: VARIANT

Синтаксис Automation:

SelectParts = Object.SelectParts(SelectType)		Получить свойство (*)
SelectParts = Object.GetSelectParts(SelectType)	=	Получить свойство (**)

Синтаксис COM:

Object.get_SelectParts(SelectType, &SelectParts)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

IZone – методы

GetGabarit – Выдать габарит

Интерфейс

Синтаксис Automation:

BOOL GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2);

Синтаксис COM:

HRESULT GetGabarit(double * X1, double * Y1, double * Z1, double * X2, double * Y2, double * Z2, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Выходные параметры:

X1, Y1, Z1, X2, Y2, Z2	- координаты габаритного параллелепипеда.

Интерфейс IZonesManager

Интерфейс Менеджера зон

Иерархия:

IDispatch

IKompasAPIObject

IZonesManager

Данный интерфейс можно получить с помощью свойства IPart7::ZonesManager.

Версия Компас v18.1

IZonesManager – свойства

CreateZonesInGlobalCS – Создавать зоны в абсолютной СК

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

CreateZonesInGlobalCS Object.CreateZonesInGlobalCS	=	Получить свойство (*)
Object.CreateZonesInGlobalCS CreateZonesInGlobalCS	=	Установить свойство (*)
CreateZonesInGlobalCS Object.GetCreateZonesInGlobalCS()	=	Получить свойство (**)
Object.SetCreateZonesInGlobalCS(CreateZonesInGlobalCS)		Установить свойство (**)

Синтаксис COM:

Object.get_CreateZonesInGlobalCS(&CreateZonesInGlobalCS)	Получить свойство
Object.put_CreateZonesInGlobalCS(CreateZonesInGlobalCS)	Установить свойство

CurrentZone – Текущая зона

Интерфейс

Тип данных: Указатель на интерфейс IZone

Синтаксис Automation:

CurrentZone = Object.CurrentZone	Получить свойство (*)
Object.CurrentZone = CurrentZone	Установить свойство (*)
CurrentZone = Object.GetCurrentZone()	Получить свойство (**)
Object.SetCurrentZone(CurrentZone)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentZone(&CurrentZone)	Получить свойство
Object.put_CurrentZone(CurrentZone)	Установить свойство

Zone – Получить зону по имени или индексу

Интерфейс

Тип данных: Указатель на интерфейс IZone

Синтаксис Automation:

Zone = Object.Zone(Index)	Получить свойство (*)
Zone = Object.GetZone(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Zone(Index, &Zone)	Получить свойство

Входные параметры:

VARIANT Index	- индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ZonesCount – Количество зон

Интерфейс

Тип данных: long

Синтаксис Automation:

ZonesCount Object.ZonesCount	=	Получить свойство (*)
ZonesCount Object.GetZonesCount()	=	Получить свойство (**)

Синтаксис COM:

Object.get_ZonesCount(&ZonesCount)	Получить свойство

Примечание:

ZoneDivision – Получить объект Разбиение зоны по имени или индексу

Интерфейс

Тип данных: Указатель на интерфейс IZoneDivision

Синтаксис Automation:

ZoneDivision Object.ZoneDivision(Index)	=	Получить свойство (*)
ZoneDivision Object.GetZoneDivision(Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ZoneDivision(Index, &ZoneDivision) — Получить свойство

Входные параметры:

VARIANT Index - индекс или имя элемента.

Примечание:

Свойство доступно только для чтения.

ZonesDivisionCount – Количество объектов Разбиение зоны

Интерфейс

Тип данных: long

Синтаксис Automation:

ZonesDivisionCount = Object.ZonesDivisionCount — Получить свойство (*)
ZonesDivisionCount = Object.GetZonesDivisionCount() — Получить свойство (**)

Синтаксис COM:

Object.get_ZonesDivisionCount(&ZonesDivisionCount) — Получить свойство

Примечание:

Свойство доступно только для чтения.

ZonesTree – Дерево зон

Интерфейс

Тип данных: Указатель на интерфейс IFeature7

Синтаксис Automation:

ZonesTree = Object.ZonesTree — Получить свойство (*)
ZonesTree = Object.GetZonesTree() — Получить свойство (**)

Синтаксис COM:

Object.get_ZonesTree(&ZonesTree)	Получить свойство

Примечание:

Свойство доступно только для чтения.

ZonesVisible – Отображать зоны

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

ZonesVisible = Object.ZonesVisible	Получить свойство (*)
Object.ZonesVisible = ZonesVisible	Установить свойство (*)
ZonesVisible = Object.GetZonesVisible()	Получить свойство (**)
Object.SetZonesVisible(ZonesVisible)	Установить свойство (**)

Синтаксис COM:

Object.get_ZonesVisible(&ZonesVisible)	Получить свойство
Object.put_ZonesVisible(ZonesVisible)	Установить свойство

IZonesManager – методы

AddZone – Добавить зону

Интерфейс

Синтаксис Automation:

LPDISPATCH AddZone(ksZoneTypeEnum ZoneType);

Синтаксис COM :

HRESULT AddZone(ksZoneTypeEnum ZoneType, IZone ** Result);

Возвращаемое значение:

	Указатель на интерфейс IZone

Входные параметры:

ZoneType	- способ создания зоны из перечисления ksZoneTypeEnum

AddZoneDivision – Добавить Разбиение зоны

Интерфейс

Синтаксис Automation:

LPDISPATCH AddZoneDivision(ksZoneDivisionTypeEnum ZoneType);

Синтаксис COM :

HRESULT AddZoneDivision(ksZoneDivisionTypeEnum ZoneType, IZoneDivision * * Result);

Возвращаемое значение:

	- указатель на интерфейс IZoneDivision

Входные параметры:

ZoneType	- способ разбиения зоны из перечисления ksZoneDivisionTypeEnum

Интерфей IZoneDivision

Интерфейс параметров разбиения зоны

Иерархия:

IDispatch

IKompasAPIObject

IZoneDivision

Данный интерфейс можно получить с помощью метода IZonesManager::AddZoneDivision или свойства IZonesManager::ZoneDivision.

Версия Компас v18.1

IZoneDivision – свойства

Parameters – Параметры зоны

Интерфейс

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Parameters = Object.Parameters		Получить свойство (*)
Parameters Object.GetParameters()	=	Получить свойство (**)

Синтаксис COM:

Object.get_Parameters(&Parameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

В зависимости от способа создания зоны IZoneDivision::ZoneDivisionType возвращается интерфейс параметров зоны.

- ksZoneDivisionRegular - Параметры разбиения зоны для способа Равномерно по осям - IZoneDivisionParametersRegular;
- ksZoneDivisionByPlanes - Параметры разбиения зоны для способа По набору плоскостей - IZoneDivisionParametersByPlanes.

Zone – Зона

Интерфейс

Тип данных: Указатель на интерфейс IZone

Синтаксис Automation:

Zone = Object.Zone	Получить свойство (*)
Object.Zone = Zone	Установить свойство (*)
Zone = Object.GetZone()	Получить свойство (**)
Object.SetZone(Zone)	Установить свойство (**)

Синтаксис COM:

Object.get_Zone(&Zone)	Получить свойство
Object.put_Zone(Zone)	Установить свойство

ZoneDivisionType – Способ разбиения зоны

Интерфейс

Тип данных: из перечисления ksZoneDivisionTypeEnum

Синтаксис Automation:

ZoneDivisionType = Object.ZoneDivisionType	=	Получить свойство (*)
Object.ZoneDivisionType = ZoneDivisionType	=	Установить свойство (*)
ZoneDivisionType = Object.GetZoneDivisionType()	=	Получить свойство (**)
Object.SetZoneDivisionType(ZoneDivisionType)		Установить свойство (**)

Синтаксис COM:

Object.get_ZoneDivisionType(&ZoneDivisionType)	Получить свойство
Object.put_ZoneDivisionType(ZoneDivisionType)	Установить свойство

Интерфейс IZoneParametersByBorderPoints

Интерфейс параметров зоны, заданной габаритным параллелепипедом.

Иерархия:

IDispatch

IKompasAPIObject

IZoneParametersByBorderPoints

Данный интерфейс можно получить с помощью метода IZone::Parameters.

Версия Компас v18.1

IZoneParametersByBorderPoints – свойство

AssociationObject – Установить опорный объект для вершины

Интерфейс

Тип данных: Указатель на интерфейс IModelObject

Синтаксис Automation:

AssociationObject	=	Получить свойство (*)
Object.AssociationObject(First)		
Object.AssociationObject(First)	=	Установить свойство (*)
AssociationObject		
AssociationObject	=	Получить свойство (**)
Object.GetAssociationObject(First)		
Object.SetAssociationObject(First, AssociationObject)		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(&AssociationObject)	First,	Получить свойство
Object.put_AssociationObject(AssociationObject)	First,	Установить свойство

Входные параметры:

BOOL	- First - TRUE - первая вершина, FALSE - вторая вершина.

BuildingType – Способ задания габаритов

Интерфейс

Тип данных: из перечисления ksGabaritBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

PointParameters – Получить интерфейс параметров точки

Интерфейс

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

PointParameters = Object.PointParameters(First)	=	Получить свойство (*)
PointParameters = Object.GetPointParameters(First)	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointParameters(First, &PointParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PointType – Тип параметров построения точки

Интерфейс

Тип данных: из перечисления ksPoint3DTypeEnum

Синтаксис Automation:

PointType = Object.PointType(First)	Получить свойство (*)
Object.PointType(First) = PointType	Установить свойство (*)
PointType = Object.GetPointType(First)	Получить свойство (**)
Object.SetPointType(First, PointType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointType(First, &PointType)	Получить свойство
Object.put_PointType(First, PointType)	Установить свойство

Входные параметры:

BOOL	- First - TRUE - первая вершина, FALSE - вторая вершина.

IZoneParametersByBorderPoints – методы

GetPoint – Получить координаты точки

Интерфейс

Синтаксис Automation:

BOOL GetPoint(BOOL First, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(BOOL First, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

First	- TRUE - первая вершина, - FALSE - вторая вершина.

Выходные параметры:

X, Y, Z	- Координаты вершины.

SetPoint – Установить координаты точки

Интерфейс

Синтаксис Automation :

BOOL SetPoint(BOOL First, double X, double Y, double Z);

Синтаксис COM :

HRESULT SetPoint(BOOL First, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

First	- TRUE - первая вершина, - FALSE - вторая вершина.

Интерфейс IZoneParametersByObjects

Интерфейс параметров зоны по суммарному габариту объектов.

Иерархия:

IDispatch

IKompasAPIObject

IZoneParametersByObjects

Данный интерфейс можно получить с помощью метода IZone::Parameters.

Версия Компас v18.1

IZoneParametersByObjects – свойства

BaseObjects – Объекты

Интерфейс

Тип данных: VARIANT

Синтаксис Automation:

BaseObjects = Object.BaseObjects	Получить свойство (*)
Object.BaseObjects = BaseObjects	Установить свойство (*)
BaseObjects = Object.GetBaseObjects()	Получить свойство (**)
Object.SetBaseObjects(BaseObjects)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseObjects(&BaseObjects)	Получить свойство
Object.put_BaseObjects(BaseObjects)	Установить свойство

Примечание.

Позволяет получать и устанавливать опорный объект или массив опорных объектов по которым определяется суммарный габарит зоны.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATCH).

Интерфейс IZoneDivisionParametersRegular

Интерфейс параметров разбиения зоны для способа Равномерно по осям.

Иерархия:

IDispatch

IKompasAPIObject

IZoneDivisionParametersRegular

Данный интерфейс можно получить с помощью метода IZoneDivision::Parameters.

Версия Компас v18.1

IZoneDivisionParametersRegular – свойства

XCount – Количество по X

Интерфейс

Тип данных: long

Синтаксис Automation:

XCount = Object.XCount	Получить свойство (*)
Object.XCount = XCount	Установить свойство (*)
XCount = Object.GetXCount()	Получить свойство (**)
Object.SetXCount(XCount)	Установить свойство (**)

Синтаксис COM:

Object.get_XCount(&XCount)	Получить свойство
Object.put_XCount(XCount)	Установить свойство

YCount – Количество по Y

Интерфейс

Тип данных: long

Синтаксис Automation:

YCount = Object.YCount	Получить свойство (*)
Object.YCount = YCount	Установить свойство (*)

```
YCount = Object.GetYCount()          Получить свойство (**)
Object.SetYCount( YCount )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_YCount( &YCount )         Получить свойство
Object.put_YCount(  YCount )         Установить свойство
```

ZCount – Количество по Z

Интерфейс

Тип данных: long

Синтаксис Automation:

```
ZCount = Object.ZCount               Получить свойство (* )
Object.ZCount = ZCount               Установить свойство (* )
ZCount = Object.GetZCount()          Получить свойство (**)
Object.SetZCount( ZCount )           Установить свойство (**)
```

Синтаксис COM:

```
Object.get_ZCount( &ZCount )         Получить свойство
Object.put_ZCount(  ZCount )         Установить свойство
```

Интерфейс IZoneDivisionParametersByPlanes

Интерфейс параметров разбиения зоны для способа по набору плоскостей.

Иерархия:

IDispatch

IKompasAPIObject

IZoneDivisionParametersByPlanes

Данный интерфейс можно получить с помощью метода IZoneDivision::Parameters.

Версия Компас v18.1

IZoneDivisionParametersByPlanes – свойства

Planes – Секущие плоскости

Интерфейс

Тип данных: VARIANT

Синтаксис Automation:

```
Planes = Object.Planes               Получить свойство (* )
Object.Planes = Planes               Установить свойство (* )
```

```
Planes = Object.GetPlanes()              Получить свойство (**)
Object.SetPlanes( Planes )               Установить свойство (**)
```

Синтаксис COM:

```
Object.get_Planes( &Planes )             Получить свойство
Object.put_Planes(  Planes )             Установить свойство
```

Примечание.

Позволяет получать и устанавливать секущую плоскость или массив секущих плоскостей для разбиения зоны.

Массив объектов возвращается в виде массива SAFEARRAY объектов LPDISPATCH (VT_ARRAY | VT_DISPATC.

Динамическое сечение

Интерфейс IDynamicCrossSection

Интерфейс параметров динамического сечения.

Иерархия:

IDispatch

IKompasAPIObject

IModelObject

IDynamicCrossSection

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionsManager::AddDynamicCrossSection и свойств IDynamicCrossSectionsManager::DynamicCrossSection, IDynamicCrossSectionsManager::CurrentDynamicCrossSection.

Версия Компас v18.1

IDynamicCrossSection – свойства

FillCutPlanes – Закрашивать сечение

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

```
FillCutPlanes = Object.FillCutPlanes          Получить свойство (* )
Object.FillCutPlanes = FillCutPlanes          Установить свойство (* )
FillCutPlanes = Object.GetFillCutPlanes()     Получить свойство (**)
Object.SetFillCutPlanes( FillCutPlanes )      Установить свойство (**)
```

Синтаксис COM:

Object.get_FillCutPlanes(&FillCutPlanes)	Получить свойство
Object.put_FillCutPlanes(FillCutPlanes)	Установить свойство

Step – Получить шаг сечения по индексу

Интерфейс

Тип данных: Указатель на интерфейс IDynamicCrossSectionStep.

Синтаксис Automation:

Step = Object.Step(Index)	Получить свойство (*)
Step = Object.GetStep(Index)	Получить свойство (**)

Синтаксис COM:

Object.get_Step(Index, &Step)	Получить свойство

Входные параметры:

long Index	- индекс шага.

Примечание:

Свойство доступно только для чтения.

StepsCount – Количество шагов сечения

Интерфейс

Тип данных: long

Синтаксис Automation:

StepsCount = Object.StepsCount	Получить свойство (*)
StepsCount = Object.GetStepsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_StepsCount(&StepsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IDynamicCrossSection – методы

AddStep – Добавить шаг

Интерфейс

Синтаксис Automation:

LPDISPATCH AddStep(ksDynamicCrossSectionStepBuildingTypeEnum BuildingType);

Синтаксис COM:

HRESULT AddStep(ksDynamicCrossSectionStepBuildingTypeEnum BuildingType, IDynamicCrossSectionStep * * Result);

Возвращаемое значение:

Указатель на интерфейс IDynamicCrossSectionStep.

Входные параметры:

BuildingType	- Способ создания шага сечения модели из перечисления ksDynamicCrossSectionStepBuildingTypeEnum.

Delete – Удаление сечения

Интерфейс

Синтаксис Automation:

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс IDynamicCrossSectionStep

Интерфейс Шаг динамического сечения.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStep

Данный интерфейс можно получить с помощью метода IDynamicCrossSection::AddStep и свойства IDynamicCrossSection::Step.

IDynamicCrossSectionStep – свойства

BuildingType – Тип шага динамического сечения

Интерфейс

Тип данных: из перечисления ksDynamicCrossSectionStepBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

Parameters – Параметры шага

Интерфейс

Тип данных: Указатель на интерфейс IKompasAPIObject

Синтаксис Automation:

Parameters = Object.Parameters	Получить свойство (*)
Parameters = Object.GetParameters()	Получить свойство (**)

Синтаксис COM:

Object.get_Parameters(&Parameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IDynamicCrossSectionStep – методы

Delete – Удаление шага

Интерфейс

Синтаксис Automation :

BOOL Delete();

Синтаксис COM:

HRESULT Delete(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс IDynamicCrossSectionsManager

Интерфейс Менеджера динамических сечений.

Иерархия:

IDispatch

IDynamicCrossSectionsManager

Интерфейс является дополнительным к интерфейсу документа IKompasDocument3D. Данный интерфейс можно получить с помощью метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

Версия Компас v18.1

IDynamicCrossSectionsManager – свойства

CurrentDynamicCrossSection – Текущее динамическое сечение

Интерфейс

Тип данных: Указатель на интерфейс IDynamicCrossSection.

Синтаксис Automation:

CurrentDynamicCrossSection Object.CurrentDynamicCrossSection	=	Получить свойство (*)
Object.CurrentDynamicCrossSection CurrentDynamicCrossSection	=	Установить свойство (*)
CurrentDynamicCrossSection Object.GetCurrentDynamicCrossSection()	=	Получить свойство (**)

Object.SetCurrentDynamicCrossSection(CurrentDynamicCrossSection)	Установить свойство (**)

Синтаксис COM:

Object.get_CurrentDynamicCrossSection (&CurrentDynamicCrossSection)	Получить свойство
Object.put_CurrentDynamicCrossSection (CurrentDynamicCrossSection)	Установить свойство

DynamicCrossSection – Получить объект по имени, индексу или указателю на размер

Интерфейс

Тип данных: Указатель на интерфейс IDynamicCrossSection.

Синтаксис Automation:

DynamicCrossSection Object.DynamicCrossSection(Index)	=	Получить свойство (*)
DynamicCrossSection Object.GetDynamicCrossSection (Index)	=	Получить свойство (**)

Синтаксис COM:

Object.get_DynamicCrossSection(Index, &DynamicCrossSection)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DynamicCrossSectionsCount – Количество динамических сечений

Интерфейс

Тип данных: long

Синтаксис Automation:

DynamicCrossSectionsCount = Object.DynamicCrossSectionsCount	Получить свойство (*)
DynamicCrossSectionsCount = Object.GetDynamicCrossSectionsCount()	Получить свойство (**)

Синтаксис COM:

Object.get_DynamicCrossSectionsCount(&DynamicCrossSectionsCount)	Получить свойство

Примечание:

Свойство доступно только для чтения.

DynamicCrossSectionModeOn – Включить/отключить режим отображения сечения модели

Интерфейс

Тип данных: BOOL

Синтаксис Automation:

DynamicCrossSectionModeOn = Object.DynamicCrossSectionModeOn	Получить свойство (*)
Object.DynamicCrossSectionModeOn = DynamicCrossSectionModeOn	Установить свойство (*)
DynamicCrossSectionModeOn = Object.GetDynamicCrossSectionModeOn()	Получить свойство (**)
Object.SetDynamicCrossSectionModeOn (DynamicCrossSectionModeOn)	Установить свойство (**)

Синтаксис COM:

Object.get_DynamicCrossSectionModeOn(&DynamicCrossSectionModeOn)	Получить свойство
Object.put_DynamicCrossSectionModeOn(DynamicCrossSectionModeOn)	Установить свойство

IDynamicCrossSectionsManager – методы

AddDynamicCrossSection – Добавить динамическое сечение

Интерфейс

Синтаксис Automation:

LPDISPATCH AddDynamicCrossSection();

Синтаксис COM :

HRESULT AddDynamicCrossSection(IDynamicCrossSection * * Result);

Возвращаемое значение:

- Указатель на интерфейс IDynamicCrossSection.

Интерфейс IDynamicCrossSectionStepParametersByFreePlane

Интерфейс параметров шага динамического сечения по произвольной плоскости.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStepParametersByFreePlane

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionStep::Parameters.

Версия Компас v18.1

IDynamicCrossSectionStepParametersByFreePlane – свойства

LocalCoordinateSystem – Локальная система координат

Интерфейс

Тип данных: Указатель на интерфейс ILocalCoordinateSystem.

Синтаксис Automation:

LocalCoordinateSystem = Получить свойство (*)
Object.LocalCoordinateSystem

LocalCoordinateSystem = Object.GetLocalCoordinateSystem()	Получить свойство (**)

Синтаксис COM:

Object.get_LocalCoordinateSystem(&LocalCoordinateSystem)	Получить свойство

Примечание:

Свойство доступно только для чтения.

IDynamicCrossSectionStepParametersByFreePlane – методы

ReverseDirection – Изменить направление отсечения

Интерфейс

Синтаксис Automation:

BOOL ReverseDirection();

Синтаксис COM:

HRESULT ReverseDirection(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс IDynamicCrossSectionStepParametersByOffsetPlane

Интерфейс параметров шага динамического сечения по смещенной плоскости.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStepParametersByOffsetPlane

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionStep::Parameters.

Версия Компас v18.1

IDynamicCrossSectionStepParametersByOffsetPlane – свойства

BaseStep – Базовый шаг

Интерфейс

Тип данных: Указатель на интерфейс IDynamicCrossSectionStep

Синтаксис Automation:

BaseStep = Object.BaseStep	Получить свойство (*)
Object.BaseStep = BaseStep	Установить свойство (*)
BaseStep = Object.GetBaseStep()	Получить свойство (**)
Object.SetBaseStep(BaseStep)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseStep(&BaseStep)	Получить свойство
Object.put_BaseStep(BaseStep)	Установить свойство

OffsetPlane – Параметры смещенной плоскости

Интерфейс

Тип данных: Указатель на интерфейс IPlane3DByOffset.

Синтаксис Automation:

OffsetPlane = Object.OffsetPlane	Получить свойство (*)
OffsetPlane = Object.GetOffsetPlane()	Получить свойство (**)

Синтаксис COM:

Object.get_OffsetPlane(&OffsetPlane)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PlaneBuildingType – Тип шага динамического сечения

Интерфейс

Тип данных: из перечисления ksCrossSectionPlaneBuildingTypeEnum.

Синтаксис Automation:

PlaneBuildingType = Object.PlaneBuildingType	Получить свойство (*)
Object.PlaneBuildingType = PlaneBuildingType	Установить свойство (*)

PlaneBuildingType = Object.GetPlaneBuildingType()	Получить свойство (**)
Object.SetPlaneBuildingType(PlaneBuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_PlaneBuildingType(&PlaneBuildingType)	Получить свойство
Object.put_PlaneBuildingType(PlaneBuildingType)	Установить свойство

IDynamicCrossSectionStepParametersByOffsetPlane – методы

ReverseDirection – Изменить направление отсечения

Интерфейс

Синтаксис Automation :

BOOL ReverseDirection();

Синтаксис COM:

HRESULT ReverseDirection(BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейс IDynamicCrossSectionStepParametersByRotatedPlane

Интерфейс параметров шага динамического сечения по плоскости под углом.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStepParametersByRotatedPlane

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionStep::Parameters.

Версия Компас v18.1

IDynamicCrossSectionStepParametersByRotatedPlane – свойства

BaseStep – Базовый шаг

Интерфейс

Тип данных: Указатель на интерфейс IDynamicCrossSectionStep.

Синтаксис Automation:

BaseStep = Object.BaseStep	Получить свойство (*)
Object.BaseStep = BaseStep	Установить свойство (*)
BaseStep = Object.GetBaseStep()	Получить свойство (**)
Object.SetBaseStep(BaseStep)	Установить свойство (**)

Синтаксис COM:

Object.get_BaseStep(&BaseStep)	Получить свойство
Object.put_BaseStep(BaseStep)	Установить свойство

Plane – Плоскость под углом к другой плоскости

Интерфейс

Тип данных: Указатель на интерфейс IPlane3DByAngle.

Синтаксис Automation:

Plane = Object.Plane	Получить свойство (*)
Plane = Object.GetPlane()	Получить свойство (**)

Синтаксис COM:

Object.get_Plane(&Plane)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PlaneBuildingType – Тип шага динамического сечения

Интерфейс

Тип данных: из перечисления ksCrossSectionPlaneBuildingTypeEnum.

Синтаксис Automation:

PlaneBuildingType = Object.PlaneBuildingType	=	Получить свойство (*)
Object.PlaneBuildingType = PlaneBuildingType	=	Установить свойство (*)
PlaneBuildingType = Object.GetPlaneBuildingType()	=	Получить свойство (**)
Object.SetPlaneBuildingType(PlaneBuildingType)		Установить свойство (**)

Синтаксис COM:

Object.get_PlaneBuildingType(&PlaneBuildingType)	Получить свойство
Object.put_PlaneBuildingType(PlaneBuildingType)	Установить свойство

Интерфейс IDynamicCrossSectionStepParametersByZone

Интерфейс параметров шага динамического сечения по зоне.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStepParametersByZone

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionStep::Parameters.

Версия Компас v18.1

IDynamicCrossSectionStepParametersByZone – свойства

Zone – Зона

Интерфейс

Тип данных: Указатель на интерфейс IZone.

Синтаксис Automation:

Zone = Object.Zone	Получить свойство (*)
Object.Zone = Zone	Установить свойство (*)
Zone = Object.GetZone()	Получить свойство (**)
Object.SetZone(Zone)	Установить свойство (**)

Синтаксис COM:

Object.get_Zone(&Zone)	Получить свойство
Object.put_Zone(Zone)	Установить свойство

Интерфейс IDynamicCrossSectionStepParametersByBorderPoints

Интерфейс параметров шага динамического сечения по габаритному параллелепипеду, заданному точками.

Иерархия:

IDispatch

IKompasAPIObject

IDynamicCrossSectionStepParametersByBorderPoints

Данный интерфейс можно получить с помощью метода IDynamicCrossSectionStep::Parameters.

Версия Компас v18.1

IDynamicCrossSectionStepParametersByBorderPoints – свойства

AssociationObject – Установить опорный объект для вершины

Интерфейс

Тип данных: Указатель на интерфейс IModelObject.

Синтаксис Automation:

AssociationObject	=	Получить свойство (*)
Object.AssociationObject(First)		
Object.AssociationObject(First)	=	Установить свойство (*)
AssociationObject		
AssociationObject	=	Получить свойство (**)
Object.GetAssociationObject(First)		
Object.SetAssociationObject(First, AssociationObject)		Установить свойство (**)

Синтаксис COM:

Object.get_AssociationObject(First, &AssociationObject)		Получить свойство
Object.put_AssociationObject(First, AssociationObject)		Установить свойство

Входные параметры:

BOOL	- First - TRUE - первая вершина, FALSE - вторая вершина.

BuildingType – Вариант построения габарита

Интерфейс

Тип данных: из перечисления ksGabaritBuildingTypeEnum.

Синтаксис Automation:

BuildingType = Object.BuildingType	Получить свойство (*)
Object.BuildingType = BuildingType	Установить свойство (*)
BuildingType = Object.GetBuildingType()	Получить свойство (**)
Object.SetBuildingType(BuildingType)	Установить свойство (**)

Синтаксис COM:

Object.get_BuildingType(&BuildingType)	Получить свойство
Object.put_BuildingType(BuildingType)	Установить свойство

PointParameters – Получить интерфейс параметров точки

Интерфейс

Тип данных: Указатель на интерфейс IKompasAPIObject.

Синтаксис Automation:

PointParameters Object.PointParameters(First)	=	Получить свойство (*)
PointParameters Object.GetPointParameters(First)	=	Получить свойство (**)

Синтаксис COM:

Object.get_PointParameters(First, &PointParameters)	Получить свойство

Примечание:

Свойство доступно только для чтения.

PointType – Тип параметров построения точки

Интерфейс

Тип данных: из перечисления ksPoint3DTypeEnum.

Синтаксис Automation:

PointType = Object.PointType(First)	Получить свойство (*)
Object.PointType(First) = PointType	Установить свойство (*)
PointType = Object.GetPointType(First)	Получить свойство (**)
Object.SetPointType(First, PointType)	Установить свойство (**)

Синтаксис COM:

Object.get_PointType(First, &PointType)	Получить свойство
Object.put_PointType(First, PointType)	Установить свойство

Входные параметры:

BOOL	- First - TRUE - первая вершина, FALSE - вторая вершина.

IDynamicCrossSectionStepParametersByBorderPoints – методы

GetPoint – Получить координату точки

Интерфейс

Синтаксис Automation:

BOOL GetPoint(BOOL First, double * X, double * Y, double * Z);

Синтаксис COM:

HRESULT GetPoint(BOOL First, double * X, double * Y, double * Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

First	- TRUE - первая вершина, FALSE - вторая вершина.

Выходные параметры:

X,Y, Z	- координаты вершины.

SetPoint – Установить координату точки

Интерфейс

Синтаксис Automation:

BOOL SetPoint(BOOL First, double X, double Y, double Z);

Синтаксис COM:

HRESULT SetPoint(BOOL First, double X, double Y, double Z, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

First	- TRUE - первая вершина, FALSE - вторая вершина.
X,Y, Z	- координаты вершины.

Дерево СЧИ

Интерфейс IProductDataManager

Менеджер Дерева СЧИ.

Иерархия:

IDispatch

IProductDataManager

Интерфейс является дополнительным к интерфейсу IKompasDocument. Данный интерфейс можно получить посредством вызова метода IUnknown::QueryInterface (const GUID far& iid, void** pif).

КОМПАС версия v18

IProductDataManager – свойства

Geometry – Список подключенных документов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

Geometry = Object.Geometry(PropObject)	Получить свойство (*)
Object.Geometry(PropObject) = Geometry	Установить свойство (*)
Geometry = Object.GetGeometry(PropObject)	Получить свойство (**)
Object.SetGeometry(PropObject, Geometry)	Установить свойство (**)

Синтаксис COM:

Object.get_Geometry(PropObject, &Geometry)	Получить свойство
Object.put_Geometry(PropObject, Geometry)	Установить свойство

Входные параметры:

IPropertyKeeper * PropObject	- указатель на объект СЧИ.

Примечание:

Свойство возвращает переменную типа VARIANT на безопасный массив SAFEARRAY (VT_ARRAY | VT_DISPATCH).

MetaProductInfo – Метаданные дерева СЧИ

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

MetaProductInfo = Object.MetaProductInfo	=	Получить свойство (*)
Object.MetaProductInfo = MetaProductInfo	=	Установить свойство (*)
MetaProductInfo = Object.GetMetaProductInfo()	=	Получить свойство (**)
Object.SetMetaProductInfo(MetaProductInfo)		Установить свойство (**)

Синтаксис COM:

Object.get_MetaProductInfo(&MetaProductInfo)	Получить свойство
Object.put_MetaProductInfo(MetaProductInfo)	Установить свойство

ObjectAttachedDocuments – Список подключенных к объекту документов

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ObjectAttachedDocuments = Object.ObjectAttachedDocuments(PropObject)	=	Получить свойство (*)
Object.ObjectAttachedDocuments(PropObject) = ObjectAttachedDocuments		Установить свойство (*)
ObjectAttachedDocuments = Object.GetObjectAttachedDocuments(PropObject)	=	Получить свойство (**)
Object.SetObjectAttachedDocuments(PropObject, ObjectAttachedDocuments)		Установить свойство (**)

Синтаксис COM:

Object.get_ObjectAttachedDocuments(PropObject, &ObjectAttachedDocuments)	Получить свойство
Object.put_ObjectAttachedDocuments(PropObject, ObjectAttachedDocuments)	Установить свойство

Входные параметры:

IPropertyKeeper * PropObject	- указатель на объект СЧИ.

Примечание:

Свойство возвращает переменную типа VARIANT на безопасный массив имен файлов, связанных с объектом СЧИ SAFEARRAY (VT_ARRAY | VT_BSTR).

ObjectMetaProductInfo – Метаданные объекта дерева СЧИ

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ObjectMetaProductInfo = Object.ObjectMetaProductInfo(PropObject)	Получить свойство (*)
Object.ObjectMetaProductInfo(PropObject) = ObjectMetaProductInfo	Установить свойство (*)
ObjectMetaProductInfo = Object.GetObjectMetaProductInfo(PropObject)	Получить свойство (**)
Object.SetObjectMetaProductInfo(PropObject, ObjectMetaProductInfo)	Установить свойство (**)

Синтаксис COM:

Object.get_ObjectMetaProductInfo(PropObject, &ObjectMetaProductInfo)	Получить свойство
Object.put_ObjectMetaProductInfo(PropObject, ObjectMetaProductInfo)	Установить свойство

Входные параметры:

IPropertyKeeper * PropObject	- указатель на объект СЧИ.

ProductObjects – Получить массив объектов дерева СЧИ

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ProductObjects Object.ProductObjects(Filter)	=	Получить свойство (*)
ProductObjects Object.GetProductObjects(Filter)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ProductObjects(Filter, &ProductObjects)	Получить свойство

Входные параметры:

long Filter	- фильтр по типам объектов.

Примечание:

Свойство доступно только для чтения.

Фильтр формируется из комбинации нужных типов объектов из перечисления ksProductObjectTypeEnum.

ProductObject – Получить объект дерева СЧИ по идентификатору

Интерфейс...

Тип данных: Указатель на интерфейс IPropertyKeeper

Синтаксис Automation:

ProductObject Object.ProductObject(UniqueMetaObjectKey)	=	Получить свойство (*)

ProductObject = Object.GetProductObject(UniqueMetaObjectKey)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ProductObject(UniqueMetaObjectKey, &ProductObject)	Получить свойство

Входные параметры:

BSTR UniqueMetaObjectKey	- уникальный идентификатор объекта.

Примечание:

Свойство доступно только для чтения.

ReferenceData – Справочные данные свойств

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ReferenceData = Object.ReferenceData	=	Получить свойство (*)
Object.ReferenceData = ReferenceData	=	Установить свойство (*)
ReferenceData = Object.GetReferenceData()	=	Получить свойство (**)
Object.SetReferenceData(ReferenceData)		Установить свойство (**)

Синтаксис COM:

Object.get_ReferenceData(&ReferenceData)	Получить свойство
Object.put_ReferenceData(ReferenceData)	Установить свойство

ReferenceDataInfo – Справочные данные свойства

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

ReferenceDataInfo = Object.ReferenceDataInfo(ReferenceDataType, ReferenceDataId)	Получить свойство (*)
Object.ReferenceDataInfo(ReferenceDataType, ReferenceDataId) = ReferenceDataInfo	Установить свойство (*)
ReferenceDataInfo = Object.GetReferenceDataInfo(ReferenceDataType, ReferenceDataId)	Получить свойство (**)
Object.SetReferenceDataInfo(ReferenceDataType, ReferenceDataId, ReferenceDataInfo)	Установить свойство (**)

Синтаксис COM:

Object.get_ReferenceDataInfo(ReferenceDataType, ReferenceDataId, &ReferenceDataInfo)	Получить свойство
Object.put_ReferenceDataInfo(ReferenceDataType, ReferenceDataId, ReferenceDataInfo)	Установить свойство

Входные параметры:

BSTR ReferenceDataType	- тип справочных данных,
BSTR ReferenceDataId	- идентификатор справочных данных.

ReferenceDataIds – Идентификаторы справочных данных

Интерфейс...

Тип данных: VARIANT

Синтаксис Automation:

ReferenceDataIds Object.ReferenceDataIds(ReferenceDataType)	=	Получить свойство (*)
ReferenceDataIds Object.GetReferenceDataIds(ReferenceDataType)	=	Получить свойство (**)

Синтаксис COM:

Object.get_ReferenceDataIds(ReferenceDataType, &ReferenceDataIds)	Получить свойство

Примечание:

Свойство доступно только для чтения.

Свойство возвращает переменную типа VARIANT на безопасный массив идентификаторов справочных данных SAFEARRAY (VT_ARRAY | VT_BSTR).

IProductDataManager – методы

AddProductObject – Добавить объект дерева СЧИ по идентификатору

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddProductObject(IPropertyKeeper * Parent, BSTR Name, ksProductObjectTypeEnum ObjectType);

Синтаксис COM:

HRESULT AddProductObject(IPropertyKeeper * Parent, BSTR Name, ksProductObjectTypeEnum ObjectType, IPropertyKeeper * * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

Parent	- владелец информационного объекта,
Name	- имя объекта,
ObjectType	- тип объекта.

Примечание:

В зависимости от типа информационного объекта выбирается владелец объекта. Например, для вставок моделей и тел владелец - исполнение.

AddReferenceData – Добавить справочные данные

Интерфейс...

Синтаксис Automation:

BSTR AddReferenceData(BSTR ReferenceDataType, BSTR ReferenceDataInfo);

Синтаксис COM:

HRESULT AddReferenceData(BSTR ReferenceDataType, BSTR ReferenceDataInfo, BSTR * Result);

Возвращаемое значение:

	- идентификатор справочных данных.

Входные параметры:

ReferenceDataType	- тип справочных данных,
ReferenceDataInfo	- заголовок справочных данных.

DeleteProductObject – Удалить объект из дерева СЧИ по идентификатору

Интерфейс...

Синтаксис Automation:

BOOL DeleteProductObject(BSTR UniqueMetaObjectKey);

Синтаксис COM:

HRESULT DeleteProductObject(BSTR UniqueMetaObjectKey, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

UniqueMetaObjectKey	- идентификатор объекта СЧИ.

DeleteReferenceData – Удалить справочные данные

Интерфейс...

Синтаксис Automation:

BOOL DeleteReferenceData(BSTR ReferenceDataType, BSTR ReferenceDataId);

Синтаксис COM:

HRESULT DeleteReferenceData(BSTR ReferenceDataType, BSTR ReferenceDataId, BOOL * Result);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

ReferenceDataType	- тип справочных данных,
ReferenceDataId	- идентификатор справочных данных.

Константы API версии 7

ButtonTypeEnum – Типы кнопок для элемента панели свойств "Набор кнопок"

ksPushButton	0	Набор обычных нефиксируемых кнопок
ksCheckButton	1	Набор фиксируемых кнопок
ksRadioButton	2	Набор кнопок с возможностью фиксации только одной кнопки

CheckStateEnum – Возможные состояния элемента управления Панели свойств "Переключатель состояния поля ввода"

ksCheckUndefined	0	Значение поля не задано; на переключателе пусто,
ksCheckCurrent	1	Поле ввода активно; на переключателе значок "галочка",
ksCheckFixed	2	Содержание поля ввода зафиксировано; на переключателе перекрестие.
ksCheckVariable	3	Значение поля ввода задано; на переключателе пусто.

ControlTypeEnum – Типы элементов управления Панели свойств

ksControlUnknown	0	Неизвестный тип	
ksControlSeparator	1	Разделитель (сепаратор)	IPropertySeparator
ksControlEditInt	2	Редактор целочисленных значений	IPropertyEdit
ksControlEditReal	3	Редактор дробных значений	IPropertyEdit
ksControlEditStr	4	Редактор строковых значений	IPropertyEdit
ksControlListInt	5	Раскрывающийся список целочисленных значений	IPropertyList
ksControlListReal	6	Раскрывающийся список дробных значений	IPropertyList

ksControlListStr	7	Раскрывающийся список строковых значений	IPropertyList
ksControlCheckBox	8	Опция	IPropertyCheckBox
ksControlMultiButton	9	Набор кнопок	IPropertyMultiButton
ksControlGrid	10	Сетка	IPropertyGrid
ksControlSlideBox	11	Окно отображения слайда, растрового изображения, группы или файла документа КОМПАС	IPropertySlideBox
ksControlUser	12	Пользовательский элемент управления	IPropertyUserControl
ksControlTextButton	13	Кнопка с текстом	IPropertyTextButton
ksControlSpinInt	14	Счетчик целочисленных значений с полем ввода	IPropertySpinEdit
ksControlSpinReal	15	Счетчик дробных значений с полем ввода	IPropertySpinEdit
ksControlFileName	16	Выбор файла	IPropertyFileName
ksControlColor	17	Выбор цвета	IPropertyColor
ksControlEditList	18	Список	IPropertyEdit
ksControlEditLength	19	Редактор длины	IPropertyEdit
ksControlEditAngle	20	Редактор угла	IPropertyEdit
ksControlEditPoint	21	Редактор координат	IPropertyEdit
ksControlListLength	22	Комбобокс длин	IPropertyList
ksControlListAngle	23	Комбобокс углов	IPropertyList
ksControlBmpList	24	Комбобокс со строкой и битмапом	IPropertyBmpList
ksControlLibExplorer	25	Отображение библиотеки документов	
ksControlListScale	26	Комбобокс со списком масштабов	IPropertyList
ksControlLineStyle	27	Комбобокс со списком стилей линий	IPropertyStyleList
ksControlOpticalProps	28	Контрол оптических свойств	IPropertyOpticalProps
ksControlEditCheckBox	29	Редактор строковых значений с чекбоксом	IPropertyEdit
ksControlPointStyle	30	Комбобокс со списком стилей точек	IPropertyStyleList
ksControlPointStyle3D	31	Комбобокс со списком стилей точек 3D	IPropertyStyleList
ksControlLineStyle3D	32	Комбобокс со списком стилей линий 3D	IPropertyStyleList
ksControlHatchStyle	33	Комбобокс со списком стилей штриховок	IPropertyStyleList
ksControlGroupBegin	34	Начало группы контролов	
ksControlGroupEnd	35	Конец группы контролов	
ksControlTwinSwitcher	36	Переключатель	
ksControlPoint3D	37	Точка 3D	
ksControlPreviewText	38	Предпросмотр текста	
ksControlAggregateTriple	40	Составной контрол из трех контролов	

ksControlBasePoint	41	Базовая точка
ksControlLinkButton	42	Набор кнопок в виде ссылок
ksControlMarking	43	Контрол обозначение

ConvertCoordTypeEnum – Типы преобразования логических координат в координаты документа

kcDocument	1	В СК документа.
kcGeoView	2	В геометрическую точку текущего вида.
kcCurrentPlane	3	В СК текущего плоского объекта.

DefaultFixTypeEnum – Тип фиксированности для умолчательных элементов управления Панели свойств

ksAllFixOff	-1	Все элементы управления расфиксированы
ksAllFix	0	Все элементы управления фиксированы
ksPointFix	1	Фиксирована точка
ksAngleFix	2	Фиксирован угол

DocumentCloseOptions – Действия при закрытии документа КОМПАС

kdDoNotSaveChanges	0	Закрыть документ без сохранения, если документ был изменен.
kdSaveChanges	1	Закрыть документ, сохранив сделанные изменения.
kdPromptToSaveChanges	2	Выдать запрос на сохранение документа, если он изменен.

DocumentTypeEnum – Типы документов КОМПАС

ksDocumentUnknown	0	Неизвестный тип
ksDocumentDrawing	1	Чертеж
ksDocumentFragment	2	Фрагмент
ksDocumentSpecification	3	Спецификация
ksDocumentPart	4	Деталь

ksDocumentAssembly	5	Сборка
ksDocumentTextual	6	Текстовый документ
ksDocumentTechnologyAssembly	7	Технологическая сборка

FilterConditionStateEnum – Состояние параметра в условии фильтрации слоев

ksStateUndefined	-1	Состояние не определено
ksStateFALSE	0	Состояние FALSE
ksStateTRUE	1	Состояние TRUE

FrameRegimeEnum – Режим отображения окна

ksFrameMinimize	0	Свернуть окно
ksFrameMaximize	1	Развернуть окно
ksFrameRestore	2	Восстановить окно

KompasAPIObjectTypeEnum – Типы объектов КОМПАС API

ksObjectUnknown	0	Неизвестный тип
ksObjectApplication	10001	Приложение - основной объект API
ksObjectDocuments	10002	Коллекция документов, открытых в приложении
ksObjectKompasError	10003	Информация о ошибке системы КОМПАС
ksObjectProcessParam	10004	Параметры процесса
ksObjectPropertyTabs	10005	Коллекция закладок панели свойств
ksObjectPropertyTab	10006	Закладка панели свойств
ksObjectPropertyControls	10007	Коллекция элементов управления вкладки Панели свойств
ksObjectPropertySeparator	10008	Разделитель (сепаратор)
ksObjectPropertyEdit	10009	Редактор
ksObjectPropertyList	10010	Раскрывающийся список
ksObjectPropertyCheckBox	10011	Опция
ksObjectPropertyMultiButton	10012	Набор кнопок
ksObjectPropertyUserControl	10013	Пользовательский элемент управления
ksObjectPropertyBmpList	10014	Комбобокс со строкой и битмапом
ksObjectPropertySlideBox	10016	Окно отображения слайда, растрового изображения, файла документа КОМПАС или группы файлов.
ksObjectPropertyGrid	10017	Сетка
ksObjectDocumentFrame	10018	Окно документа

ksObjectDocumentFrames	10019	Коллекция окон документа
ksObjectPropertyManager	10020	Панель свойств
ksObjectDrawingDocument	10021	Документ - Чертеж
ksObjectFragmentDocument	10022	Документ - Фрагмент
ksObjectSpcDocument	10023	Документ - Спецификация
ksObjectPartDocument	10024	Документ - Деталь
ksObjectAssemblyDocument	10025	Документ - Сборка
ksObjectTextDocument	10026	Документ - Текстовый
ksObjectPropertyTextButton	10027	Кнопка с текстом
ksObjectPropertySpinEdit	10028	Поле ввода со счетчиком
ksObjectViewsAndLayersManager	10029	Менеджер слоев и видов графического документа
ksObjectViews	10030	Коллекция видов
ksObjectView	10031	Вид
ksObjectAssociationView	10032	Ассоциативный вид
ksObjectLayerGroups	10033	Коллекция групп слоев
ksObjectDrawingOblects	10034	Коллекция объектов графического документа
ksObjectLayerGroup	10035	Группа слоев
ksObjectLayers	10036	Коллекция слоев
ksObjectLayer	10037	Слой
ksObjectLayerFilterCondition	10038	Условие фильтрации слоев
ksObjectLayerFilterConditions	10039	Коллекция условий фильтрации слоев
ksObjectDocumentSettings	10040	Настройки документа
ksObjectDocument2DSettings	10041	Настройки графического документа
ksObjectDrawingDocumentSettings	10042	Настройки чертежа
ksObjectFragmentDocumentSettings	10043	Настройки фрагмента
ksObjectDocument3DSettings	10045	Настройки 3D документа
ksObjectLibraryManager	10050	Менеджер библиотек
ksObjectProcedure	10051	Процедура прикладной библиотеки
ksObjectProceduresLibraries	10052	Коллекция прикладных библиотек
ksObjectProceduresLibrary	10053	Прикладная библиотека
ksObjectProcedures	10054	Коллекция процедур прикладной библиотеки
ksObjectInsertsLibraries	10055	Коллекция библиотек элементов (документов)
ksObjectInsertsLibrary	10056	Библиотека элементов (документов)
ksObjectInserts	10057	Коллекция элементов (документов) библиотеки элементов (документов)
ksObjectInsert	10058	Элемент библиотеки элементов (документов)
ksObjectSpecificationDescriptions	10059	Коллекция описаний спецификации
ksObjectSpecificationDescription	10060	Описание спецификации
ksObjectSpecificationStyle	10061	Стиль спецификации
ksObjectSpecificationColumnStyles	10062	Коллекция стилей колонок спецификации
ksObjectSpecificationColumnStyle	10063	Стиль колонки спецификации
ksObjectSpecificationSectionStyles	10064	Коллекция стилей разделов спецификации
ksObjectSpecificationSectionStyle	10065	Стиль раздела спецификации
ksObjectSpecificationTuning	10066	Настройки спецификации
ksObjectSpecificationTuningSections	10067	Коллекция настроек разделов спецификации
ksObjectSpecificationTuningSection	10068	Настройка раздела спецификации

ksObjectSpecificationSubsections	10069	Коллекция подразделов спецификации
ksObjectSpecificationSubsection	10070	Подраздел спецификации
ksObjectAdditionalBlockStyles	10071	Коллекция стилей блоков дополнительных разделов
ksObjectAdditionalBlockStyle	10072	Стиль блока дополнительных разделов
ksObjectAdditionalBlockTunings	10073	Коллекция настроек блоков дополнительных разделов
ksObjectAdditionalBlockTuning	10074	Настройка блока дополнительных разделов
ksObjectAdditionalBlockSectionTunings	10075	Коллекция настроек разделов блока дополнительных разделов
ksObjectAdditionalBlockSectionTuning	10076	Настройка раздела блока дополнительных разделов
ksObjectSheetFormat	10077	Формат листа
ksObjectTextStyle	10078	Параметры стиля текста
ksObjectFont	10079	Параметры шрифта
ksObjectTabulators	10080	Коллекция позиции табуляторов
ksObjectTabulator	10081	Позиция табулятора
ksObjectSpecificationBaseObjects	10083	Коллекция базовых объектов спецификации
ksObjectSpecificationCommentObjects	10084	Коллекция вспомогательных объектов спецификации
ksObjectSpecificationObject	10085	Объект спецификации
ksObjectSpecificationBaseObject	10086	Базовый объект спецификации
ksObjectSpecificationCommentObject	10087	Вспомогательный объект спецификации
ksObjectSpecificationColumns	10088	Коллекция колонок объекта спецификации
ksObjectSpecificationColumn	10089	Колонка объекта спецификации
ksObjectSpecificationColumnItems	10090	Коллекция элементов колонки объекта спецификации
ksObjectSpecificationColumnItem	10091	Элемент колонки объекта спецификации
ksObjectAttachedDocuments	10092	Коллекция присоединенных документов к объекту спецификации
ksObjectAttachedDocument	10093	Параметры присоединенного документа к объекту спецификации
ksObjectPropertyFileName	10094	Выбор файла
ksObjectPropertyEditList	10095	Список
ksObjectLayoutSheets	10096	Коллекция листов оформления
ksObjectLayoutSheet	10097	Параметры листа оформления
ksObjectConverter	10098	Конвертер КОМПАС-документов
ksObjectCheckSum	10099	Контрольная сумма
ksObjectProgressBar	10100	Индикатор процесса
ksObjectPropertyEditList	10101	Список
ksObjectPropertyLibExplorer	10102	Отображение библиотеки документов
ksObjectVariable7	10103	Параметрическая переменная модели
ksObjectInsertionParameters	10104	Параметры вставки фрагмента и растрового объекта в текст
ksObjectMath2D	10105	Математика 2D
ksObjectSelectionManager	10106	Менеджер селектированных объектов
ksObjectChooseManager	10107	Менеджер выбора (подсветки) объектов
ksObjectStamp	10108	Штамп
ksObjectPropertyStyleList	10109	Комбобокс со стилем
ksObjectPrintJob	10110	Задание на печать

ksObjectPrintJob_Sheet	10111	Задание на печать::Интерфейс листа документа
ksObjectPropertyOpticalProps	10112	Контрол оптических свойств
ksObjectPropertyEditCheckBox	10113	Объединенный редактор с чекбоксом
ksObjectPropertyGroupBegin	10116	Начало группы контролов
ksObjectPropertyGroupEnd	10117	Конец группы контролов
ksObjectThreadPattern	10118	Параметры стандарта резьбы
ksObjectThreadDialogParam	10119	Параметры диалога выбора стандарта резьбы
ksObjectPropertyPreviewText	10120	Предпросмотр текста
ksObjectPropertyAggregateControl	10121	Составной контрол
ksObjectPropertyLinkButton	10122	Набор кнопок в виде ссылок
ksObjectPropertyMarking	10123	Контрол обозначение
ksObjectContentDialogParam	10124	Параметры диалога с произвольным наполнением
ksObjectPropertyBasePoint	10125	Базовая точка
ksObjectTextDocumentSection	10126	Раздел текстового документа
ksObjectUserDataStoragesMng	10500	Менеджер пользовательских хранилищ
ksObjectUserDataStorages	10501	Коллекция объектов пользовательского хранилища
ksObjectUserDataStorage	10502	Объект пользовательского хранилища данных
ksObjectAttribute	10504	Атрибут
ksObjectColumnInfo	10505	Параметры столбца табличного атрибута
ksObjectAttributeType	10506	Параметры типа атрибута
ksObjectAttrTypeMng	10507	Менеджер типов атрибутов
ksObjectProperty	10508	Свойство
ksObjectPropertyMng	10509	Менеджер свойств
ksObjectReportProcess	10510	Интерфейс для управления процессом Создать отчет
ksObjectReport	10511	Интерфейс Отчета
ksObjectReportStyle	10512	Интерфейс стиля Отчета
ksObjectReportStyleColumn	10513	Интерфейс колонки стиля Отчета
ksObjectPropertyKeeper	10514	Объект дерева СЧИ
ksObjectStyles	10515	Коллекция стилей
ksObjectCurvesStyles	10516	Коллекция стилей кривых
ksObjectHatchsStyles	10517	Коллекция стилей штриховок
ksObjectTextsStyles	10518	Коллекция стилей текстов
ksObjectStyle	10519	Стиль
ksObjectCurveStyle	10520	Стиль кривой
ksObjectHatchStyle	10521	Стиль штриховки
ksObjectSpecificationStyles	10522	Коллекция стилей спецификации
ksObjectText	10700	Текст
ksObjectTextLine	10701	Строка текста
ksObjectTextItem	10702	Компонент строки текста
ksObjectTableCell	10703	Ячейка таблицы
ksObjectExternalTessellation	10704	Объект с внешней триангуляцией
ksObjectExternalGDI	10709	Внешний GDI-объект
ksObjectLibArraySettings	10710	Интерфейс для выбора состояний библиотек в настройках
ksObjectTextTable	10711	Таблица в тексте текстового документа
ksObjectPropertyTwinSwitcher	10713	Переключатель
ksObjectPropertyPoint3D	10714	Точка 3D
ksObjectPart7	11000	3D компонент
ksObjectModelObject	11001	3D объект

ks3dMateConstraint	11002	3D сопряжение
ksObjectParts7	11003	Коллекция 3D компонентов
ksObjectVariableTable	11004	Таблица переменных
ksObjectProgressBar	11005	Индикатор прогресса
ksObjectLineDimensions3D	11030	Коллекция линейных размеров 3D
ksObjectBaseLineDimension3D	11031	Линейный размер 3D (от отрезка до точки)
ksObjectLineDimension3D	11032	Линейный размер 3D (на плоскости)
ksObjectRadialDimension3D	11033	Радиальный размер 3D
ksObjectDiametralDimension3D	11034	Диаметральный размер 3D
ksObjectRadialDimensions3D	11035	Коллекция радиальных размеров 3D
ksObjectDiametralDimensions3D	11036	Коллекция диаметральных размеров 3D
ksObjectAngleDimension3D	11037	Угловой размер 3D
ksObjectAngleDimensions3D	11038	Коллекция угловых размеров 3D
ksObjectLocalCoordinateSystems	11039	Коллекция локальных систем координат
ksObjectLocalCoordinateSystem	11040	Локальная система координат
ksObjectLocalCSAxesDirectionParam	11041	Параметры для типа ориентации ЛСК-направление осей
ksObjectLocalCSRotateParam	11042	Параметры для типа ориентации ЛСК-вращение вокруг осей СК
ksObjectLocalCSEulerParam	11043	Параметры для типа ориентации ЛСК-система углов Эйлера
ksObjectSpline3D	11044	Сплайн
ksObjectSplines3D	11045	Коллекция сплайнов
ksObjectCurveVertexParam	11046	Параметры вершины кривой
ksObjectPolyLines	11047	Коллекция 3D ломаных
ksObjectPolyLine	11048	3D ломаная
ksObjectLeaders3D	11049	Коллекция линий-выносок 3D
ksObjectLeader3D	11050	Линия-выноска 3D
ksObjectMarkLeader3D	11051	Знак маркировки 3D
ksObjectRough3D	11052	Обозначение 3D шероховатости
ksObjectRoughs3D	11053	Коллекция обозначений 3D шероховатости
ksObjectPositionLeader3D	11054	Линия-выноска для обозначения позиции 3D
ksObjectBrandLeader3D	11055	Знак клеймения 3D
ksObjectBase3D	11056	Обозначение 3D базы
ksObjectBases3D	11057	Коллекция обозначений 3D базы
ksObjectTolerances3D	11058	Коллекция допуска формы 3D
ksObjectTolerance3D	11059	Обозначение допуска формы 3D
ksObjectControlPoints	11060	Коллекция контрольных точек
ksObjectControlPoint	11061	Контрольная точка
ksObjectConjunctivePoints	11062	Коллекция присоединительных точек
ksObjectConjunctivePoint	11063	Присоединительная точка
ksObjectSplitLines	11064	Коллекция линий разъема
ksObjectSplitLine	11065	Линия разъема
ksObjectSurfacePatches	11066	Коллекция заплаток
ksObjectSurfacePatch	11067	Заплатка
ksObjectFaceRemovers	11068	Коллекция операций удаления граней
ksObjectFaceRemover	11069	Операция удаления граней
ksObjectSurfaceSewers	11070	Коллекция операций сшивки поверхностей
ksObjectSurfaceSewer	11071	Операция сшивки поверхностей
ksObjectNurbsSurfaces	11072	Коллекция NURBS-поверхностей
ksObjectNurbsSurface	11073	NURBS-поверхность

ksObjectSurfacesIntersectionCurves	11074	Коллекция кривых пересечений поверхностей
ksObjectSurfacesIntersectionCurve	11075	Кривая пересечения поверхностей
ksObjectEquidistants3D	11076	Коллекция эквидистант 3D
ksObjectEquidistant3D	11077	Эквидистанта 3D
ksObjectTrimmedCurves	11078	Коллекция операций усечения кривой
ksObjectTrimmedCurve	11079	Операция усечения кривой
ksObjectFeaturePatterns	11080	Коллекция операций копирования
ksObjectLinearPattern	11081	Массив операций по сетке
ksObjectCircularPattern	11082	Массив операций по концентрической сетке
ksObjectPathPattern	11083	Массив операций вдоль кривой
ksObjectPartsLinearPattern	11084	Массив компонентов по сетке для сборки
ksObjectPartsCircularPattern	11085	Массив компонентов по концентрической сетке для сборки
ksObjectPartsPathPattern	11086	Массив компонентов вдоль кривой для сборки
ksObjectAuxLinearPattern	11087	Массив вспомогательной геометрии по сетке
ksObjectAuxCircularPattern	11088	Массив вспомогательной геометрии по концентрической сетке
ksObjectAuxPathPattern	11089	Массив вспомогательной геометрии вдоль кривой
ksObjectPointDrivenPattern	11090	Массив операций по точкам
ksObjectPartsPointDrivenPattern	11091	Массив компонентов по точкам
ksObjectDerivedPattern	11092	Массив по образцу
ksObjectMirrorPattern	11093	Зеркальный массив
ksObjectShellMirrorPattern	11094	Зеркально отобразить тело или операцию
ksObjectAuxMirrorPattern	11095	Зеркальный массив вспомогательной геометрии
ksObjectRuledSurfaces	11096	Коллекция операций создания линейчатой поверхности
ksObjectRuledSurface	11097	Линейчатая поверхность
ksObjectExtensionSurfaces	11098	Коллекция операций продления поверхности
ksObjectExtensionSurface	11099	Операция продления поверхности
ksObjectEquidistantSurfaces	11100	Коллекция операций построения эквидистант поверхности
ksObjectEquidistantSurface	11101	Операция построения эквидистанты поверхности
ksObjectTrimmedSurfaces	11102	Коллекция Операций усечения поверхности
ksObjectTrimmedSurface	11103	Операция усечения поверхности
ksObjectVector3D	11104	Вектор 3D
ksObjectVector3DBy2VertexesParameters	11105	Параметры вектора 3D по двум вершинам
ksObjectVector3DByCoefficientsParameters	11106	Параметры вектора 3D по коэффициентам
ksObjectVector3DBy2AnglesParameters	11107	Параметры вектора 3D по двум углам
ksObjectVector3DByObjectParameters	11108	Параметры вектора 3D по ребру или плоскости

ksObjectVector3DAlongSurfaceNormalParameters	11109	Параметры вектора 3D, перпендикулярного грани в указанной точке
ksObjectVector3DByCurveParameters	11110	Параметры вектора 3D по базисному вектору в точке кривой
ksObjectVector3DByScreenNormalParameters	11111	Параметры вектора 3D перпендикулярно плоскости экрана
ksObjectVector3DByLocalCSParameters	11112	Параметры вектора 3D по углу в плоскости СК и по оси СК
ksObjectConnectCurves	11113	Коллекция операций соединения кривых
ksObjectConnectCurve	11114	Операция соединения кривых
ksObjectFilletCurves	11115	Коллекция операций скругления кривых
ksObjectFilletCurve	11116	Операция соединения кривых
ksObjectSurfaceThickenings	11117	Коллекция операций придания толщины поверхности
ksObjectSurfaceThickening	11118	Операция придания толщины поверхности
ksObjectArcs3D	11119	Коллекция дуг 3D
ksObjectArc3D	11120	3D дуга
ksObjectAuxPointDrivenPattern	11121	Массив вспомогательной геометрии по точкам
ksObjectBodiesPointDrivenPattern	11122	Массив тел по точкам
ksObjectTablePattern	11123	Массив операций по таблице из файла
ksObjectPartsTablePattern	11124	Массив компонентов по таблице из файла
ksObjectAuxTablePattern	11125	Массив компонентов по таблице из файла
ksObjectBodiesTablePattern	11126	Массив вспомогательной геометрии по таблице из файла
ksObjectRotateds	11127	Коллекция операций вращения
ksObjectRotated	11128	Операция вращения
ksObjectCutRotated	11129	Операция 'вырезать вращением'
ksObjectExtrusionSurfaces	11130	Коллекция поверхностей выдавливания
ksObjectExtrusionSurface	11131	Операция поверхность выдавливания
ksObjectRotatedSurfaces	11132	Коллекция поверхностей вращения
ksObjectRotatedSurface	11133	Операция поверхность вращения
ksObjectPoint3DParamBySphere	11134	Точка, заданная сферическими координатами
ksObjectPoint3DParamByCylinder	11135	Точка, заданная цилиндрическими координатами
ksObjectMeshPointsSurfaces	11136	Коллекция поверхностей по сети точек
ksObjectMeshPointsSurface	11137	Поверхность по сети точек
ksObjectCloudPointsSurfaces	11138	Коллекция поверхностей по пласту (облаку) точек
ksObjectCloudPointsSurface	11139	Поверхность по пласту (облаку) точек
ksObjectImportedSurfaces	11140	Коллекция импортированных поверхностей
ksObjectImportedSurface	11141	Импортированная поверхность
ksObjectBodiesLinearPattern	11142	Массив тел по сетке
ksObjectBodiesCircularPattern	11143	Массив тел по концентрической сетке
ksObjectBodiesPathPattern	11144	Массив тел вдоль кривой
ksObjectScalings3D	11145	Коллекция операций масштабирования тел и поверхностей

ksObjectScaling3D	11146	Масштабирование тел и поверхностей
ksObjectCurveOutLine	11147	Линия очерка
ksObjectCurveOutLines	11148	Коллекция линий очерка
ksObjectCurveByLaw	11149	Кривая по закону
ksObjectCurveByLaws	11150	Коллекция кривых по закону
ksObjectIsoparametricCurve	11151	Изопараметрическая кривая
ksObjectIsoparametricCurves	11152	Коллекция изопараметрических кривых
ksObjectIsoparametricCurvesSet	11153	Группа изопараметрических кривых
ksObjectIsoparametricCurvesSets	11154	Коллекция групп изопараметрических кривых
ksObjectSplineOnSurface	11155	Сплайн по поверхности
ksObjectSplinesOnSurfaces	11156	Коллекция сплайнов по поверхностям
ksObjectProjectionCurve	11157	Проекционная кривая
ksObjectProjectionCurves	11158	Коллекция проекционных кривых
ksObjectCurveBy2Projections	11159	Проекционная кривая
ksObjectCurvesBy2Projectionses	11160	Коллекция проекционных кривых
ksObjectContour3D	11161	Контур 3D
ksObjectContours3D	11162	Коллекция контуров 3D
ksObjectLineSegment3D	11163	Отрезок 3D
ksObjectLineSegments3D	11164	Коллекция отрезков 3D
ksObjectConicSpiral3D	11179	Коническая спираль
ksObjectCylindricSpiral3D	11180	Цилиндрическая спираль
ksObjectSpirals3D	11181	Коллекция спиралей
ksObjectPointsArrOnCurve	11182	Группа точек по кривой
ksObjectPointsArrsOnCurves	11183	Коллекция групп точек по кривым
ksObjectPointsArrOnSurfaces	11184	Группа точек по поверхности
ksObjectPointsArrsOnSurfaces	11185	Коллекция групп точек по поверхностям
ksObjectPointsArrFromFile	11186	Группа точек из файла
ksObjectPointsArrsFromFiles	11187	Коллекция групп точек из файлов
ksObjectAxis3D	11188	Вспомогательная ось 3D
ksObjectAxes3D	11189	Коллекция вспомогательных осей 3D
ksObjectAxis3DBy2Points	11190	Ось через две вершины
ksObjectAxis3DBy2Planes	11191	Ось на пересечении плоскостей
ksObjectAxis3DByConeface	11192	Ось конической поверхности
ksObjectAxis3DByEdge	11193	Ось через ребро
ksObjectAxis3DByPointAndObject	11194	Ось через вершину по объекту
ksObjectAxis3DByOperation	11195	Ось операции
ksObjectPlanes3D	11196	Коллекция плоскостей 3D
ksObjectPlane3D	11197	Плоскость 3D
ksObjectPlane3DByPlaneCurve	11198	Плоскость через плоскую кривую
ksObjectPlane3DTangentToFaceInPoint	11199	Плоскость касательная к грани в точке 3D
ksObjectPlane3DByOffset	11200	Смещенная плоскость
ksObjectPlane3DBy3Points	11201	Плоскость, проходящая через три вершины
ksObjectPlane3DByAngle	11202	Плоскость, под углом к другой плоскости
ksObjectPlane3DByEdgeAndPoint	11203	Плоскость через ребро и вершину
ksObjectPlane3DParallelByPoint	11204	Плоскость через вершину параллельно другой плоскости
ksObjectPlane3DPerpendicularByEdge	11205	Плоскость, проходящая через вершину перпендикулярно ребру
ksObjectPlane3DNormalToSurface	11206	Нормальная плоскость
ksObjectPlane3DMiddle	11207	Средняя плоскость 3D
ksObjectPlane3DByEdgeAndPlane	11208	Плоскость через ребро параллельно / перпендикулярно грани 3D

ksObjectPlane3DBy2Edge	11209	Плоскость через ребро параллельно / перпендикулярно другому ребру 3D
ksObjectPlane3DTangentToFace	11210	Плоскость касательная к грани 3D
ksObjectUserObject3D	11211	Пользовательский объект 3D
ksObjectUserObjects3D	11212	Коллекция пользовательских объектов 3D
ksObjectLinearPatternAnyCopy	11213	Копирование произвольных объектов по сетке
ksObjectCircularPatternAnyCopy	11214	Копирование произвольных объектов по окружности
ksObjectPathPatternAnyCopy	11215	Копирование произвольных объектов по кривой
ksObjectPointDrivenPatternAnyCopy	11217	Копирование произвольных объектов по точкам
ksObjectTablePatternAnyCopy	11218	Копирование произвольных объектов по таблице
ksObjectLinearUnhistoriedDimension	11219	Импортированный линейный размер
ksObjectAngularUnhistoriedDimension	11220	Импортированный угловой размер
ksObjectRadialUnhistoriedDimension	11221	Импортированный радиальный размер
ksObjectDiametralUnhistoriedDimension	11222	Импортированный диаметральный размер
ksObjectPlacement3D	11223	Интерфейс локальной системы координат (положение объекта)
ksObjectLayers3D	11224	Коллекция слоев в 3D
ksObjectLayer3D	11225	Слой в 3D
ksObjectLayerGroups3D	11226	Коллекция групп слоев в 3D
ksObjectLayerGroup3D	11227	Группа слоев в 3D
ksObjectDocument3DManager	11228	Менеджер 3D документа
ksObjectToleranceRecalc	11229	Пересчет модели
ksObjectSpecRough3D	11230	Неуказанная шероховатость 3D
ksObjectSketchBreakLinearDimension	11231	Управляющий линейный размер эскиза 3D с обрывом
ksObjectMathCurve3D	11232	Математическая кривая в трехмерном пространстве
ksObjectMathSurface3D	11233	Математическая поверхность в трехмерном пространстве
ksObjectBilletObsolete	11234	Деталь заготовка и Зеркальная деталь
ksObjectBilletsObsoletes	11235	Коллекция деталей заготовок и зеркальных деталей
ksObjectCopyGeometry	11236	Копия геометрии
ksObjectCopiesGeometry	11237	Коллекция копий геометрии
ksObjectCollectionGeometry	11238	Коллекция геометрии
ksObjectCollectionsGeometry	11239	Коллекции геометрии
ksObjectUserWireFrame3D	11240	Пользовательский объект Каркас 3D
ksObjectThreads	11241	Коллекция условных обозначений резьбы
ksObjectThread	11242	Условное обозначение резьбы
ks3dMateConstraints	11243	Коллекция 3D сопряжений
ksMate3DByAngle	11244	Сопряжение под углом
ksMate3DByTangent	11245	Сопряжение по касательной
ksMate3DSymmetry	11246	Сопряжение симметрия
ksMate3DTransmission	11247	Сопряжение механическое перемещение
ksMate3DCamGear	11248	Сопряжение кулачок-толкатель

ksMate3DDependentPosition	11249	Сопряжение зависимое положение
ksObjectHoles3D	11250	Коллекции отверстий 3D
ksObjectHole3D	11251	Отверстие 3D
ksObjectCountersinkHoleParameters	11252	Параметры отверстия с зенковкой
ksObjectSpotfacingHoleParameters	11253	Параметры отверстия с цековкой
ksObjectCountersinkSpotfacingHolePara meters	11254	Параметры отверстия с зенковкой и цековкой
ksObjectConicHoleParameters	11255	Параметры конического отверстия
ksObjectChamfers	11256	Коллекции фасок
ksObjectChamfer	11257	Фаска
ksObjectFillets	11258	Коллекции скруглений
ksObjectFillet	11259	Скругление
ksObjectInclines	11260	Коллекции операций уклон
ksObjectIncline	11261	Операция Уклон
ksObjectRibs	11262	Коллекции операций Ребро жесткости
ksObjectRib	11263	Ребро жесткости
ksObjectShells	11264	Коллекции операций Оболочка
ksObjectShell	11265	Оболочка
ksObjectBooleans	11266	Коллекция булевых операций
ksObjectBoolean	11267	Булева операция
ksObjectCuts	11268	Коллекция операций 'сечение'
ksObjectCut	11269	Операция 'сечение'
ksObjectLofts	11270	Коллекция операций по сечениям
ksObjectLoft	11271	Операция по сечениям
ksObjectLoftSurfaces	11272	Коллекция поверхностей по сечениям
ksObjectLoftSurface	11273	Операция поверхность по сечениям
ksObjectCoupling	11274	Цепочка в операции по сечениям
ksObjectEvolutions	11275	Коллекция кинематических операций
ksObjectEvolution	11276	Кинематическая операция
ksObjectEvolutionSurfaces	11277	Коллекция кинематических поверхностей
ksObjectEvolutionSurface	11278	Кинематическая поверхность
ksObjectVertex	11279	Вершина
ksObjectEdge	11280	Ребро
ksObjectFace	11281	Грань
ksObjectLoop7	11282	Цикл
ksObjectOrientedEdge7	11283	Ориентированное ребро
ksObjectUnionsComponents	11284	Коллекция операций объединение компонентов
ksObjectUnionComponents	11285	Операция объединения компонентов
ksObjectMoldCavities	11286	Коллекция операций вычитания компонентов
ksObjectMoldCavity	11287	Операция вычитания компонентов
ksObjectMacroObjects3D	11288	Коллекция макроэлементов 3D
ksObjectMacroObject3D	11289	3D макроэлемент
ksObjectNurbsSurfacesByCurvesMeshs	11290	Коллекция поверхностей по сети кривых
ksObjectNurbsSurfaceByCurvesMesh	11291	Поверхность по сети кривых
ksObjectJointSurfaces	11292	Коллекция поверхностей соединения
ksObjectJointSurface	11293	Поверхность соединения
ksObjectBodyRepositions	11294	Коллекция операций перепозиционирования тела, поверхности
ksObjectBodyReposition	11295	Перепозиционирование тела, поверхности
ksObjectDistanceAngleMeasurements3 D	11296	Коллекция измерений расстояния и угла

ksObjectDistanceAngleMeasurement3D	11297	Измерение расстояния и угла
ksObjectEdgeLengthMeasurements3D	11298	Коллекция измерений длины ребра
ksObjectEdgeLengthMeasurement3D	11299	Измерение длины ребра
ksObjectAreaMeasurements3D	11300	Коллекция измерений площади
ksObjectAreaMeasurement3D	11301	Измерение площади
ksObjectSheetMetalSketchBends	11302	Коллекция операций сгиб по эскизу
ksObjectSheetMetalSketchBend	11303	Сгиб по эскизу
ksObjectSheetMetalClosedCorners	11304	Коллекция операций Замыкание углов
ksObjectSheetMetalClosedCorner	11305	Замыкание углов
ksObjectSheetMetalPlates	11306	Коллекция операций Пластина
ksObjectSheetMetalPlate	11307	Пластина
ksObjectSheetMetalUndercuts	11308	Коллекция операций Подсечка
ksObjectSheetMetalUndercut	11309	Подсечка
ksObjectSheetMetalBendedStraightens	11310	Коллекция операций Согнуть/Разогнуть
ksObjectSheetMetalBendedStraighten	11311	Согнуть/Разогнуть
ksObjectSheetMetalPressFormings	11312	Коллекция операций открытая или закрытая штамповка
ksObjectSheetMetalPressForming	11313	Открытая или закрытая штамповка
ksObjectSheetMetalShoulders	11314	Коллекция операций буртик
ksObjectSheetMetalShoulder	11315	Буртик
ksObjectSheetMetalJalousies	11316	Коллекция операций жалюзи
ksObjectSheetMetalJalousie	11317	Жалюзи
ksObjectSheetMetalRibs	11318	Коллекция операций ребро усиления
ksObjectSheetMetalRib	11319	Ребро усиления
ksObjectSheetMetalRuledShells	11320	Коллекция операций Обечайка
ksObjectSheetMetalRuledShell	11321	Обечайка
ksObjectSheetMetalLinearRuledShells	11322	Коллекция операций Линейчатая обечайка
ksObjectSheetMetalLinearRuledShell	11323	Линейчатая обечайка
ksObjectLibraryHoleParameters	11324	Параметры отверстия из библиотеки
ksObjectFullFillets	11325	Коллекция операций полного скругления
ksObjectFullFillet	11326	Полное скругление
ksObjectZone	11327	Зона
ksObjectZonesManager	11328	Менеджер зон
ksObjectZoneParametersByObjects	11329	Параметры зоны по суммарному габариту объектов
ksObjectZoneParametersByBorderPoints	11330	Параметры зоны, заданной габаритным параллелепипедом
ksObjectZoneDivision	11331	Разбить зону
ksObjectZoneDivisionParametersByPlanes	11332	Параметры разбиения зоны для способа По набору плоскостей
ksObjectZoneDivisionParametersRegular	11333	Параметры разбиения зоны для способа Равномерно по осям
ksObjectDynamicCrossSection	11334	Динамическое сечение
ksObjectDynamicCrossSectionStep	11335	Шаг динамического сечения
ksObjectDynamicCrossSectionStepParametersByFreePlane	11336	Параметры шага динамического сечения по произвольной плоскости
ksObjectDynamicCrossSectionStepParametersByOffsetPlane	11337	Параметры шага динамического сечения по смещенной плоскости
ksObjectDynamicCrossSectionStepParametersByRotatedPlane	11338	Параметры шага динамического сечения по плоскости под углом
ksObjectDynamicCrossSectionStepParametersByZone	11339	Параметры шага динамического сечения по зоне

ksObjectDynamicCrossSectionStepParametersByBorderPoints	11340	Параметры шага динамического сечения по габаритному параллелепипеду заданному точками
ksObjectAxisLine3D	11341	Осевая линия
ksObjectRestoredSurfaces	11342	Коллекция операций восстановленная поверхность
ksObjectRestoredSurface	11343	Восстановленная поверхность
ksObjectDrawingObject	13000	Графический объект
ksObjectDrawingText	13001	Текст на чертеже
ksObjectDrawingTexts	13002	Коллекция текстов на чертеже
ksObjectStraightAxis	13003	Прямая ось
ksObjectCircleAxis	13004	Круговая ось
ksObjectArcAxis	13005	Дуговая ось
ksObjectBuildingAxes	13006	Коллекция строительных осей
ksObjectAxisJut	13007	Выступ оси
ksObjectMarkNodes	13008	Коллекция узлов для вставки марки
ksObjectMarkNode	13009	Узел для вставки дополнительных марок
ksObjectMarkOnLeader	13010	Марка/позиционное обозначение с линией-выноской
ksObjectMarkOnLine	13011	Марка/позиционное обозначение на линии
ksObjectMarkInsideForm	13012	Марка/позиционное обозначение без линии-выноски
ksObjectMarks	13013	Коллекция марок
ksObjectCutUnitMarking	13014	Обозначение узла в сечении
ksObjectCutUnitMarkings	13015	Коллекция обозначений узла в сечении
ksObjectUnitMarking	13016	Обозначение узла
ksObjectUnitMarkings	13017	Коллекция обозначений узлов
ksObjectUnitNumber	13018	Номер узла
ksObjectUnitNumbers	13019	Коллекция номеров узлов
ksObjectMultiTextLeader	13020	Выносная надпись к многослойным конструкциям
ksObjectMultiTextLeaders	13021	Коллекция выносных надписей к многослойным конструкциям
ksObjectBrace	13022	Фигурная скобка
ksObjectBraces	13023	Коллекция фигурных скобок
ksObjectLineSegments	13024	Коллекция отрезков
ksObjectLineSegment	13025	Отрезок
ksObjectRadialDimension	13028	Радиальный размер
ksObjectDiametralDimension	13029	Диаметральный размер
ksObjectBreakRadialDimension	13030	Радиальный размер с изломом
ksObjectRadialDimensions	13031	Коллекция радиальных размеров
ksObjectBreakRadialDimensions	13032	Коллекция радиальных размеров с изломом
ksObjectDiametralDimensions	13033	Коллекция диаметральных размеров
ksObjectLineDimension	13041	Линейный размер
ksObjectLineDimensions	13042	Коллекция линейных размеров
ksObjectBreakLineDimension	13043	Линейный размер с обрывом
ksObjectBreakLineDimensions	13044	Коллекция линейных размеров с обрывом
ksObjectHeightDimension	13045	Размер высоты
ksObjectHeightDimensions	13046	Коллекция размеров высоты
ksObjectAngleDimension	13047	Угловой размер
ksObjectBreakAngleDimension	13048	Угловой размер с обрывом

ksObjectAngleDimensions	13049	Коллекция угловых размеров
ksObjectArcDimension	13050	Размер дуги окружности
ksObjectArcDimensions	13051	Коллекция размеров дуг окружностей
ksObjectLeader	13052	Простая линия выноски
ksObjectLeaders	13053	Коллекция линий выносок
ksObjectRough	13054	Обозначение шероховатости
ksObjectRoughs	13055	Коллекция обозначений шероховатости
ksObjectMarkLeader	13056	Знак маркировки
ksObjectBrandLeader	13057	Знак клеймения
ksObjectPositionLeader	13058	Линия выноски для обозначения позиции
ksObjectChangeLeader	13059	Обозначение изменения
ksObjectBase	13060	Обозначение базы
ksObjectBases	13061	Коллекция обозначений базы
ksObjectDrawingTable	13062	Таблица на чертеже
ksObjectDrawingTables	13063	Коллекция таблиц на чертеже
ksObjectTolerance	13064	Допуск формы
ksObjectTolerances	13065	Коллекция допусков формы
ksObjectCutLines	13066	Коллекция линий разреза/сечения
ksObjectCutLine	13067	Линия разреза/сечения
ksObjectViewPointer	13068	Стрелка взгляда
ksObjectViewPointers	13069	Коллекция стрелок взгляда
ksObjectDrawingContour	13072	Контур на чертеже
ksObjectDrawingContours	13073	Коллекция контуров на чертеже
ksObjectCircles	13074	Коллекция окружностей
ksObjectCircle	13075	Окружность
ksObjectPoints	13076	Коллекция точек
ksObjectPoint	13077	Точка
ksObjectBeziers	13078	Коллекция Bezier сплайнов
ksObjectBezier	13079	Bezier сплайн
ksObjectMacroObjects	13080	Коллекция макроэлементов
ksObjectMacroObject	13081	Макроэлемент
ksObjectLines	13082	Коллекция линий
ksObjectLine	13083	Линия
ksObjectPolyLines2D	13084	Коллекция полилиний
ksObjectPolyLine2D	13085	Полилиния
ksObjectNurbses	13086	Коллекция Nurbs-сплайнов
ksObjectNurbs	13087	Nurbs-сплайн
ksObjectRasters	13088	Коллекция растровых объектов
ksObjectRaster	13089	Растровый объект
ksObjectOleDrawingObjects	13090	Коллекция OLE-объектов
ksObjectOleDrawingObject	13091	OLE-объект
ksObjectEllipses	13092	Коллекция эллипсов
ksObjectEllipse	13093	Эллипс
ksObjectEllipseArcs	13094	Коллекция дуг эллипсов
ksObjectEllipseArc	13095	Дуга эллипса
ksObjectRectangles	13096	Коллекция прямоугольников
ksObjectRectangle	13097	Прямоугольник
ksObjectRegularPolygons	13098	Коллекция многоугольников
ksObjectRegularPolygon	13099	Многоугольник
ksObjectEquidistants	13100	Коллекция эквидистант
ksObjectEquidistant	13101	Эквидистанта
ksObjectInsertionFragment	13102	Вставка фрагмента
ksObjectInsertionView	13103	Вставка вида из другого чертежа
ksObjectInsertionObjects	13104	Коллекция вставок фрагментов и видов
ksObjectInsertionDefinition	13105	Описание вставки фрагмента и вида
ksObjectCentreMarkers	13106	Коллекция обозначений центров

ksObjectCentreMarker	13107	Обозначение центра
ksObjectRemoteElements	13108	Коллекция выносных элементов
ksObjectRemoteElement	13109	Выносной элемент
ksObjectAxisLines	13110	Коллекция осевых линий
ksObjectAxisLine	13111	Осевая линия
ksObjectHatchParam	13112	Параметры штриховки
ksObjectDrawingGroup	13113	Группа объектов
ksObjectDrawingGroups	13114	Коллекция групп объектов
ksObjectCurve2D	13115	Математическая 2D кривая
ksObjectHatches	13116	Коллекция штриховок
ksObjectHatch	13117	Штриховка
ksObjectColourings	13118	Коллекция заливок
ksObjectColouring	13119	Заливка
ksObjectSpecRough	13120	Неуказанная шероховатость
ksObjectTechnicalDemand	13121	Технические требования
ksObjectAnnotativeLineSegment	13122	Аннотационный отрезок
ksObjectAnnotativeCircle	13123	Аннотационная окружность
ksObjectAnnotativeEllipse	13124	Аннотационный эллипс
ksObjectAnnotativeArc	13125	Аннотационная дуга
ksObjectAnnotativeEllipseArc	13126	Аннотационная дуга эллипса
ksObjectAnnotativePolyLine	13127	Аннотационная полилиния
ksObjectAnnotativePoint	13128	Аннотационная точка
ksObjectAnnotativeText	13129	Текст с аннотационной точкой привязки
ksObjectBuildingCutLines	13131	Коллекция линий разреза/сечения для СПДС
ksObjectBuildingCutLine	13132	Линия разреза/сечения для СПДС
ksObjectWaveLines	13133	Коллекция волнистых линий
ksObjectWaveLine	13134	Волнистая линия
ksObjectBrokenLines	13135	Коллекция линий обрыва с изломами
ksObjectBrokenLine	13136	Линия обрыва с изломами
ksObjectCopyObjectParam	13137	Параметры копирования объектов
ksObjectCurveCopyObjectParam	13138	Параметры копирования объектов вдоль кривой
ksObjectCircleCopyObjectParam	13139	Параметры копирования объектов по окружности
ksObjectCircularCopyObjectParam	13140	Параметры копирования объектов по концентрической сетке
ksObjectMeshCopyObjectParam	13141	Параметры копирования объектов по сетке
ksObjectLocalCoordinateSystem2D	13142	Локальная система координат
ksObjectLocalCoordinateSystems2D	13143	Коллекция локальных систем координат
ksObjectAttachedLeader	13144	Присоединенная линия выноски (не имеет текстов)
ksObjectAttachedLeaders	13145	Коллекция присоединенных линий выносок
ksObjectLoadCombinationsParam	13146	Параметры типа загрузки документа
ksObjectOpenDocumentParam	13147	Параметры открытия документа
ksObjectAssociationTables	13148	Коллекция ассоциативных таблиц
ksObjectAssociationTable	13149	Ассоциативная таблица
ksObjectNurbsesByPoints	13150	Коллекция NURBS-кривых по точкам
ksObjectNurbsByPoints	13151	NURBS-кривая по точкам
ksObjectProcess2D	13152	Процесс 2D из библиотеки
ksObjectPhantom2D	13153	Параметры фантома 2D
ksObjectProcess3D	13154	Процесс 3D из библиотеки

ksObjectUserDesignationCompObj	13155	Составной пользовательский объект обозначение 3D
ksObjectUserFolder	13156	Пользовательская директория
ksObjectUserFolders	13157	Пользовательские директории
ksObjectMeshObject3D	13158	Пространственный полигональный объект
ksObjectConicCurves	13159	Коллекция конический кривых
ksObjectConicCurve	13160	Коническая кривая
ksObjectCircularsCentries	13161	Коллекция круговых сеток центров
ksObjectCircularCentres	13162	Круговая сетка центров
ksObjectLinearsCentries	13163	Коллекция линейных сеток центров
ksObjectLinearCentres	13164	Линейная сетка центров
ksObjectFindObjectParameters	13165	Параметры поиска объектов
ksObjectManipulators	13166	Коллекция манипуляторов
ksObjectPlacement3DManipulator	13167	Манипулятор системы координат
ksObjectEditDoubleManipulator	13168	Манипулятор в виде редактора числового значения
ksObjectMouseEnterLeaveParameters	13169	Параметры отображения точки, позволяющей определить место применения контрола
ksObjectConditionIntersect	13170	Условное пересечение
ksObjectConditionIntersects	13171	Коллекция условных пересечений

ksAccuracyEnum – Количество знаков после запятой

ksAccuracyDefault	-1	По умолчанию. Использовать настройки
ksAccuracy0	0	0.
ksAccuracy1	1	0,1.
ksAccuracy2	2	0,12.
ksAccuracy3	3	0,123.
ksAccuracy4	4	0,1234.
ksAccuracy5	5	0,12345.
ksAccuracy6	6	0,123456.
ksAccuracy7	7	0,1234567.
ksAccuracy8	8	0,12345678.
ksAccuracy9	9	0,123456789.

ksAlignEnum – Выравнивание

ksAlignLeft	0	Выравнивание слева
ksAlignCenter	1	Выравнивание по центру
ksAlignRight	2	Выравнивание справа
ksAlignAllWidth	3	Выравнивание на всю ширину

ksAlignDecimal	3	Выравнивание по десятичной точке
ksAlignDefault	-1	По умолчанию (из стиля)

ksAlignmentTypeEnum – Тип ориентации объекта

ksATArbitrary	0	Произвольная
ksATHorizontal	1	Горизонтальная
ksATVertical	2	Вертикальная

ksAllocationEnum – Размещение текста относительно точки привязки

ksAlLeft	0	Слева
ksAlCentre	1	По центру
ksAlRight	2	Справа

ksAngleDimTypeEnum – Тип углового размера

ksADMinAngle	0	На минимальный (острый) угол
ksADMaxAngle	1	На максимальный (тупой) угол
ksADMoreAngle	2	На угол более 180 гр

ksAngleEnum – Углы поворота, кратные 90 градусам

ksAngle0	0	0 градусов
ksAngle90	1	90 градусов
ksAngle180	2	180 градусов
ksAngle270	3	270 градусов

ksAnnotationSymbolEnum – Аннотационные символы

ksUnknownSymbol	0	Символ не определен
ksDotPoint	1	Точка
ksPlusPoint	2	Крестик
ksXPoint	3	X-точка
ksSquarePoint	4	Квадрат
ksTrianglePoint	5	Треугольник

ksCirclePoint	6	Окружность
ksAsteriskPoint	7	Звезда
ksStrikeSquarePoint	8	Перечеркнутый квадрат
ksPlusPointTwo	9	Утолщенный плюс

ksAnnotativeTerminatorSignEnum – Типы специальных символов для аннотационных объектов

ksASUnknown	0	Не задан
ksASArrowInside	101	Стрелка (ласточкин хвост) изнутри
ksASArrowOutside	102	Стрелка (ласточкин хвост) снаружи
ksASNotchTail	103	Засечка с продолжением кривой (с хвостиком)
ksASUpHalfArrow	104	Верхняя половина стрелки изнутри
ksASDownHalfArrow	105	Нижняя половина стрелки изнутри
ksASBigArrowInside	106	Большая стрелка изнутри (7мм)
ksASArrowOrdinate	107	Стрелка для размера высоты (штрихи длиной 4мм под углом 45гр)
ksASTriangle	108	Треугольник по направлению кривой
ksAScircleRad2	109	Окружность радиусом 2мм тонкой линией - для шероховатости и линии-выноски
ksASCentreMarker	110	Обозначение фиктивного центра в виде большого креста
ksASGlueSign	111	Знак склеивания
ksASSolderingSign	112	Знак пайки
ksASSewingSign	113	Знак сшивания
ksASCrampSign	114	Знак соединения внахлестку металлическими скобами
ksASCornerCrampSign	115	Знак углового соединения металлическими скобами
ksASMontageJointSign	116	Знак монтажного шва
ksASNotch	117	Засечка без продолжения кривой (без хвостика)
ksASBaseTriangle	118	Треугольник по текущей СК - для базы
ksASClosedArrowInside	119	Закрытая стрелка изнутри
ksASClosedArrowOutside	120	Закрытая стрелка снаружи
ksASOpenArrowInside	121	Открытая стрелка изнутри
ksASOpenArrowOutside	122	Открытая стрелка снаружи
ksASRightAngleInside	123	Стрелка 90гр изнутри
ksASRightAngleOutside	124	Стрелка 90гр снаружи
ksASDot	125	Точка (диаметр равен длине стрелки размера)
ksASSmallDot	126	Точка маленькая (диаметр равен 0,6 длины стрелки размера)
ksASPoint	127	Вспомогательная точка
ksASLeftNotch	128	Засечка с наклоном влево

ksAPITypeEnum – Тип API

ksAPIUndef	0	Интерфейс неопределённого типа

ksAPI5Auto	1	API5 - интерфейсы автоматизации
ksAPI7Dual	2	API7 - дуальные интерфейсы
ksAPI3DCom	3	API 3D COM - интерфейсы

ksArc3DBuildingTypeEnum – Способ создания 3D дуги

ksArc3DByPoints	0	По трем точкам
ksArc3DByCentre	1	По центру, углам и плоскости
ksArc3DByDirrection	2	По двум точкам и направлению
ksArc3DByTanCurve	3	Касательно к кривой

ksArc3DParameterEnum – Индекс параметра 3D дуги

ksArc3DCenter	0	Точка центра
ksArc3DPoint1	1	Точка 1
ksArc3DPoint2	2	Точка 2
ksArc3DPoint3	3	Точка 3
ksArc3DAngle1	1	Угол 1
ksArc3DAngle2	2	Угол 2
ksArc3DRadius	3	Радиус

ksArchMeasureEnum – Способ задания глубины прогиба

ksArchMeasureByCoefficient	0	Глубина прогиба задана коэффициентом прогиба, в %
ksArchMeasureByLength	1	Глубина прогиба задана расстоянием

ksArrowEnum – Тип стрелки линии-выноски

Тип стрелки	Константа		Объекты со стрелками				
			Размеры	Линии-выноски	Обозначения позиций	Многослойные линии-выноски	Марки и позиционные обозначения
	ksLeaderWithoutArrow	0	+	+	+	+	+
Вспомогательная точка	ksLeaderPoint	1	+	+	+	+	+

Стрелка	ksLeaderArrow	2	+	+	+	+	+
Без стрелки	ksWithoutArrow	0	+	+	+	+	+
Вспомогательная точка	ksPoint	1	+	+	+	+	+
Стрелка	ksArrow	2	+	+	+	+	+
Верхняя половина стрелки изнутри	ksUpHalfArrow	3	+	+	-	+	+
Нижняя половина стрелки изнутри	ksDownHalfArrow	4	+	+	-	+	+
Засечка	ksNotch	5	+	+	-	+	+
Засечка с наклоном влево	ksLeftNotch	6	+	+	-	+	+
Угол 90 град	ksRightAngle	7	+	+	+	+	+
Стрелка закрытая	ksClosedArrow	8	+	+	+	+	+
Стрелка открытая	ksOpenArrow	9	+	+	+	+	+
Точка	ksDot	10	+	+	+	+	+
Точка маленькая	ksSmallDot	11	+	+	+	+	+
Треугольник 60 град	ksTriangle60	12	-	+	-	+	+
Треугольник 90 град	ksTriangle90	13	-	+	-	+	+

ksAttributeTypeEnum – Тип данных для типа атрибута

ksATUnknown	-1	Неизвестный
ksATString	0	Строка
ksATDouble	1	Число
ksATFixedTable	2	Таблица с фиксированным числом строк
ksATVariableTable	3	Таблица с переменным числом строк

ksBasisVectorTypeEnum – Типы базисного вектора

ksTangentVector	0	Касательный вектор
ksNormalVector	1	Вектор главной нормали
ksBinormalVector	2	Вектор бинормали

ksBendAngleReleaseTypeEnum – Способ освобождения угла сгиба

ksBendAngleBendOnly	0	Только сгиб
ksBendAngleIn	1	Сгиб и продолжение
ksBendAngleAllBends	2	Все сгибы

ksBendDisposalEnum – Тип размещения сгиба на ребре

ksBendDisposalAllLength	0	По всей длине
ksBendDisposalCentre	1	По центру
ksBendDisposalLeft	2	Отступ слева
ksBendDisposalRight	3	Отступ справа
ksBendDisposalTwo	4	Два отступа
ksBendDisposalLeftAndWidth	5	Отступ слева и по ширине
ksBendDisposalRightAndWidth	6	Отступ справа и по ширине

ksBendLengthTypeEnum – Тип определения длины

ksBendLengthByContinue	0	По продолжению
ksBendLengthByContour	1	По контуру
ksBendLengthByTouch	2	По касанию
ksBendLengthByContourInternal	3	По контуру внутри
ksBendLengthByTangentInternal	4	По касанию внутри

ksBendOffsetTypeEnum – Тип смещения

ksBendOffsetIn	0	Смещение внутрь
ksBendOffsetOut	1	Смещение наружу
ksBendOffsetLineOutside	2	По внешней линии контура
ksBendOffsetLineInside	3	По внутренней линии контура
ksBendOffsetByTouch	4	По касанию к сгибу
ksBendByCentre	5	По центру

ksBendReleaseTypeEnum – Тип освобождения сгиба

ksBendReleaseByRect	0	Прямоугольное
ksBendReleaseByCircle	1	Скругленное

ksBendSideTypeEnum – Тип построения боковой стороны сгиба

ksBendSideByAngle	0	По уклону и углу
ksBendSideByWidening	1	По расширению

ksBendTypeEnum – Способ сгиба

ksLineBend	0	По линии сгиба
ksBendLineOutside	1	Линия сгиба снаружи
ksBendLineInside	2	Линия сгиба внутри
ksBendByTouch	3	По касанию
ksBendByCentre	4	По центру

ksBisectorVariant – Вариант решения биссектрисы для двух прямых

ksBVNone	0	Неопределенное направление (ближайшее решение)
ksBVNormalSum	1	Биссектриса вдоль суммы нормалей прямых/отрезков
ksBVNormalDiff	2	Биссектриса вдоль разности нормалей прямых/отрезков

ksBmpSizeEnum – Размеры иконок

ksBmp1616	0	16*16
ksBmp2424	1	24*24
ksBmp3232	2	32*32
ksBmp4848	3	48*48

ksBreakLineTypeEnum – Тип линии разрыва

ksBLNotImage	0	Не отображается
ksBLStraight	1	Прямая
ksBLBreak	2	С изломом
ksBLVawe	3	Волнистая

ksCellBoundariesEnum – Типы границ таблицы

ksCBLeftBorder	Левая граница
ksCBRightBorder	Правая граница
ksCBTopBorder	Верхняя граница
ksCBBottomBorder	Нижняя граница
Групповые границы	
ksCBHorisontMidleBorder	Внутренняя горизонтальная граница

ksCBVerticalMidleBorder		Внутренняя вертикальная граница
ksCBExternalBorders		Внешние границы
ksCBAllBorders		Все границы
ksCBExternalBorders		Внутренние границы
ksCBInternalBorders		Нет границ

ksCentreMarkerEnum – Тип обозначения центра

ksCMUnknown	-1	Неизвестный
ksCMPlusPoint	0	Маленький крестик
ksCMOneAxis	1	Одна ось
ksCMTwoAxis	2	Две оси

ksChamferBuildingTypeEnum – Типы построения фаски

ksChamferSideAngle	0	По стороне и углу
ksChamferTwoSides	1	По двум сторонам

ksChangeLeaderSignEnum – Тип значка для обозначения изменения

ksCLSSquare	0	Квадрат
ksCLSCircle	1	Окружность
ksCLSBracketSquare	2	Квадратные скобки
ksCLSBracketCircle	3	Круглые скобки
ksCLSBracketCorner	4	Угловые скобки

ksCheckBoxVisualStyleEnum – Визуальный стиль чекбокса

ksCheckBoxDefault	0	Обычный
ksCheckBoxSwitcher	1	Переключатель

ksCheckSumVersionEnum – Версии контрольных сумм

KsCsrCurrent	0	Версия текущей задачи КОМПАС
ksCsrKompas10SP2	-1	Версия KOMPAS 10 SP2
ksCsrKompas11	-5	Версия KOMPAS 11

ksCsrKompas11SP1	-6	Версия KOMPAS 11 SP1
ksCsrKompas12	-10	Версия KOMPAS 12
ksCsrKompas12SP1	-11	Версия KOMPAS 12 SP1
ksCsrKompas13	-15	Версия KOMPAS 13
ksCsrKompas13SP1	-16	Версия KOMPAS 13 SP1
ksCsrKompas14SP1	-21	Версия KOMPAS 14 SP1
ksCsrKompas14SP2	-22	Версия KOMPAS 14 SP2
ksCsrKompas15	-25	Версия KOMPAS 15
ksCsrKompas15Sp2	-27	Версия KOMPAS 15 SP2
ksCsrKompas16	-30	Версия KOMPAS 16
ksCsrKompas16Sp1	-31	Версия KOMPAS 16 SP1
ksCsrKompas17	-35	Версия KOMPAS 17
ksCsrKompas17Sp1	-36	Версия KOMPAS 17 SP1
ksCsrKompas18	-37	Версия KOMPAS 18

Примечание:

ksCsrKompas10SP2 = 0x0A001023

ksCsrKompas13SP1 - До выхода КОМПАС 13 SP1 равен версии текущей задачи.

ksChooseBodiesType – Типы действий над телами для операций

ksNewBody	0	Новое тело.
ksAutomaticDefinition	1	Автоматический выбор тел.
ksManualEditing	2	Ручное указание тел.
ksAllBodies	3	Все тела.

Примечание:

Для операций сечение плоскостью и сечение эскизом значение ksNewBody интерпретируется как ksAutomaticDefinition.

ksChooseManagerTypeEnum – Тип менеджера выбора объектов

ksChMUnknown	-2	Неизвестный
ksChMAllColors	-1	Любой менеджер
ksChMLevel1ColorBase	0	Основная группа выбора. Цвет - Указание 1
ksChMLevel1Color1	1	Дополнительная группа выбора. Цвет - Указание 2
ksChMLevel1Color2	2	Дополнительная группа выбора. Цвет - Указание 3
ksChMLevel2ColorBase	100	Основная группа выбора. Уровень 2. Цвет - Указание 1
ksChMLevel2Color1	101	Дополнительная группа выбора. Уровень 2. Цвет - Указание 2
ksChMLevel2Color2	102	Дополнительная группа выбора. Уровень 2. Цвет - Указание 3

ksChoosePartsType – Способ определения области применения для компонентов в сборочной операции

ksChAutomaticDefinition	1	Автоопределение
ksChManualEditing	2	Ручное редактирование
ksChAllParts	3	Все компоненты
ksChNoLibraryParts	4	Все компоненты, кроме библиотечных

ksChooseType – Область применения

ksChBodiesAndParts	1	Компоненты и тела
ksChParts	2	Компоненты
ksChBodies	3	Все компоненты

ksCircularPatternBuildingTypeEnum – Способ построения массива по концентрической сетке

ksCPSaveAll	0	Стандартная схема
ksCPChessOrderByAxis1	1	Шахматный порядок - сдвиг вдоль первой оси (концентрическое направление
ksCPChessOrderByAxis2	2	Шахматный порядок - сдвиг вдоль второй оси (радиальное направление)

ksCloudPointsSurfaceBuildingTypeEnum – Тип поверхности по пласту (облаку) точек

ksCLByPoints	0	Сплайновая поверхность по точкам
ksCLByPole	1	Сплайновая поверхность по полюсам
ksCLPolyhedral	2	Многогранная поверхность

ksCloudTypeEnum – Способ распознавания сети точек

ksCLAuto	0	Автоматически
ksCLLocalCS	1	В плоскости СК
ksCLScreen	2	В плоскости экрана

ksColouringTypeEnum – Тип заливки

ksColouringSolid	0	Одноцветная
ksColouringLinear	1	Линейная
ksColouringAngle	2	Угловая
ksColouringCone	3	Коническая
ksColouringCircle	4	Радиальная
ksColouringSquare	5	Квадратная
ksColouringCylinder	6	Цилиндрическая

ksConicCurvePontIndexEnum – Индекс точки конической кривой

ksCCBeginPoint	0	Начальная точка
ksCCEndPoint	1	Конечная точка
ksCCIntersectPoint	2	Точка пересечения
ksCCPointOnCurve	3	Точка на кривой

ksConjunctivePointTypeEnum – Способ построение присоединительной точки

ksCPByObject	0	По объекту
ksCPManualDirection	1	Ручное задание направление осей

ksConnectTypeEnum – Тип соединения кривых

ksCTUnknown	-1	Неизвестный
ksCTPosition	0	Соединение по позиции
ksCTTangent	1	Соединение по касательной
ksCTNormal	2	Соединение перпендикулярно
ksCTSmooth	3	Гладкое соединение

ksConstraintTypeEnum – Типы параметрических ограничений

ксCUnknown	0	Неизвестный тип
ksCFixedPoint	1	Фиксировать точку
ksCPointOnCurve	2	Точка на кривой
ksCHorizontal	3	Горизонталь
ksCVertical	4	Вертикаль
ksCParallel	5	Параллельность двух прямых или отрезков

ksCPerpendicular	6	Перпендикулярность двух прямых или отрезков
ksCEqualLength	7	Равенство длин двух отрезков
ksCEqualRadius	8	Равенство радиусов двух дуг/окружностей.
ksCHAlignPoints	9	Выравнивать две точки по горизонтали
ksCVAlignPoints	10	Выравнивать две точки по вертикали
ksCMergePoints	11	Совпадение двух точек
ksCAssociation	12	Ассоциативная связь
ksCDimWithVariable	13	Размер с переменной
ksCFixedDim	14	Фиксированный размер
ksCTangentTwoCurves	15	Касание двух кривых
ksCSymmetryTwoPoints	16	Симметрия двух точек относительно отрезка
ksCCollinear	17	Коллинеарность двух отрезков
ksCFixedAngle	18	Фиксированный угол
ksCFixedLenght	19	Фиксированная длина
ksCPointOnCurveMiddle	20	Точка на середине кривой
ksCBisector	21	Биссектриса
ksCConcentricity	22	Совпадение центров окружностей, дуг, эллипсов и точек

Примечание.

При использовании API 5 частичным аналогом данного перечисления является набор типов параметрических ограничений.

ksClosingClosedTypeEnum – Способ замыкания углов листового тела

ksClosingCJoint	0	Замыкание встык
ksClosingCOver	1	Замыкание с перекрытием
ksClosingCTightClosure	2	Плотное замыкание

ksClosingCorneringEnum – Обработка угла при замыкании

ksCorneringNone	0	Без обработки
ksCorneringJointOnBorder	1	Стык по кромке
ksCorneringJointOnChord	2	Стык по хорде
ksCorneringCircled	3	Круговая

ksClosingHolePlacementEnum – Размещение отверстия при круговой обработке угла

ksCorneringHPBend	0	На пересечении сгибов
ksCorneringHPAngle	1	В точке угла
ksCorneringHPPoint	2	Через точку угла

ksClosingTypeEnum – Тип замыкания операции сгиб по эскизу листового тела

ksClosingAngles	0	Замыкание смежных углов
ksClosingBegin	1	Замыкание в начале
ksClosingEnd	2	Замыкание в конце

ksCrossSectionPlaneBuildingTypeEnum – Способ построения секущей плоскости для шага динамического сечения

ksCrossSectionPlaneByModelPlan	0	По плоскости модели
ksCrossSectionPlaneByOtherStep	1	По плоскости сечения предыдущего шага

ksCurveStyleTypeEnum – Тип стиля кривой

ksCSTSoldLine	0	Сплошная
ksCSTBrokenLine	1	Прерывистая

ksCurvePenTypeEnum – Способ задания параметров пера

ksCPTIndependent	0	Задано значением
ksCPTBasicLine	1	Как у основной линии

ksCPTThinLine	2	Как у тонкой линии
ksCPTHeavyLine	3	Как у утолщенной линии

ksEndFaceTypeEnum – Форма торца отверстия

ksEFFlat	0	Плоский
ksEFConic	1	Конический
ksEFSphere	2	Сферический

ksHoleTypeEnum – Тип отверстия

ksHTBase	0	Простое
ksHTCounterbore	1	С цековкой
ksHTCountersinking	2	С зенковкой
ksHTCounterdrill	3	С цековкой и зенковкой
ksHTConic	4	Коническое.
ksHTLfrLibrary	5	Отверстие из библиотеки

ksCountersinkTypeEnum – Способ определения параметров зенковки

ksCTDiameterAngle	0	По диаметру и углу
ksCTDepthAngle	1	По глубине и углу
ksCTDiameterDepth	2	По диаметру и глубине

ksConicTypeEnum – Способ определения параметров конического отверстия

ksCNDiameter	0	По диаметру верхнего основания конуса
ksCNAngle	1	По углу конуса

ksContentDialogNotifyEnum – События диалога с произвольным наполнением

ksCDCreateContentCallback	1	Создание контента
ksCDDestroyContent	2	Удаление контента
ksCDExecuteCommand	3	Выполнить команду
ksCDButtonUpdate	4	Установка состояния кнопки панели

ksContourSegmentEnum – Типы сегментов контура

ksCSUnknown	-1	Неизвестный объект
ksCSLineSeg	1	Отрезок
ksCSCircle	2,	Окружность
ksCSArc	3	Дуга
ksCSBezier	8	Bezier сплайн
ksCSEllipse	32	Эллипс
ksCSNurbs	33	Nurbs сплайн
ksCSEllipseArc	34	Дуга эллипса

ksContour3DTypeEnum – Тип контура

ksCTAuto	0	Автоопределение типа
ksCTSpace	1	Произвольный контур
ksCTSurface	2	Контур на грани
ksCTSketch	3	Контур эскиза

ksCoordLawEnum – Порядок законов

ksCLawX	0	Закон X
ksCLawY	1	Закон Y
ksCLawZ	2	Закон Z

ksCopyGeometryBuildingTypeEnum – Способ построения операции копия геометрии

ksCGBTWithoutGrouping	0	Без группировки.

ksCGBTBodyFaceGrouping	1	Собрать грани по исходным элементам. С группировкой граней одного тела

ksCornerTypeEnum – Тип угла объекта для прямоугольника и многоугольника

ksCTNoProcess	0	Без обработки
ksCTChamfer	1	Фаска
ksCTFillet	2	Скругление

ksCurveProjectionTypeEnum – Тип проекции кривой

ksCPNearest	0	Ближайшая проекция
ksCPNearestByDirection	1	По направлению

ksCutBuildingTypeEnum – Способ создания сечения

ksCutByContour	0	Сечение контуром
ksCutByPlane	1	Сечение плоскостью

ksCurveStyleEnum – Системные стили линии

ksCSHidden	-1	Невидимая (скрытая; для кривых)
ksCSUnvisible	0	Невидимая (для таблицы)
ksCSNormal	1	- основная,
ksCSThin	2	- тонкая,
ksCSAxial	3	- осевая,
ksCSDashed	4	- штриховая,
ksCSBrokenLine	5	- для линии обрыва
ksCSConstruction	6	- вспомогательная,
ksCSThick	7	- утолщенная,
ksCSDash2Dots	8	- пунктир 2,
ksCSDashedNormal	9	- штриховая осн.
ksCSNormalDashDot	10	- осевая осн.
ksCSConstraints		для ограничений
ksCSDedreesOfFreedom		для степеней свобод

	11	- тонкая линия, включаемая в штриховку
ksCSISO02Dashed	12	- ISO 02 штриховая линия,
ksCSISO03DashedLSpace	13	- ISO 03 штриховая линия (дл. пробел),
ksCSISO04DashDotLDash	14	- ISO 04 штрихпунктирная линия (дл. штрих),
ksCSISO05DashDotLDash2Dots	15	- ISO 05 штрихпунктирная линия (дл. штрих 2 пунктира),
ksCSISO06DashDotLDash3Dots	16	- ISO 06 штрихпунктирная линия (дл. штрих 3 пунктира),
ksCSISO07Dotted	17	- ISO 07 пунктирная линия,
ksCSISO08DashDotLShDashes	18	- ISO 08 штрихпунктирная линия (дл. и кор. штрихи),
ksCSISO09DashDot1L2ShDashes	19	- ISO 09 штрихпунктирная линия (дл. и 2 кор. штриха),
ksCSISO10DashDot	20	- ISO 10 штрихпунктирная линия,
ksCSISO11DashDot2Dashes	21	- ISO 11 штрихпунктирная линия (2 штриха),
ksCSISO12DashDot2Dots	22	- ISO 12 штрихпунктирная линия (2 пунктира),
ksCSISO13DashDot3Dots	23	- ISO 13 штрихпунктирная линия (3 пунктира),
ksCSISO14DashDot2Dashes2Dots	24	- ISO 14 штрихпунктирная линия (2 штриха 2 пунктира),
ksCSISO15DashDot2Dashes3Dots	25	- ISO 15 штрихпунктирная линия (2 штриха 3 пунктира).

ksDepthTypeEnum – Способ определения глубины отверстия

ksDTValue	0	На расстояние
ksDTReachThrough	1	Через все
ksDTObject	2	До объекта

ksDimensionArrowPosEnum – Размещение стрелок относительно выносной линии

ksDimArrowInside	0	Стрелки изнутри
ksDimArrowOutside	1	Стрелки снаружи
ksDimArrowAuto	2	Авторазмещение стрелок

ksDimensionBaseEnum – Параметр отрисовки текста

ksDimBaseCenter	0	От центра
ksDimBaseP1	1	От первой выносной линии
ksDimBaseP2	2	От второй выносной линии
ksDimCommonBase	3	Размер с общей размерной линией (текст размера выводится вертикально у второй выносной линии)
ksDimFirstCommonBase	4	Первый размер в цепочке размерных линий (у первой выносной линии отрисовывается текст '0', текст размера выводится вертикально у второй выносной линии).

ksDimensionDeviationEnum – Отклонения номинального значения размера

ksDimDeviation	0	Отклонения
ksDimLimits	1	Предельные значения (предельные значения записываются одно над другим)
ksDimLineLimits	2	Предельные значения в одну строку (предельные значения записываются друг за другом через дефис)

ksDimensionTextAlignEnum – Выравнивание размерной надписи

ksDimACentreLowFont	0	По центру, с уменьшенным шрифтом
ksDimAUpperBoundary	1	По верхней границе
ksDimACentre	2	По центру
ksDimALowerBoundary	3	По нижней границе

ksDimensionTextBracketsEnum – Размер в скобках

ksDimBracketsOff	0	Размер без скобок
ksDimBrackets	1	Размер в круглых скобках
ksDimSquareBrackets	2	Размер в квадратных скобках

ksDimensionTextPosEnum – Положение размерной надписи относительно выносной линии

ksDimTextParallelOnLine	0	Параллельно, над линией
ksDimTextParallelInCut	1	Параллельно, в разрезе линии
ksDimTextHorizontalInCut	2	Горизонтально, в разрезе линии

ksDimensionTextTypeEnum – Тип размерной надписи

ksDimTAuto	0	Автоматическое
ksDimTManual	1	Ручное
ksDTPOnShelf	2	На полке

ksDimTextFormatEnum – Формат отображения размерной надписи

ksDimTextFormatGMS	0	Градусы-минуты-секунды
ksDimTextFormatGDD	1	Градусы и десятичные доли градуса

Примечание.

Используется только для углового размера.

ksDirectionTypeEnum – Типы направлений выдавливания

dtNormal	0	прямое направление (для тонкой стенки - наружу)
dtReverse	1	обратное направление (для тонкой стенки - внутрь)
dtBoth	2	в обе стороны
dtMiddlePlane	3	от средней плоскости

Примечание:

1. При типе направления dtMiddlePlane в методах SetSideParam и GetSideParam параметр depth интерпретируется как общая глубина выдавливания и задается следующим образом:

   SetSideParam(TRUE, etBlind, depth, ...)

2. В API5 соответствует Типам направлений выдавливания.

ksDocumentFormatEnum – Форматы листа

ksFormatA0	0	Формат А0
ksFormatA1	1	Формат А1
ksFormatA2	2	Формат А2
ksFormatA3	3	Формат А3
ksFormatA4	4	Формат А4
ksFormatA5	5	Формат А5
ksFormatUser	6	ksFormatUser

ksDocumentsLibraryInsertionTypeEnum – Типы документов в библиотеке документов КОМПАС

ksInsertionUnknown	0	Неизвестный тип
ksInsertionFragment	2	Фрагмент
ksInsertionPart	4	Деталь
ksInsertionAssembly	5	Сборка
ksInsertionTextual	6	Текстовый документ
ksInsertionRaster	20	Растр
ksInsertionTable	21	Таблица
ksInsertionTxtFile	22	Текстовый документ. *.txt

ksDrawingObjecParamTypeEnum – Тип параметров объекта

ksAllParam	-1	Параметры в системе координат владельца.
ksSheetAllParam	-2	Параметры в системе координат листа.
ksViewAllParam	-7	Параметры в системе координат вида.

Примечание:

При выдаче параметров в системе координат владельца для геометрических объектов учитывается также текущая суммарная матрица, задаваемая функцией Mtr.

Примерами параметров, зависящих от суммарной матрицы, являются координаты точек, углы, длины, радиусы и т.п. Она не распространяется на параметры аннотационных объектов, так как аннотационные объекты не масштабируются. Примерами параметров аннотационных объектов, не зависящих от текущей суммарной матрицы, являются высота шрифта, сужение.

ksDynamicCrossSectionStepBuildingTypeEnum – Способ создания шага сечения модели

ksDCSTUnknown	-1	Неопределенный
ksDCSTByFreePlane	0	Произвольная плоскость
ksDCSTByOffsetPlane	1	Смещенная плоскость
ksDCSTByRotatedPlane	2	Повернутая плоскость
ksDCSTByZone	3	Задано зоной
ksDCSTByBorderPoints	4	Параллелепипед

ksD3ConverterOptionsEnum – Константы управляющие разрешением на чтение или запись объектов в дополнительные форматы jgs, sat, xt, step, stl, VRML

ksD3COBodyes	0	Разрешение на чтение\запись твёрдых тел
ksD3COSurfaces	2	Разрешение на чтение\запись поверхностей
ksD3COCurves	4	Разрешение на чтение\запись кривых
ksD3COSketches	6	Разрешение на чтение (не применяется) \запись эскизов
ksD3COInvisibleObjects	8	Разрешение на чтение (не применяется) \запись невидимых объектов
ksD3COPoints	10	Разрешение на чтение\запись точек
ksD3CODocumentProperties	12	Разрешение на чтение\запись информации о документе (автор, организация, комментарии)
D3COTechnicalDemand	14	Разрешение на чтение\запись технических требований
ksD3CODimensions	16	Разрешение на чтение\запись размеров
ksD3COAttributes	18	Разрешение на чтение\запись атрибутов объектов
ksD3CBRep	20	Разрешение на чтение\запись форм изделий в граничном представлении (только в JT)
ksD3CPolygonal	22	Разрешение на чтение\запись полигональных форм изделий
ksD3CPolygonalLOD0	24	Разрешение на чтение\запись полигональных форм изделий уровня детализации 0
ksD3CAssociated	26	Разрешение на чтение ассоциированной геометрии (резьбы и др)
ksD3COStyle	28	Разрешение на чтение\запись элементов оформления (цвет, начертание, и т.п.)

ksEditableStateEnum – Способ редактирования

ksESUndefined	-1	Неопределенное состояние
ksESDisable	0	Не проецировать
ksESEnable	1	Проецировать
ksESByLayer	2	По слою

ksEditColorTypeEnum –Тип цвета редактирования

ksECSelectObject	0	Подсвечивание - Выделенный объект
ksECChooseObject1	1	Подсвечивание - Указанный объект 1
ksECChooseObject2	2	Подсвечивание - Указанный объект 2
ksECChooseObject3	3	Подсвечивание - Указанный объект 3
ksECPassiveParts	4	Контекстное редактирование - Пассивные компоненты
ksECFacePhantom	5	Фантом грани
ksECLabelsPhantom	6	Фантом надписи
ksECDimensions	7	Размеры

ksEditListCommandEnum – Идентификаторы стандартных команд для элемента панели свойств – Список

ksListItemNew	1	Создать новый элемент в списке
ksListItemDelete	2	Удалить выделенные элементы из списка
ksListItemMoveUp	3	Переместить вверх
ksListItemMoveDown	4	Переместить вниз
ksListItemEdit	5	Редактировать элемент в списке

ksEditListTypeEnum – Тип списка панели свойств

ksEditList	0	Обычный список
ksCheckList	1	Список флагов
ksRadioList	2	Список переключателей

ksEquidistantTypeEnum – Тип построения эквидистанты

ksETUnknown	-1	Неизвестный
ksETLeft	0	Слева по направлению
ksETRight	1	Справа по направлению
ksETBoth	2	С двух сторон

ksEquidistant3DCutModeEnum – Обход углов эквидистанты 3D

ksECMUnknown	0	Не определен
ksECMLineSeg	1	Обход срезом
ksECMCircle	2	Обход дугой

ksEndTypeEnum – типы операций выдавливания

etBlind	0	строго на глубину
etThroughAll	1	через всю деталь
etUpToVertexTo	2	на расстояние до вершины
etUpToVertexFrom	3	на расстояние за вершину
etUpToSurfaceTo	4	на расстояние до поверхности
etUpToSurfaceFrom	5	на расстояние за поверхность
etUpToNearSurface	6	до ближайшей поверхности

Примечание:

1. При типах выдавливания etUpToVertexTo, etUpToVertexFrom, etUpToSurfaceTo и etUpToSurfaceFrom в методах SetSideParam и GetSideParam параметр depth интерпретируется как глубина, вычитаемая или добавляемая к расстоянию до указанного объекта. Объект, определяющий глубину, задается с помощью метода SetDepthObject.
2. В API5 соответствует Типам операций выдавливания.

ksEvolutionShiftSketchTypeEnum – Тип движения сечения в кинематической операции

ksEvShiftParallel	0	Образующая переносится параллельно самой себе
ksEvShiftKeepAngle	1	Образующая при переносе сохраняет исходный угол с направляющей
ksEvShiftOrtogonal	2	Плоскость образующей выставляется и сохраняется ортогональной направляющей

ksExtensionLimitTypeEnum – Способ ограничения

ksETLUnknown	-1	Неизвестный
ksETLength	0	На заданную длину
ksETLVertex	1	До вершины

ksExtensionSurfaceTypeEnum – Тип продления поверхности

ksESTUnknown	-1	Неизвестный.
ksESTSelf	0	Той же поверхностью.
ksESTTangent	1	По касательной.
ksESTDirection	2	По направлению.

ksExternalFilesTypesEnum – Тип внешнего файла

ksEFTUnknown	-1	Неизвестный тип
ksEFTDocumentFile	0	Файл документа
ksEFTCurveStyleLibrary	1	Библиотека стилей кривых
ksEFTTextStyleLibrary	2	Библиотека стилей текстов
ksEFTHatchStyleLibrary	3	Библиотека стилей штриховок
ksEFTAttributeTypesLibrary	4	Библиотека типов атрибутов
ksEFTLayoutsLibrary	5	Библиотека оформлений
ksEFTFragmentFile	6	Фрагмент
ksEFTFragmentsLibrary	7	Библиотека фрагментов
ksEFTSheetConnectedToSpc	8	Лист сборки, подключенный к спецификации
ksEFTSystemFile	9	Системный файл
ksEFTIngotsLibrary	10	Библиотека типовых элементов (3D)
ksEFTPartFile	11	Деталь (3D)
ksEFTRasterFile	12	Растровое изображение
ksEFTDocConnectedToSpcObj	13	Документ, подключенный к объекту спецификации
ksEFTSpcConnectedToSheet	14	Спецификация, подключенная к листу сборки
ksEFTDocConnectedToSpcObj InOtherDocs	15	Документ, подключенный к объектам спецификации в других документах
ksEFTAssemblyFile	16	Файл-сборка (3D)
ksEFTModelsLibrary	17	Библиотека моделей
ksEFTTemporaryFile	18	Временный файл
ksEFTSourceFileForVariable	19	Файлы-источники для переменных
ksEFTServiceFile	20	Служебный файл
ksEFTDraftFile	21	Чертеж
ksEFTHyperLink	22	Гиперссылка
ksEFTDataFile	23	Файл данных
ksEFTReadingRegimFile	24	Файл режима чтения компонента
ksEFTCopyExternalGeometry	27	Файл-источник копии внешней геометрии
ksEFExternalBilletPart	28	Файлы деталей заготовок
ksEFTExternalLayoutGeometry	29	Файлы компоновочной геометрии
ksEFTMetaDataSource	30	Файл-источник метаданных
ksEFTExternalRefContextFile	32	Файл контекста внешней ссылки

ksFeatureStateEnum – Состояние объекта

ksFSNone	0	Состояния не определены
ksFSLocked	0x1	Объект заблокирован\Только чтение

ksFSRebuild	0x2	Необходимо перестроить модель
ksFSInside3dMacro	0x4	Объект включен в состав трехмерного макроэлемента
ksFSLCSDependent	0x8	Зависимость от ЛСК
ksFSComprisedOfParts	0x10	Тело состоит из частей
ksFSCurrent	0x20	Текущая СК
ksFSReadOnlyAccessM3d	0x40	Файл компонента (m3d) имеет доступ “Только чтение”
ksFSDetailedFold	0x80	Развернутый сгиб
ksFSSketchUndefined	0x100	Эскиз не определен
ksFSSketchDefined	0x200	Эскиз определен
ksFSSketchRedefined	0x400	Эскиз переопределен
ksFSEditRestricted	0x800	Редактирование запрещено, но запрет можно снять
ksFSEditImpossible	0x1000	Редактирование запрещено
ksFSFixedComponent	0x2000	Компонент зафиксирован
ksFSPutInRecalcState	0x4000	Вставка компонента в пересчитанном состоянии
ksFSReadOnlyAccessA3d	0x8000	Файл компонента (a3d) имеет доступ “Только чтение”
ksFSUncuttable	0x10000	Неразрезаемый компонент
ksFSUpdateNeeded	0x100000	Необходимость обновления
ksFSBrokenLink	0x200000	Разрыв связи копии геометрии с источником

ksFacetCullingMode – Режим фильтрации отображаемых граней внешнего объекта

ksFSMNone	0	Отображается всё (не фильтруется)
ksFSMFront	1	Отбрасывается передняя грань
ksFSMBack	2	Отбрасывается задняя грань
ksFSMAll	3	Ничего не отображается

ksFilletBuildingTypeEnum – Типы построения скругления

ksFilletCircleArc	0	Дугой окружности
ksFilletEllipseArc	1	Дугой эллипса
ksFilletCoefficient	2	С коэффициентом (0 < K < 1)
ksFilletHord	3	С постоянной хордой

ksFilletOffsetModeEnum – Способ расчета смещения для точек останова скругления

ksFilletOffsetByPersent	0	В процентах от длины кривой.
ksFilletOffsetByLength	1	По длине сегмента
ksFilletOffsetByAngle	2	По центральному углу дуги

ksFindObjectsTypeEnum – Тип поиска объектов

ksFindObjByType	-1	Объект заданного типа
ksFindAll	0	Любые объекты
ksFindAnyCurve	1	Любая кривая или прямая
ksFindCircleAnalog	4	Дуга или окружность
ksFindtTimmedCurve	8	Кривая, имеющая граничные точки
ksFindEllipseAnalog	13	Эллипс или дуга эллипса.
ksFindHatchBoundary	15	Кривые, подходящие для границы штриховки. (Дополнительно проверяется стиль кривых)

ksFindObjectParametersNotifyEnum – События функции поиска объектов

ksFOPFilterObjectr	1	Фильтрация объетов

ksGabaritBuildingTypeEnum – Способ задания габаритов

ksGabaritByBorderPoints	0	По двум вершинам
ksGabaritByCenterAndBorder	1	По центру и вершине

ksHatchStyleEnum – Системные стили штриховки

ksHatchMetal	0	Металл
ksHatchNonMetal	1	Не металл
ksHatchTimber	2	Дерево

ksHatchNaturalStone	3	Камень естественный
ksHatchCeramics	4	Керамика
ksHatchConcrete	5	Бетон
ksHatchGlass	6	Стекло
ksHatchLiquid	7	Жидкость
ksHatchNaturallyGround	8	Естественный грунт
ksHatchSpreadGround	9	Насыпной грунт
ksHatchArtificialStone	10	Камень искусственный
ksHatchReinforcedConcrete	11	Железобетон
ksHatchTenseReinforcedConcrete	12	Напряженный железобетон
ksHatchLongitudalTimber	13	Дерево в продольном сечении
ksHatchSand	14	Песок

ksHeightDimTypeEnum – Тип размеров высоты

ksHDFrontView	0	Для вида спереди или разреза, с полкой и стрелкой, возможна выносная линия.
ksHDTopView	1	Для вида сверху без линии-выноски
ksHDTopViewLeader	2	Для вида сверху с линией-выноской

ksHoleCutTypeEnum – Тип построения отверстия и выреза

ksHoleCutByWidth	0	По толщине
ksHoleCutByDepth	1	На глубину
ksHoleCutUpToSurface	2	До грани

ksHyperLinkTypeEnum – Тип гиперссылки

ksHLUnknown	0	Неизвестный
ksHLFile	1	Ссылка на файл или web-страницу
ksHLObject	2	Ссылка на объект
ksHLMail	3	Адрес электронной почты

ksHypertextTypeEnum – Тип ссылки на текст

Ссылки на вид:		
ksHTViewScale	1	Масштаб вида
ksHTViewUnfold	3	Значок «развернуто»

ksHTViewTurn	4	Значок «повернуто»
ksHTViewTurnAngle	5	Повернуто на угол
ksHTViewName	6	Имя вида
Ссылки на объект:		
ksHTObjectText	0	Основной текст объекта
ksHTObjectText1	1	Первый дополнительный текст
ksHTObjectText2	2	Второй дополнительный текст
ksHTObjectText3	3	Третий дополнительный текст
ksHTObjectText4	4	Четвертый дополнительный текст
ksHTObjectNumberingGroup	5	Группа нумерации
Ссылки на переменную:		
ksHTVariableValue	0	Значение переменной
ksHTVariableName	1	Имя переменной
ksHTVariableExpression	2	Выражение - Имя = значение
ksHTVariableComment	3	Комментарий
Ссылка на обозначение позиции:		
ksHTPositionNumber	0x10	Номер позиции
ksHTSpecificationObjectMarking	0x80	Обозначение из объекта спецификации подключенного к линии выноски
ksHTSpecificationObjectName	0x81	Наименование из объекта спецификации подключенного к линии выноски
ksHTSpecificationObjectCount	0x82	Количество из объекта спецификации подключенного к линии выноски
Ссылка на свойство объекта или документа:		
ksHTObjectProperty	0x80	Свойство объекта
Ссылка на положение объекта:		
ksHTObjectZone	-1	Обозначение зоны
ksHTObjectSheet	-2	Номер листа
Ссылка на положение вида и его обозначения:		
ksHTViewTitleZone	-1	Обозначение зоны вида
ksHTViewTitleSheet	-2	Номер листа вида
ksHTViewBaseObjectZone	-3	Обозначение зоны знака вида (стрелки вида, линии сечения, выносного элемента)
ksHTViewBaseObjectSheet	-4	Номер листа знака вида (стрелки вида, линии сечения, выносного элемента)

ksJalousieBuildingTypeEnum – Способ построения жалюзи

ksJalousieExtract	0	Вытяжка
ksJalousieTrim	1	Подрезка

ksJalousieFormEndEnum – Форма торца жалюзи

ksJalousieTangent	0	По направлению
ksJalousieNormal	1	По нормали к толщине

ksJalousieHeightTypeEnum – Способ задания высоты жалюзи

ksJalousieAllHeight	0	Полная
ksJalousieUpToFaceHeight	1	От грани
ksJalousieSlotHeight	2	Высота прорези

ksInsertionTypeEnum – Тип вставки фрагмента или вида

ksTUnknown	-1	Неизвестный
ksTBodyFragment	0	Вставка внешнего фрагмента. Взять в документ
ksTReferenceFragment	1	Вставка внешнего фрагмента. Внешней ссылкой
ksTLocalFragment	3	Вставка локального фрагмента
ksTBodyView	4	Вставка вида другого чертежа. Взять в документ
ksTReferenceView	5	Вставка вида другого чертежа. Внешней ссылкой

ksKOMPASConverterEnum – Типы внутренних конвертеров КОМПАС 3D

ksConverterToRaster	0	Конвертация в растровый формат
ksConverterToSAT	1	Конвертация в формат SAT
ksConverterToXT	2	Конвертация в формат XT
ksConverterToSTEP	3	Конвертация в формат STEP
ksConverterToIGES	4	Конвертация в формат IGES
ksConverterToVRML	5	Конвертация в формат VRML
ksConverterToSTL	6	Конвертация в формат STL

ksKompasModuleEnum – Модули Компас

ksKompasModule2D	1	Модуль 2D
ksKompasModule3D	2	Модуль 3D
ksKompasPrint	100	Модуль печати
ksKompasExport	101	Модуль экспорта

ksKompasVariantEnum – Константы вариантов реализаций Компас

ksKompasPro	0	КОМПАС.
ksKompasHome	1	КОМПАС-Home.
ksKompasViewer	2	КОМПАС-Viewer.
ksKompasSpds	4	КОМПАС-SPDS.
ksKompasGraphic	256	КОМПАС-График.
ksKompasInvisible	512	КОМПАС-Invisible.
ksKompasLatin	1024	КОМПАС - иностранная версия.
ksKompasConsumer	4096	КОМПАС-Consumer.
ksKompasStudy	8192	КОМПАС-Study.

ksLawTypeEnum – Типы законов

ksTLawConst	0	Константный
ksTLawLinear	1	Линейный
ksTLawCube	2	Кубический
ksTLawByExpression	3	По выражению

ksLeaderSignEnum – Тип значка для линии-выноски

ksLSignNone	0	Без знака
ksLGlueSign	1	Знак склеивания
ksLSolderingSign	2	Знак пайки
ksLSewingSign	3	Знак сшивания
ksLCrampSign	4	Знак соединения внахлестку металлическими скобками
ksLcornerCrampSign	5	Знак углового соединения металлическими скобами
ksLMontageJointSign	6	Знак монтажного шва

ksLengthBuildingTypeEnum – Способ расчета длины продолжения сгиба листового тела

ksLBDistance	0	На расстояние

ksLBToVertex	2	До вершины
ksLBToSurface	3	До поверхности

ksLengthUnitEnum – Единицы измерения длины

ksLUMillimetres	0	Миллиметры
ksLUCentimetres	1	Сантиметры
ksLUnDecimetres	2	Дециметры
ksLUnMetres	3	Метры
ksLUKilometres	4	Километры

ksLengthUnitsEnum – Единицы измерения длины

ksLUnSM	0	Сантиметры
ksLUnMM	1	Миллиметры
ksLUnDM	2	Дециметры
ksLUnM	3	Метры
ksLUnDocument	4	Настройки документа

ksLibraryStyleEnum – Стили отображения прикладных библиотек

ksLibraryStyleUnknown	0	Неизвестный стиль
ksLibraryStyleMenu	1	Отображение в виде меню
ksLibraryStyleDialog	2	Отображение в виде диалога
ksLibraryStyleWindow	3	Отображение в виде специального окна
ksLibraryStyleBar	4	Отображение в виде панели инструментов
ksLibraryStyleInvisible	5	Невидимый режим

ksLibraryTypeEnum – Типы библиотек

ksLibraryUnknown	0	Неизвестный тип.
ksLibraryProcedure	1	Прикладная библиотека.
ksLibraryFragment	2	Библиотека фрагментов.
ksLibraryModel	3	Библиотека моделей.
ksLibraryDocuments	4	Универсальная библиотека документов

ksLineDimensionOrientationEnum – Тип ориентации линейного размера

ksLinDParallel	0	Параллельно объекту

ksLinDHorizontal	1	Горизонтально
ksLinDVertical	2	Вертикально

ksLinearPatternBuildingTypeEnum – Способ построения массива по сетке

ksLPSaveAll	0	Оставлять копии внутри сетки
ksLPSaveAlongPerimeter	1	Оставлять копии только по периметру сетки
ksLPSaveAlongAxially	2	Оставлять копии только по осям сетки
ksLPChessOrderByAxis1	3	Шахматный порядок - сдвиг вдоль первой оси
ksLPChessOrderByAxis2	4	Шахматный порядок - сдвиг вдоль второй оси

ksLineSegment3DTypeEnum – Тип построения отрезка 3D

ksLSTTwoPoints	0	По 2 точкам
ksLSTPointLenghtAngle	1	По точке, длине и углу наклона

ksLoadStateEnum – Тип загрузки компонента

ksLUnknown	-1	Неопределен
ksLCompletely	0	Загружен полностью
ksLUnload	1	Пустой, не загружен
ksLTriangles	2	Упрощенный. Триангуляция
ksLPartially	3	Частичная загрузка
ksLGabarit	4	Габарит

Описание.

Компонент загружен полностью - загружена триангуляция (для отрисовки), загружена модель. Можно редактировать.

Компонент не загружен - не отображается в окне документа. Редактировать нельзя.

Компонент загружен частично - загружена только триангуляция (для отрисовки). Редактировать нельзя.

Примечание.

Тип загрузки относится к вставкам компонентов в сборке 3D.

ksLoftBuildingType – Способы построения элемента по сечениям у крайних сечений

ksLoftAuto	0	Автоматически
ksLoftByNormal	1	По нормали

ksLoftByObject	2	По объекту
ksLoftCupola	3	Купол

ksManipulatorTypeEnum – Способ разбиения зоны

ksPlacement3DManipulator	1	Манипулятор системы координат
ksEditDoubleManipulator	2	Манипулятор - редактор веществен-ного значения

ksManipulatorPrimitiveEnum – Тип примитива манипулятора

ksMPNone	0	Неизвестный тип
ksMPAxisX	1	Ось X
ksMPAxisY	2	Ось Y
ksMPAxisZ	3	Ось Z
ksMPPlaceXOY	4	Плоскость XOY
ksMPPlaceXOZ	5	Плоскость XOZ
ksMPPlaceYOZ	6	Плоскость YOZ
ksMPConturXY	7	Граница плоскости XOY
ksMPConturXZ	8	Граница плоскости XOZ
ksMPConturYZ	9	Граница плоскости YOZ.
ksMPTextX	10	Текст X
ksMPTextY	11	Текст Y.
ksMPTextZ	12	Текст Z.
ksMPOriginal	13	Начало координат

ksManipulatorModeEnum – Режимы работы манипулятора

ksManipulatorModeDefault	1	По умолчанию
ksManipulatorModeNotHandle Editor	2	Запрет ручного редактирования

ksMateConstraintAlignmentEnum – Варианты выравнивания направлений для сопряжений

ksMCAlignmentOpposite	-1	Противонаправленные
ksMCAlignmentClosest	0	Ориентация согласно ближайшего решения
ksMCAlignmentCooriented	1	Сонаправленные, с одинаковой ориентацией

ksMCAlignmentUnknown	2	Нет определенной ориентации
Дополнительные		
ksMCAlignment_1	3	Дополнительный вариант 1
ksMCAlignment_2	4	Дополнительный вариант 2
ksMCAlignment_3	5	Дополнительный вариант 3
ksMCReverse_1	6	Противонаправленные. Дополнительный вариант 1
ksMCReverse_2	7	Противонаправленные. Дополнительный вариант 2
ksMCReverse_3	8	Противонаправленные. Дополнительный вариант 3

ksMarkInsideFormEnum – тип формы для марки (без линии-выноски)

ksMFormEmpty	0	Без формы.
ksMFormCircle	1	Окружность.
ksMFormRectangle	2	Прямоугольник
ksMFormSquare	3	Квадрат
ksMFormRhomb1	4	Ромб 1
ksMFormRhomb2	5	Ромб 2
ksMFormHexagon	6	Шестиугольник
ksMFormTriangle1	7	Треугольник 1
ksMFormTriangle2	8	Треугольник 2
ksMFormChamferedRectangle	9	Скругленный прямоугольник
ksMFormCircleWidthVerticalDelimer	10	Окружность с вертикальным разделителем
ksMFormDoubleCircle	11	Двойная окружность

ksMarkNodeEnum – Тип узла марки

ksMarkCircle	0	Окружность
ksMarkRefCircle	1	Указатель ориентации оси с окружностью
ksMarkText	2	Текст

ksMarkOnLinePosTypeEnum – Положение марки относительно линии

ksMTextAboveLine	0	Текст над линией
ksMTextOnLine	1	Текст на линии
ksMTextUnderLine	2	Текст под линией

ksMassSettingModeEnum – Варианты задания МЦХ

ksCalculateParam	0	Расчет параметров
ksManualMass	1	Ручное задание массы

ksMassUnitsEnum – Единицы измерения массы

ksMUnGR	0	Граммы
ksMUnKG	1	Килограммы
ksMUnDocument	2	Настройки документа

ksMaterialPropertyTypeEnum – Тип события выбора материала

ksMPNewPartDocumentSettings	1	Вызов из настроек новых документов для настроек детали

ksMateTangentTypeEnum – Вид касания для сопряжения касание

ksMTangentUnknown	0	Неопределено
ksMTangentByPoint	1	Касание в общем случае (контакт точкой)
ksMTangentByGeneratrixLine	2	Касание по образующей прямой (например два цилиндра с параллельными осями)
ksMTangentByCircle	3	Касание по окружности (например сфера в конусе)

ksMateMotionTypeEnum – Тип движения компонента для механического сопряжения

ksMMotionUnknown	0	Неопределено
ksMMotionLinear	1	Линейное перемещение
ksMMotionRotation	2	Вращение

ksMathCurve3DTypeEnum – Тип математической кривой в трехмерном пространстве

ks3DCurve	0	Кривая
ksLine3D	1	Прямая
ksLineSegment3D	2	Отрезок прямой
ksArc3D	3	Окружность, эллипс, дуга
ksSpiral	4	Спираль
ksConeSpiral	5	Коническая спираль
ksCurveSpiral	6	Спираль по образующей кривой
ksCrookedSpiral	7	Спираль по направляющей кривой
ksPolyCurve3D	8	Кривая, построенная по точкам
ksPolyline3D	9	Полилиния
ksNurbs3D	10	NURBS кривая
ksBezier3D	11	Кривая Безье
ksHermit3D	12	Составной кубический сплайн Эрмита
ksCubicSpline3D	13	Кубический сплайн
ksPlaneCurve	14	Плоская кривая в пространстве
ksOffsetCurve3D	15	Эквидистантная кривая
ksTrimmedCurve3D	16	Усеченная кривая
ksReparamCurve3D	17	Репараметризованная кривая
ksBridgeCurve3D	18	Кривая-мостик, соединяющая две кривые
ksCharacterCurve3D	19	Кривая, координатные функции которой заданы в символьном виде
ksContourOnSurface	20	Контур на поверхности
ksContourOnPlane	21	Контур на плоскости
ksSurfaceCurve	22	Кривая на поверхности
ksSilhouetteCurve	23	Силуэтная кривая
ksSurfaceIntersectionCurve	24	Кривая пересечения поверхностей
ksBSpline	25	B-сплайн
ksContour3D	26	Контур

ksMathSurface3DTypeEnum – Тип математической поверхности в трехмерном пространстве

ks3DSurface	0	Поверхность
ksElementarySurface	1	Элементарная поверхность
ksPlane	2	Плоскость
ksConeSurface	3	Коническая поверхность
ksCylinderSurface	4	Цилиндрическая поверхность
ksSphereSurface	5	Сфера
ksTorusSurface	6	Тор
ksSweptSurface	7	Поверхность движения
ksExtrusionSurface	8	Поверхность перемещения
ksRevolutionSurface	9	Поверхность вращения
ksEvolutionSurface	10	Кинематическая поверхность
ksExactionSurface	11	Кинематическая поверхность с поворотными торцами
ksExpansionSurface	12	Плоскопараллельная поверхность

ksElevationSurface	13	Поверхность по сечениям с направляющей
ksSpiralSurface	14	Спиральная поверхность
ksRuledSurface	15	Линейчатая поверхность
ksSectorSurface	16	Секториальная поверхность
ksPolySurface	17	Поверхность, определяемая точками
ksHermitSurface	18	Hermit - поверхность, определяемая точками
ksSplineSurface	19	NURBS поверхность, определяемая точками
ksGridSurface	20	Поверхность, определяемая точками
ksTriBezierSurface	21	Треугольная Bezier поверхность, определяемая точками
ksTriSplineSurface	22	Треугольная NURBS поверхность, определяемая точками
ksOffsetSurface	23	Эквидистантная поверхность
ksDeformedSurface	24	Деформированная поверхность
ksNurbsSurface	25	NURBS поверхность, определяемая кривыми
ksCornerSurface	26	Поверхность по трем кривым
ksCoverSurface	27	Поверхность по четырем кривым
ksCoonsPatchSurface	28	Бикубическая поверхность Кунса по четырем кривым
ksMeshSurface	29	Поверхность на сетке кривых
ksLoftedSurface	30	Поверхность Эрмита, определяемая кривыми
ksSmoothSurface	31	Поверхность сопряжения
ksChamferSurface	32	Поверхность фаски
ksFilletSurface	33	Поверхность скругления
ksChannelSurface	34	Поверхность скругления с переменным радиусом
ksJoinSurface	35	Поверхность соединения
ksCurveBoundedSurface	36	Поверхность, усеченная кривыми (контурами) на поверхности
ksBendedUnbendedSurface	37	Поверхность, полученная сгибом/разгибом
ksCylindricBendedSurface	38	Поверхность, полученная цилиндрическим сгибом
ksCylindricUnbendedSurface	39	Поверхность, полученная цилиндрическим разгибом
ksConicBendedSurface	40	Поверхность, полученная коническим сгибом
ksConicUnbendedSurface	41	Поверхность, полученная коническим разгибом
ksExplorationSurface	42	Поверхность заметания с масштабированием и поворотом образующей кривой
ksFreeSurface	200	Тип для поверхностей, созданных пользователем

ksMeshAroundPointTypeEnum – Тип сетки, построенной вокруг точки

ksMALinear	0	Прямоугольная сетка (метрический шаг)
ksMAUVLinear	1	Прямоугольная сетка (параметрический шаг)
ksMACircular	2	Концентрическая сетка
ksMAHexagonal	3	Гексогональная сетка

ksMeshPointsSurfaceBuildingTypeEnum – Тип поверхности по сети точек

ksMPByPoints	0	Сплайновая поверхность по точкам
ksMPByPole	1	Сплайновая поверхность по полюсам

ksMeasureResultEnum – Результат измерения расстояния и угла между поверхностями

ksMResUnknown	0	Не определен
ksMResAxisAxisCoaxial	1	Оси совпадают
ksMResAxisAxisParallel	2	Оси параллельны
ksMResAxisAxisIntersect	3	Оси пересекаются
ksMResAxisAxisDistant	4	Оси на расстоянии
ksMResAxisSurfColinear	5	Ось лежит на поверхности
ksMResAxisSurfParallel	6	Ось параллельна поверхности
ksMResAxisSurfIntersect	7	Ось пересекает поверхность
ksMResAxisSurfDistant	8	Ось на расстоянии от поверхности
ksMResSurfSurfColinear	9	Одна поверхность лежит на другой
ksMResSurfSurfParallel	10	Поверхности параллельны
ksMResSurfSurfIntersecting	11	Поверхности пересекаются

ksMlEndLimiterEnum – Типы ограничений на концах мультилинии

ksMEndUnknown	0	Без ограничения
ksMEndLine	1	Прямолинейный
ksMEndArc	2	Дуговой
ksMEndPolyline	3	Ломаный

ksMlVertexLimiterEnum – Типы ограничений в вершинах мультилинии

ksMVeUnknown	0	Без ограничения
ksMVeArc	1	Дуговой
ksMVeAngle	2	Угловой стык
ksMVeTangent	3	Касательный стык

ksMlVertexTrackingEnum – Типы обхода вершин мультилинии

ksMTrShear	0	Обход срезом
ksMTrFillet	1	Обход скруглением
ksMTrEquaFillet	2	Обход скруглением с одинаковым радиусом

ksModelDrawingElementsEnum – Возможные элементы отрисовки модели

ksModelDrawingElementNone	0x0000	Пусто
ksModelDrawingElementBackground	0x0001	Фоновая подложка
ksModelDrawingElementCompBodies	0x0010,	Тела компонента
ksModelDrawingElementCompObjs	0x0020	Объекты компонента
ksModelDrawingElementAuxObjs	0x0100	Внешние объекты
ksModelDrawingElementNotifyLibs	0x0200	Библиотеки на подписке
ksModelDrawingElementEditor	0x1000	Элементы редактора, включая эскиз и фантомы
ksModelDrawingElementWidgets	0x2000	Элементы управления, оконная сетка и пр.
ksModelDrawingElementMeshCutter	0x4000	Сечение модели
ksModelDrawingElementAll	0xffff	Все элементы сразу

ksModelRenderTypeEnum – Вариант отрисовки

ksModelRenderNone	-1	Без аппаратного ускорения
ksModelRenderAuto	0	Автоматическое определение
ksModelRenderPerfected	1	Улучшенный. Триангуляция лежит на GPU. Минимальное число отрисовочных вызовов
ksModelRenderMedium	2	Базовый. Триангуляция лежит на GPU. Группировка по телам
ksModelRenderLow	3	Совместимый. Старая реализация. Минимальные требования к GPU

ksModelPerformanceLevelEnum – Качество сглаживания

ksMPLDisabledAntialiasing	0	Без сглаживания
ksMPLLowAntialiasing	1	Низкое качество сглаживания
ksMPLNormalAntialiasing	2	Среднее качество сглаживания
ksMPLHightAntialiasing	3	Высокое качество сглаживания

ksModelTransparencyTypeEnum – Прозрачность

ksModelTransparencyMesh	1	Сетчатая прозрачность
ksModelTransparencyRealistic	2	Реалистичная прозрачность

ksModelObjectParamTypeEnum – Тип параметров объекта

ksMOAllParam	0	Параметры в системе координат ЛСК объекта
ksMOPartAllParam	1	Параметры в системе координат детали
ksMOCurrentLSKAllParam	2	Параметры в системе текущей ЛСК

ksMultiThicknessGroupTypeEnum – Тип разнотолщинной группы

ksArbitraryThicknesses	0	Группы произвольных толщин
ksSetsThicknesses	1	Группы наборов толщин

ksNewDocumentSettingsTypeEnum – Тип настроек новых документов

ksNewPartDocumentSettings	1	Настройки новых документов для детали

ksNurbsByPointsAproximationTypeEnum – Способ вычисления шага аппроксимации для точек сплайна

ksNBPAproximationStepByCurvature	0	Вычисление шага аппроксимации с учетом радиуса кривизны
ksNBPAproximationStepByDeviation	1	Вычисление шага аппроксимации по угловой толерантности

ksNurbsByPointsBuildingTypeEnum – Способ формирования точек сплайна

ksNByPBTUndefined	0	Неопределенный
ksNByPBLinear	1	Линейный
ksNByPBChordLength	2	По длине хорды
ksNByPBCentripetal	3	Центростремительный

ksNurbsByPointsPointConstraintsEnum – Вариант управления точкой сплайна

ksNByPBPCNone	0	Нет
ksNByPBPCTangent	1	Касательностью
ksNByPBPCSmoothG2	2	Касательностью и кривизной
ksNByPBPCNormal	3	Перпендикулярно

ksObjectsFilter3DEnum – Способ фильтрации 3D объектов

ksFilterAll	0	Фильтровать все
ksFilterFaces	1	Фильтровать грани
ksFilterEdges	2	Фильтровать ребра
ksFilterVertexs	3	Фильтровать вершины
ksFilterCPlanes	4	Фильтровать конструктивные плоскости
ksFilterCAxis	5	Фильтровать конструктивные оси
ksFilterParts	6	Фильтровать компоненты
ksFilterBodies	7	Фильтровать тела

ksFilterSurfaces	8	Фильтровать поверхности
ksFilterSketches	9	Фильтровать эскизы
ksFilterCurves	10	Фильтровать кривые
ksFilterCS	11	Фильтровать системы координат
ksFilterControlPoints	12	Фильтровать контрольные и присоединительные точки
ksFilterPoints3D	13	Фильтровать точки
ksFilterDesignations	14	Фильтровать обозначения
ksFilterThread	15	Фильтровать условные обозначения резьбы

ksOperationResultEnum – Результат операции

ksOperationUnion	0	Объединение
ksOperationNewBody	1	Новое тело
ksOperationCut	2	Вычитание
ksOperationIntersect	3	Пересечение

ksOffsetGapType – Типы смещений зазора

ksOGParametrU	0	По параметру U
ksOGLength	1	По длине
ksOGDraftPosition	2	Угол расположения

ksOrientationTypeEnum – Тип ориентирования ЛСК

ksAxisOrientation	0	Направление осей
ksEulerCorners	1	Система углов Эйлера
ksOrientByObject	2	Ориентирование по объекту

ksOutputColorTypeEnum – Цвет вывода на печать

ksPJ_OCBlack	0	В черных линиях
ksPJ_OCByView	1	Цветом вида
ksPJ_OCByLayer	2	Цветом слоя
ksPJ_OCByObject	3	Цветом объекта

ksPartAccessTypeEnum – Тип доступа к компоненту

ksATUncertainty	-1	Неопределенный
ksATEditable	0	Редактирование
ksATReadOnly	1	Только чтение
ksATDisable	2	Доступ запрещен

ksPart7CollectionTypeEnum – Тип коллекции компонентов

ksAllParts	0	Все компоненты (включая копии из операций копирования)
ksUniqueParts	1	Первые экземпляры вставок компонентов

ksPatternBasePointTypeEnum – Способ задания базовой точки

ksCRAuto	0	Автоопределение
ksCRManual	1	Ручное указание
ksCRFirstObject	2	По первому в списке

ksPatternExemplarsOrientationTypeEnum – Способ ориентации экземпляров массива

ksOrientationSave	0	Сохранять исходную ориентацию
ksOrientationByNormal	1	Доворачивать до нормали
ksOrientationByObject	2	Ориентировать по объекту

ksPhantomTypeEnum – Типы фантома

ksUnknownPhantom	0	Нет фантома
ksMoveGroupPhantom	1	Сдвиг группы
ksLineSegPhantom	2	Отрезок
ksRectanglePhantom	3	Прямоугольник
ksAngleLineSegPhantom	4	Отрезок с заданным углом
ksHalfRectanglePhantom	5	Половина прямоугольника
ksUserPhantom	6	Пользовательский фантом
ksCirclePhantom	7	Окружность

ksPLMChangesEnum – Отличие в системе версионирования

ksPLMChangeUndefined	0	Состояние не задано
ksPLMNoChanges	1	Нет различий
ksPLMChangesNotCommited	2	Изменения не загружены в PLM
ksPLMUpdateNeeded	3	Есть изменения в PLM
ksPLMConflict	4	Конфликт

ksPLMStatusEnum – Статус в системе версионирования

ksPLMStateUndefined	0	Состояние не задано
ksPLMStateNotRegistered	1	Не зарегистрирован в PLM
ksPLMStateAvailable	2	Доступен
ksPLMStateInProgress	3	Взят в работу
ksPLMStateBlocked	4	Заблокирован
ksPLMStateError	5	Ошибка

ksPLMObjectNotifyEnum – События объектов версионирования

ksPLMStatusChanged	1001	Изменение статуса в системе версионирования
ksOrientationByObject	1002	Изменение признака отличия в системе версионирования

ksPointsArrOnCurveTypeEnum – Способ построения точек группы по кривой

ksPAOCByPointsCount	0	Равномерно по длине (Построение по заданному количеству точек)
ksPAOCByStep	1	Шаг по кривой (Построение по расстоянию между точками)
ksPAOCByParametricStep	2	Равномерный шаг по параметру кривой (Построение по параметрическому расстоянию между точками)

ksPointsArrOnSurfaceTypeEnum – Способ построения точек группы по поверхности

ksPAOSByPointsUVCount	-1	По количеству точек по U и V
ksPAOSByLinearDeflection	0	По линейному отклонению
ksPAOSByAngularDeflection	1	По угловому отклонению
ksPAOSByMeshAroundPoint	2	По сетке вокруг заданной точки

ksPoint3DCurveParamTypeEnum – Типы смещений при способе построения точки вдоль кривой

ksOffsetByU	1	По параметру U,%
ksOffsetByLen	2	По длине дуги
ksOffsetByAngle	3	По центральному углу дуги

ksPoint3DSurfaceParamTypeEnum – Типы смещений при способе построения точки на поверхности

ksOffsetByUV	1	По параметрам U и V,%
ksOffsetByLenFromObj	2	По расстояниям от плоских объектов
ksOffsetByCoords	3	По координатам на плоскости
ksOffsetByCylinderCoords	4	Смещение по цилиндрическим координатам
ksOffsetBySphereCoords	5	Смещение по сферическим координатам

ksPoint3DTypeEnum – Способы построения пространственной точки

ksPUnknown	0	Неизвестный тип
ksPParamCoord	1	По координатам от опорного объекта
ksPDisplace	2	По смещению от опорного объекта
ksPIntersect	3	На пересечении опорных объектов

ksPCenter	4	В центре опорного объекта
ksPCurve	5	На кривой со смещением
ksPSurface	6	На поверхности
ksPProjection	7	Проецированием
ksPCylindrCoord	8	По цилиндрическим координатам
ksPSphericCoord	9	По сферическим координатам

ksPointLocationTypeEnum – Положение двумерной точки относительно двумерной кривой

ksPLUndefined	0	Положение не определено, кривая разомкнута
ksPLOutside	1	Точка снаружи замкнутой кривой
ksPLOnCurve	2	Точка на кривой
ksLPInside	3	Точка внутри замкнутой кривой

ksPositionLederFormEnum – Тип формы для позиционной линии-выноски

ksPLSingleText	0	Простой текст
ksPLOpenText	1	Открытый текст
ksPLCircle	2	Круг
ksPLHexagon	3	Шестиугольник
ksPLCircleWithSeparator	4	Круг с разделителем

ksPressFormingHeightTypeEnum – Способ задания высоты штамповки

ksPressFormingAllHeight	0	Полная
ksPressFormingOutHeight	1	Снаружи
ksPressFormingInHeight	2	Внутри

ksProcessContextMenuType – Тип процессного меню

ksProcessDefaultContextMenu	0	Обычное меню процесса
ksProcessContextMenuHide	-2	Без меню
ksProcessContextPanel	-3	Контекстная панель с контролами
ksProcessContextIconMenu	-4	Меню с иконками

ksProcess3DManipulatorsNotifyEnum – События манипуляторов процесса 3D

ksRotateManipulator	1	Вращение манипулятора относительно оси на угол
ksMoveManipulator	2	Перемещение манипулятора
ksClickManipulatorPrimitive	3	Клик или двойной клик по примитиву манипулятора
ksBeginDragManipulator	4	Начало перемещения манипулятором примитива
ksEndDragManipulator	5	Завершение перемещения манипулятором примитива
ksCreateManipulatorEdit	6	Создание редактора для ввода значения, управляющего положением манипулятора
ksDestroyManipulatorEdit	7	Удаление редактора для ввода значения, управляющего положением манипулятора
ksChangeManipulatorValue	8	Завершение редактирования значения в редакторе манипулятора

ksProcessObjectsFilter3DEnum – Режим использования прямоугольной рамки для выделения объектов в процессе

ksProcessFilterBodies	0	Фильтровать тела
ksProcessFilterSurfaces	1	Фильтровать поверхности
ksProcessFilterObjects	2	Объекты модели, не являющиеся операциями
ksProcessFilterOperations	3	Операции
ksProcessFilterParts	4	Фильтровать компоненты
ksProcessFilterExcludeExternalObjects	1000	Отсекать объекты других компонентов, не находящиеся сейчас в режиме редактирования на месте

ksProtectProductStatusEnum – Состояния (статус) защиты продукта

ksProtectUnknown	0	Неопределенный, состояние неизвестно
ksProtectDemo	1	Недоступный, деморежим.
ksProtectDisabled	2	Запрещенный, отключенный
ksProtectExpired	3	Недействительный, просроченный
ksProtectNormal	4	Нормальный, рабочий
ksProtectFailed	5	Неудачный, неисправный, повреждённый

ksRuledBorderEnum – Типы кромки оснований

ksRuledBodder1	0	Перпендикулярно плоскости листа
ksRuledBodder2	1	Совпадение с поверхностями оснований

ksRuledJointEnum – Типы кромки стыка

ksRuledJoint1	0	Перпендикулярно поверхности листа
ksRuledJoint2	1	Параллельно друг другу

ksSHRibBuildingTypeEnum – Способ построения ребра усиления

ksSHRibBuildingHeightAngle	0	По стороне и углу
ksSHRibBuildingTwoHeights	1	По двум сторонам
ksSHRibBuildingDepthAngle	2	По глубине и углу

ksSHRibCutingTypeEnum – Форма сечения ребра усиления

ksSHRibCuttingV	0	V-образная
ksSHRibCuttingU	1	U-образная

ksSegmentationMethodEnum – Способ сегментации эскиза

ksSmQuantity	0	По количеству сегментов
ksSmLength	1	По длине сегментов
ksSmAngle	2	По углу
ksSmHeight	3	По величине отклонения от хорды

ksSelectionBandMode – Режим использования прямоугольной рамки для выделения объектов

ksSelectionNoBand	0	Не использовать селектирование рамкой
ksSelectionWhenNoNearestObject	2	Начинать тащить рамку, когда под курсором нет объекта

ksSpcUsedTypeEnum – Признаки использования в спецификации

ksSpcUsedInAllDescriptions	0	Отображать во всех описаниях
ksSpcUsedInCurrentDescription	1	Отображать в текущем описании

ksSplineTransitionTypeEnum – Способ создания сопряжения в заданной вершине сплайна

ksSTTNone	0	Не задан
ksSTTByParam	1	Параметрами
ksSTTConstraint	2	Сопряжением

ksStylesLibraryTypeEnum – События функции поиска объектов

ksCurveStyleLibrary	1	Библиотека стилей кривых (*.lcs)

ksHatchStyleLibrary	2	Библиотека стилей штриховок (*.lhs)
ksTextStyleLibrary	3	Библиотека стилей текстов (*.lts)
ksStampStyleLibrary	4	Библиотека стилей описаний штампов (*.lyt)
ksGraphicLayoutStyleLibrary	5	Библиотека стилей оформлений графических документов и спецификаций (*.lyt)
ksTextLayoutStyleLibrary	6	Библиотека стилей оформлений текстовых документов (*.lyt)
ksSpcLayoutStyleLibrary	7	Библиотека стилей оформлений спецификаций (*.lyt)

ksTextureTypeEnum – Тип текстуры

ksBaseProperties	-1	Все
ksTexture	0	Текстура
ksRelief	1	Рельеф
ksCutting	2	Вырезы

ksTransitionVectorIndexEnum – Индекс вектора в точке сопряжения

ksTVTangent	0	Касательный
ksTVNormal	1	Нормальный
ksTVBNormal	2	Бинормальный

ProcessTypeEnum – Типы процессов КОМПАС API

prUnknown	0	Неизвестный процесс
prPoint	10000	Точка
prPointAlong	10001	Точки по кривой
prIntersectPoint	10002	Точки пересечения 2-х кривых
prAllIntersectPoint	10003	Все точки пересечения кривой
prPointOnDistance	10004	Точка на кривой на заданном расстоянии от другой точки
prLineSeg	10005	Отрезок
prParallelLineSeg	10006	Параллельный отрезок

prPerpendLineSeg	10007	Перпендикулярный отрезок
prTanLineSegByOutsidePnt	10008	Касательный отрезок через внешнюю точку
prTanLineSegByPntOn	10009	Касательный отрезок через точку кривой
prTangent2LineSeg	10010	Отрезок, касательный к 2 кривым
prContourLineSeg	10011	Отрезок в контуре
prContourParallelLineSeg	10012	Параллельный отрезок в контуре
prContourPerpendLineSeg	10013	Перпендикулярный отрезок в контуре
prContourTanLineSegByOutsidePnt	10014	Касательный отрезок через внешнюю точку в контуре
prLine	10015	Вспомогательная прямая
prVerticalLine	10016	Вертикальная прямая
prHorizontalLine	10017	Горизонтальная прямая
prPerpendLine	10018	Перпендикулярная прямая
prParallelLine	10019	Параллельная прямая
prTangent2Line	10020	Прямая, касательная к 2 кривым
prTanLineByPntOn	10021	Касательная прямая через точку кривой
prTanLineByOutsidePnt	10022	Касательная прямая через внешнюю точку
prBisectorLine	10023	Биссектриса
prCircle	10024	Окружность
prCircle3Points	10025	Окружность по 3 точкам
prCircleCentreOnEl	10026	Окружность с центром на объекте
prCircleTangent	10027	Окружность, касательная к 1 кривой
prCircleTangent2	10028	Окружность, касательная к 2 кривым
prCircleTangent3	10029	Окружность, касательная к 3 кривым
prCircle2Points	10030	Окружность по 2 точкам
prCircleArc	10031	Дуга
prArc3Points	10032	Дуга по 3 точкам
prArc2PointsAngle	10033	Дуга по 2 точкам и углу раствора
prArc2Points	10034	Дуга по 2 точкам
prArcTangent	10035	Дуга, касательная к кривой
prContourArc	10036	Дуга по трем точкам в контуре
prContourConArc	10037	Сопряженная дуга в контуре
prEllipse	10038	Эллипс
prEllipseGabDiagonal	10039	Эллипс по диагонали прямоугольника
prEllipseTangent2	10040	Эллипс, касательный к 2 кривым
prEllipseCentre3Points	10041	Эллипс по центру и 3 точкам
prEllipseParallel3Points	10042	Эллипс по 3 вершинам параллелограмма
prEllipseParallelCentre2Points	10043	Эллипс по центру, середине стороны и вершине параллелограмма
prEllipseGabCentrePoint	10044	Эллипс по центру и вершине прямоугольника
prBezier	10045	Кривая Безье
prContourBezier	10046	Сплайн в контуре
prPolyline	10047	Ломаная
prNurbs	10048	NURBS-кривая
prContourNurbs	10049	NURBS-кривая в контуре
prRectangle	10050	Прямоугольник
prRectangleCentrePoint	10051	Прямоугольник по центру и вершине
prPolygon	10052	Многоугольник
prEquidToObj	10053	Эквидистанта кривой
prAssemblyEquid	10054	Эквидистанта по стрелке
prLineDimension	10055	Линейный размер
prCommonBaseLineDim	10056	Линейный размер от общей базы

prChainLineDim	10057	Линейный цепной размер
prCommonLineLineDim	10058	Линейный размер с общей размерной линией
pr2ObjectsLineDim	10059	Линейный размер от отрезка до точки
prCutLineDimension	10060	Линейный размер с обрывом
prAngleDimension	10061	Угловой размер
prCommonBaseAngleDim	10062	Угловой размер от общей базы
prChainAngleDim	10063	Угловой цепной размер
prCommonLineAngleDim	10064	Угловой размер с общей размерной линией
prCutAngleDimension	10065	Угловой размер с обрывом
prRadialDimension	10066	Радиальный размер
prDiametralDimension	10068	Диаметральный размер
prArcDimension	10069	Размер дуги окружности
prOrdinateDimension	10070	Размер высоты
prLeader	10071	Линия-выноска
prBrandLeader	10072	Знак клеймения
prMarkLeader	10073	Знак маркировки
prPositionLeader	10074	Обозначение позиций
prChangeLeader	10075	Знак изменения
prHatch	10076	Штриховка
prText	10077	Ввод текста
prTable	10078	Ввод таблицы
prRough	10079	Шероховатость
prBase	10080	База
prCutLine	10081	Линия разреза
prViewPointer	10082	Стрелка взгляда
prRemoteElement	10083	Выносной элемент
prAxedLineSegment	10084	Осевая линия по двум точкам
prCentreMarker	10085	Обозначение центра
prAssemblyContour	10086	Собрать контур
prFormTolerance	10087	Допуск формы
prInsertRaster	10088	Вставить растровый объект
prMakeMacro	10089	Объединить в макроэлемент
prInsertFragment	10090	Вставить внешний фрагмент
prCreateSheetView	10091	Создать вид
prInsertOLEObject	10092	Вставить OLE-объект
prCreateStandartSheetView	10093	Создать стандартные виды
prCreateSectionSheetView	10094	Создать вид разрез/сечение
prCreateArbitrarySheetView	10095	Создать произвольный вид
prCreateProjectionSheetView	10096	Создать проекционный вид
prCreateArrowSheetView	10097	Создать вид по стрелке
prCreateRemoteSheetView	10098	Создать вид - выносной элемент
prCreateLocalSheetView	10099	Создать местный вид
prCreateLocalSectionSheetView	10100	Создать местный вид-разрез
prCreateBrokenSheetView	10101	Создать вид с разрывом
prContour	10102	Непрерывный ввод объектов
prChamfer	10103	Фаска между пересекающимися объектами
prChamferPolyContour	10104	Фаска на углах объекта
prFillet	10105	Скругление между пересекающимися объектами
prFilletPolyContour	10106	Скругление на углах объекта
prProjectionObject	10107	Спроецировать объект на плоскость эскиза

prSmartDimension	10108	Авторазмер
prSmartAxedLineSegment	10109	Автоосевая
prMeasurePointProperties	10110	Измерить координаты точки в локальной системе координат
prMeasureDistance2Points	10111	Измерить расстояние между двумя точками
prMeasureDistance2PointsByCurve	10112	Измерить расстояние между двумя точками на кривой
prMeasureDistancePointObject	10113	Измерить расстояние от кривой до точки
prMeasureDistance2Curves	10114	Измерить расстояние между двумя кривыми
prMeasureAngle2Lines	10115	Измерить угол между двумя прямыми/ отрезками
prMeasureAngle3Points	10116	Измерить угол, заданный 3 точками
prPerimeter	10117	Измерить длину кривой и суммарную длину
prMeasureArea	10118	Измерить площадь
prMix	10119	Расчет массово-центровочных характеристик плоских фигур
prMix3DRevolution	10120	Расчет массово-центровочных характеристик тел вращения
prMix3DExtrision	10121	Расчет массово-центровочных характеристик тел выдавливания
prObjectShift	10122	Сдвиг выделенных объектов
prObjectShiftAngleLen	10123	Сдвиг с заданием угла и расстояния
prObjectRotate	10124	Поворот выделенных объектов
prObjectScale	10125	Масштабирование выделенных объектов
prObjectSymmetry	10126	Симметричное отображение выделенных объектов
prObjectMultiply	10127	Копирование выделенных объектов
prObjectMultiplyByCurve	10128	Копирование выделенных объектов по кривой
prObjectMultiplyByCircle	10129	Копирование выделенных объектов по окружности
prObjectMultiplyByRing	10130	Копирование выделенных объектов по концентрической сетке
prObjectMultiplyByMesh	10131	Копирование выделенных объектов по сетке
prMoveDeformation	10132	Деформация сдвигом
prRotateDeformation	10133	Деформация поворотом
prScaleDeformation	10134	Деформация масштабированием
prCutObjectPart	10135	Усечь кривую
prCutObjectPartBy2Points	10136	Усечь кривую двумя точками
prJustify	10137	Выровнять кривую по границе
prRemoveChamfer	10138	Удалить фаску или скругление
prBreakCurve	10139	Разбить кривую на две части
prBreakCurveNParts	10140	Разбить кривую на N равных частей
prBlackBox	10141	Очистить заданную область
prConvertToNurbs	10142	Преобразовать геометрический объект или текст в NURBS-кривые
prParametricHorizontal	10143	Установить горизонтальность отрезка или прямой

prParametricVertical	10144	Установить вертикальность отрезка или прямой
prParametricXAlign	10145	Выровнять по горизонтали две характерные точки объектов
prParametricYAlign	10146	Выровнять по вертикали две характерные точки объектов
prParametricMergePoints	10147	Объединить две точки
prParametricPointOnCurve	10148	Задать размещение точки на кривой
prParametricPointSymmetry	10149	Симметрия 2 точек относительно оси
prParametricParallel	10150	Установить параллельность двух прямых и/или отрезков
prParametricNormal	10151	Установить перпендикулярность двух прямых и/или отрезков
prParametricColinear	10152	Установить коллинеарность двух прямых и/или отрезков
prParametricTangent	10153	Установить касание двух кривых
prParametricFixPoint	10154	Зафиксировать координаты точки
prParametricEqualRadiuses	10155	Установить равенство радиусов двух дуг и/или окружностей
prParametricEqualLength	10156	Установить равенство длин двух отрезков
prParametricFixDimension	10157	Зафиксировать значение размера
prParametricChangeDimension	10158	Установить значение размера
prParametricSelected	10159	Параметризовать выделенные объекты
prParametricDeleteObjConstraints	10160	Показать/удалить ограничения
prParametricDeleteAllConstraints	10161	Удалить все ограничения
prSelectObject	10162	Выделить отдельный объект
prSelectLayer	10163	Выделить слой указанием лежащего на этом слое объекта
prSelectSheetView	10164	Выделить вид указанием точки внутри этого вида
prSelectWithRect	10165	Выделить объекты внутри прямоугольной рамки
prSelectOutSideRect	10166	Выделить объекты снаружи от прямоугольной рамки
prSelectWithCutRect	10167	Выделить объекты, пересекающиеся с прямоугольной рамкой
prSelectWithCutPolyline	10168	Выделить объекты, пересекающиеся с ломаной
prExcludeObject	10169	Исключить отдельный объект
prExcludeLayer	10170	Исключить слой указанием лежащего на этом слое объекта
prExcludeSheetView	10171	Исключить вид указанием точки внутри этого вида
prExcludeWithRect	10172	Исключить объекты внутри прямоугольной рамки
prExcludeOutSideRect	10173	Исключить объекты снаружи от прямоугольной рамки
prExcludeWithCutRect	10174	Исключить объекты, пересекающиеся с прямоугольной рамкой
prExcludeWithCutPolyline	10175	Исключить объекты, пересекающиеся с ломаной
prSmartLine	10176	Автолиния

prBrace	10177	Фигурная скобка
prAutoDimL	10178	Авторазмер - ввод линейного размера
prAutoDimA	10179	Авторазмер - ввод углового размера
prAutoDimD	10180	Авторазмер - ввод диаметрального размера
prAutoDimR	10181	Авторазмер - ввод радиального размера
prAutoDimLToPoint	10182	Авторазмер - ввод линейного размера от отрезка до точки
prAutoDimLBreak	10183	Авторазмер - ввод линейного размера с обрывом от отрезка до отрезка осевой линией
prAutoDimABreak	10184	Авторазмер - ввод углового размера с обрывом от отрезка до отрезка осевой линией
prTechnicalDemandPlacement	10185	Технические требования-размещение
prDirectAxis	10186	Прямая координационная ось
prArcAxis	10187	Дуговая координационная ось
prCircleAxis	10188	Круговая координационная ось
prWaveLine	10189	Волнистая линия
prMarkOnLDRPosNum	10190	Марка/Позиционное обозначение на линии выноске
prMarkWoLDRPosNum	10191	Марка/Позиционное обозначение без линии выноски
prKnotNumber	10192	Номер узла
prUnitMarking	10193	Обозначение узла
prCutUnitMarking	10194	Обозначение узла в сечении
prMultiTextLeader	10195	Выносная надпись к многослойным конструкциям
prColouring	10196	Заливка
prMultiLine	10197	Мультилиния
prBuildingCutLine	10198	Линия разреза/сечения для СПДС
prBrokenLine	10199	Линия обрыва с изломами
prCreateReport	10200	Создать отчет
prCreateAttachedLeaders	10201	Редактировать оформление составного объекта
prEditProperties	10202	Процесс редактирования свойств объекта или документа
prParametricBisector	10203	Создание ограничения биссектриса
prParametricFixedLenght	10204	Фиксировать длину
prParametricFixedAngle	10205	Фиксировать угол
prParametricPointOnCurveMiddle	10206	Точка на середине кривой
prTechnicalDemand	10207	Ввод\Редактирование технических требований
prSpecRough	10208	Ввод\Редактирование неуказанной шероховатости
prMoveSpecRough	10209	Ручное размещение неуказанной шероховатости
prDeleteHistory	10210	Удаление истории построения
prUndo	10211	Undo
prRedo	10212	Redo
prEmbodimentsReport	10213	Создать таблицу исполнений
prArrayParamReport	10214	Создать таблицу параметров массива
prConicCurve	10215	Коническая кривая

prConicCurve4Or5Point	10216	Коническая кривая по 4 или 5 точкам
prMarkOnLeader	10217	Марка/позиционное обозначение с линией-выноской
prCutLineMultiple	10218	Линия сложного разреза/сечения
prBuildingCutLineMultiple	10219	Линия сложного разреза/сечения для СПДС
prCircularCentres	10220	Круговая сетка центров
prLinearCentres	10221	Линейная сетка центров
prBaseExtrusion	20000	Базовая операция выдавливания
prBossExtrusion	20001	Приклеивание выдавливанием
prCutExtrusion	20002	Вырезать выдавливанием
prExtrusionSurface	20003	Поверхность выдавливания
prBaseRotated	20004	Базовая операция вращения
prBossRotated	20005	Приклеивание вращением
prCutRotated	20006	Вырезать вращением
prRotatedSurface	20007	Поверхность вращения
prBaseEvolution	20008	Кинематическая операция
prBossEvolution	20009	Приклеить кинематически
prCutEvolution	20010	Вырезать кинематически
prEvolutionSurface	20011	Кинематическая поверхность
prBaseLoft	20012	Базовая операция по сечениям
prBossLoft	20013	Приклеивание по сечениям
prCutLoft	20014	Вырезать по сечениям
prLoftSurface	20015	Поверхность по сечениям
prFillet3D	20016	Операция "фаска"
prChamfer3D	20017	Операция "скругление"
prCutByPlane	20018	Операция "сечение поверхностью"
prCutBySketch	20019	Операция "сечение эскизом"
prMeshCopy	20020	Операция копирования по сетке
prCircularCopy	20021	Операция копирования по концентрической сетке
prCurveCopy	20022	Операция копирования по кривой
prMirrorCopy	20023	Операция "зеркальный массив"
prMirrorAllCopy	20024	Операция "зеркально отразить все"
prDerivativePartArray	20025	Операция массив по образцу для сборки
prMeshPartArray	20026	Операция массив по сетке для сборки
prCircularPartArray	20027	Операция массив по концентрической сетке для сборки
prCurvePartArray	20028	Операция массив по кривой для сборки
prIncline	20029	Операция "уклон"
prShell	20030	Операция "оболочка"
prRib	20031	Операция "ребро жесткости"
prHole	20032	Отверстие
prThread	20033	Условное изображение резьбы
prCPlaneOffset	20034	Смещенная плоскость
prCPlane3Points	20035	Плоскость по 3-м точкам
prCPlaneAngle	20036	Плоскость под углом
prCPlaneEdgePoint	20037	Плоскость через ребро и вершину
prCPlaneParallel	20038	Плоскость через вершину параллельно другой плоскости
prCPlanePerpendicular	20039	Плоскость через вершину перпендикулярно ребру
prCPlaneNormalToSurface	20040	Нормальная плоскость
prCPlaneTangentToSurface	20041	Касательная плоскость

prCPlaneLineToEdge	20042	Плоскость через ребро параллельно/ перпендикулярно другому ребру
prCPlaneLineToFlat	20042	Плоскость через ребро параллельно/ перпендикулярно грани
prCAxis2Points	20043	Ось по двум точкам
prCAxis2Planes	20044	Ось по двум плоскостям
prCAxisConeface	20045	Ось конической грани
prCAxisEdge	20046	Ось, проходящая через ребро
prCAxisOperation	20047	Ось операции
prPolyline3D	20048	Ломаная
prSpline3D	20049	Сплайн
prCylindricSpiral	20050	Цилиндрическая спираль
prConicSpiral	20051	Коническая спираль
prImportedSurface	20052	Импортированная поверхность
prInsertScetch	20053	Эскиз из библиотеки
prEditScetch	20054	Редактировать эскиз
prOrientationScetch	20055	Разместить эскиз на плоскости
prInPlacePartEdit	20056	Редактировать компонент на месте
prOutPlacePartEdit	20057	Редактировать компонент в своем окне
prAddDetail	20058	Вставить деталь в сборку
prAddAssembly	20059	Вставить сборку в сборку
prMateCoincident	20060	Сопряжения компонентов - Совпадение
prMateConcentric	20061	Сопряжения компонентов - Соосность
prMateParallel	20062	Сопряжения компонентов - Параллельность
prMatePerpendicular	20063	Сопряжения компонентов - Перпендикулярность
prMateOnDistance	20064	Сопряжения компонентов - На расстоянии
prMateOnAngle	20065	Сопряжения компонентов - Под углом
prMateTangent	20066	Сопряжения компонентов - Касание
prPartVariables	20067	Просмотр и редактирование переменных
prCopyBilletPart	20068	Создание детали путем копирования детали из другого файла
prMakeMoldCavity	20069	Вычесть компоненты
prMakeUnionComps	20070	Объединить компоненты
prAddPartFromFile	20071	Добавить компонент из файла
prMovePart	20072	Переместить компонент
prRotatePartWC	20073	Повернуть компонент вокруг центральной точки
prRotatePartAxis	20074	Повернуть компонент вокруг оси
prRotatePartPoint	20075	Повернуть компонент вокруг точки
prMakeSplitLine	20076	Построение линии разъема
prMeasureDistance3D	20077	Измерить расстояние и угол
prMeasurePerimeter3D	20078	Измерить длину ребра
prMeasureArea3D	20079	Измерить площадь
prMeasureMix3D	20080	Вычисление массово-центровочных характеристик
prMeasureInterferenceVolumes	20081	Проверка коллизий
prBaseShMtSolid	20082	Листовое тело
prShMtBend	20083	Построение сгиба вдоль ребра листового тела
prShMtCombinedBend	20084	Построение сгибов вдоль ребер листового тела по эскизу

prShMtBendLine	20085	Создание сгиба в листовом теле по прямолинейному объекту
prShMtBendHook	20086	Создание подсечки в листовом теле по прямолинейному объекту
prShMtHole	20087	Построение круглого отверстия на грани листового тела
prShMtCut	20088	Построение выреза на грани листового тела
prBaseShMtPlate	20089	Добавление пластины к листовому телу
prShMtClosedCorner	20090	Замыкание углов двух смежных элементов листового тела
prShMtBendStraighten	20091	Разгибание элементов листового тела
prShMtBendBended	20092	Сгибание элементов листового тела
prShMtBendParamUnfold	20093	Настроить параметры развертки листового тела
prPatchSurface	20094	Создание поверхности по замкнутому контуру
prSewSurface	20095	Сшивка поверхностей
prMakeFaceRemover	20096	Удалить грани
prCPlaneMiddle	20097	Средняя плоскость
prCPointControl	20098	Контрольная точка
prCPointConjunctive	20099	Присоединительная точка
prCAggregateOper	20100	Булева операция
prCPlaneLineToFlat	20101	Плоскость через ребро параллельно/ перпендикулярно грани
prPoint3D	20103	Конструктивная 3D точка
prLocalCoordinateSystem	20104	Локальная система координат
prLineDimention3DPlane	20105	Размер по двум объектам и планару.
prLineDimention3D	20106	Размер по ребру и точке
prAngleDimention3D	20107	Угловой размер
prRough3D	20108	Шероховатость
prTolerance3D	20109	Допуск формы и расположения поверхностей
prBrandLeader3D	20110	Клеймение
prMarkerLeader3D	20111	Маркировка
prPositionLeader3D	20112	Обозначение позиции
prBase3D	20113	База
prLeader3D	20114	Линия-выноска
prSaveBody	20115	Процесс сохранения тела в деталь
prCreateSketch	20116	Процесс создания\редактирования эскиза
prMeasureInformation	20117	Информация об объекте
prEquidistant3D	20118	Эквидистанта 3D
prChoiceOperationResult	20119	Выбор результата 3D операции
prChoiceBodyUnit	20120	Выбор частей тела
prSelectCurrentCS	20121	Выбрать текущую СК в модели
prShmtRuledOperation	20122	Обечайка
prArc3D	20123	Дуга в пространстве
prConnectCurve	20124	Сопряжения пространственных кривых – соединение
prTrimCurve	20125	Сопряжения пространственных кривых – обрезка
prFilletCurve	20126	Сопряжения пространственных кривых – скругление
prSwithOwnCS	20127	Перенести в СК модели

prScalingOperation	20128	Маштабирование тела или поверхности
prPointDrivenPattern	20129	Массив пространственных объектов по точкам
prNurbs3DByObjects	20130	3D сплайн по объектам
prCurveOperationCrossing	20131	Кривая пересечения пространственных объектов
prConvertToNurbsSurface3D	20133	NURBS-поверхность по объектам
prNurbsSurface3DByPoints	20134	NURBS-поверхность по точкам
prNurbsSurface3DByCurves	20135	NURBS-поверхность по сети кривых
prArrayPointsFromFile	20136	Массив пространственных точек из файла
prArrayPointsOnCurve	20137	Массив пространственных точек вдоль кривой
prArrayPointsByCloud	20138	NURBS-поверхность по облаку точек
prOffsetSurface	20139	Эквидистанта поверхности
prAuxObjectMultiplyByMesh	20140	Копирование вспомогательной геометрии в пространстве по плоской параллелограмной сетке
prAuxObjectMultiplyByRing	20141	Копирование вспомогательной геометрии в пространстве по плоской концентрической сетке
prAuxObjectMultiplyByCurve	20142	Копирование вспомогательной геометрии в пространстве вдоль кривой
prTrimmedSurface	20143	Усечение поверхности по объекту
prSurfaceToBody	20144	Создание тела из поверхности приданием ей толщины
prAxisByDirection	20145	Ось через вершину по объекту
prRuledSurface	20146	Линейчатая поверхность
prExtensionSurface	20147	Продление поверхности
prCPlaneTangentAtPoint	20148	Плоскость, касательная к грани в точке
prCPlaneAtCurve	20149	Плоскость через плоскую кривую
prArrayPintsOnSyrface	20150	Группа точек по поверхности
prAuxObjectMultiplyMirror	20151	Зеркальная копия вспомогательной геометрии в пространстве
prOutlineCurve	20152	Построение линии очерка поверхности
prSplineOnSurface	20153	Сплайн на поверхности
prPartsPointDrivenPattern	20154	Массив компонентов по точкам
prChooseLinearPattern	20155	Процесс выбора трехмерных объектов для копирования По сетке
prChooseCircularPattern	20156	Процесс выбора трехмерных объектов для копирования По концентрической сетке
prChooseCurvePattern	20157	Процесс выбора трехмерных объектов для копирования По кривой
prChoosePointDrivenPattern	20158	Процесс выбора трехмерных объектов для копирования По точкам
prChooseTablePattern	20159	Процесс выбора трехмерных объектов для копирования По таблице
prChooseMirrorPattern	20160	Процесс выбора трехмерных объектов для построении симметричной копии
prTablePattern	20161	Массив операций по таблице
prAuxTablePattern	20162	Массив вспомогательной геометрии по таблице
prPartsTablePattern	20163	Массив компонентов по таблице

prAuxPointDrivenPattern	20164	Массив вспомогательной геометрии по точкам
prBodiesLinearPattern	20165	Массив тел по сетке
prBodiesCircularPattern	20166	Массив тел по концентрической сетке
prBodiesCurvePattern	20167	Массив тел по кривой
prBodiesPointDrivenPattern	20168	Массив тел по точкам
prBodiesTablePattern	20169	Массив тел по таблице
prContour3D	20170	Контур 3D
prCurveOper2Projection	20171	Пространственная кривая по 2 проекциям
prCurveByLaw	20172	Пространственная кривая по закону
prBodyReposition	20173	Изменить положение тела или поверхности
prIsoparamCurve	20174	Изопараметрическая пространственная кривая
prIsoparamCurveArr	20175	Группа изопараметрических кривых на поверхности
prBindingMesh	20176	Редактировать как сплайн по сетке
prSaveBodyAs	20177	Сохранение тела в деталь
prBlendSurface	20178	Поверхность соединения
prLineSegment3D	20179	Отрезок 3D
prEmbodiment	20180	Создание исполнения
prCreateSpecificationObjects	20181	Создать объекты спецификации
prDeleteSpecificationObjects	20182	Удалить объекты спецификации
prCreateSpecificationFromAssembly	20183	Создать спецификацию по сборке
prSpecRough3D	20184	Ввод\Редактирование неуказанной шероховатости 3D
prShmtRuledCowling	20185	Линейчатая обечайка
prAddLocalPartFromFile	20187	Добавить локальную деталь из файла
prAddLayoutGeometryFromFile	20188	Добавить компоновочную геометрию
prAddBilletPartFromFile	20189	Добавить деталь заготовку
prMateSymmetry	20190	Сопряжения компонентов - Симметрия
prMateDependent	20191	Сопряжения компонентов - Зависимое положение
prMateCamGear	20192	Сопряжения компонентов - Кулачковый механизм. Кулачек-толкатель
prMateRotation	20193	Сопряжения компонентов - Вращение-Вращение
prMateRotationTransfer	20194	Сопряжения компонентов - Вращение-Перемещение
prTechnicalDemand3D	20195	Ввод\Редактирование технических требований 3D
prHoleSimple	20196	Отверстие простое
prHoleCounterbore	20197	Отверстие с цековкой
prHoleCountersinking	20198	Отверстие с зенковкой
prHoleCounterdrill	20199	Отверстие с зенковкой и цековкой
prHoleConic	20200	Отверстие коническое
prPoint3DCoord	20201	Конструктивная 3D точка. Точка по координатам
prPoint3DDisplace	20202	Конструктивная 3D точка. Точка переносом
prPoint3DIntersect	20203	Конструктивная 3D точка. Точка на пересечении
prPoint3DCurve	20204	Конструктивная 3D точка. Точка на кривой

prPoint3DSurface	20205	Конструктивная 3D точка. Точка на поверхности
prPoint3DCenter	20206	Конструктивная 3D точка.Точка в центре
prPoint3DProjection	20207	Конструктивная 3D точка. Точка в центре
prPoint3DCylindrCoord	20208	Конструктивная 3D точка. Цилиндрические координаты
prPoint3DSphericCoord	20209	Конструктивная 3D точка. Сферические координаты
prShMtBendObject	20210	Листовой металл, сгибы листовых операций
prShMtClocingPressForming	20211	Листовой металл, закрытая штамповка
prShMtOpeningPressForming	20212	Листовой металл, открытая штамповка
prShMtShoulder	20213	Листовой металл, буртик
prShMtJalousie	20214	Листовой металл, жалюзи
prShMtRib	20215	Ребро усиления
prAxis3D	20216	Осевая линия
prFullFillet	20217	Полное скругление
prRestoredSurface	20218	Восстановленная поверхность
prCurvatureGraph	20219	График кривизны
prContinuityCheck	20220	Проверка непрерывности
prZoomWindow	32411	Увеличить масштаб окном
prMoveView	32418	Сдвинуть изображение
prPanoramaView	32419	Приблизить/отдалить изображение
prRotateView	32420	Повернуть изображение (для 3D-окна)
prEditSelectedObject	35736	Редактировать выделенный объект
prEditCopy	0xE122	Копировать в буфер обмена
prEditCut	0xE123	Вырезать в буфер обмена
prEditPaste	0xE125	Вставить из буфера обмена

ksProjectionOptionEnum – Опции проецирования

ksPOUndefined	-1	Неопределенное состояние
ksPODisable	0	Не проецировать
ksPOEnable	1	Проецировать
ksPOByLayer	2	По слою

ksPropertyTypeEnum – Типы свойств

ksPropertyDataTypeUnknown	0	Неизвестный
ksPropertyDataTypeLong	1	Целый
ksPropertyDataTypeDouble	2	Вещественный
ksPropertyDataTypeString	3	Строка
ksPropertyDataTypeBoolean	4	Логический
ksPropertyDataTypeColorRGB	5	Цвет RGB
ksPropertyDataTypeHatchStyle	6	Стиль штриховки
ksPropertyDataTypeGroup	7	Группа свойств

ksRedrawDocumentModeEnum – Режим перерисовки окон документа

ksRedrawFull	0	Полная перерисовка
ksRedrawAnimation	1	Выполнение анимации сцены
ksRedrawSelection	2	Селектирование или подсветка
ksRedrawOperationPhantom	3	Операционные фантомы
ksRedrawDimensions	4	Операционные размеры
ksRedrawPhantomObjects	5	Фантомные объекты
ksRedrawHighlightObjects	6	Подсветка объектов под курсором
ksRedrawWidgets	7	Манипуляторы, 3D хот-точки и т.п

ksRecoverErrorEnum – Признак ошибок при открытии документа с восстановлением

ksRNoError	0	Проверка завершена. Ошибок не найдено
ksRecover	1	Проверка завершена. Найденные ошибки исправлены
ksRNoOpen	2	Проверка завершена. Открыть документ не удалось
ksRAlreadyOpen	3	Файл уже открыт в одном из окон. Проверка невозможна
ksRProtected	4	Файл защищен с помощью приложения КОМПАС-Защита. Проверка невозможна

ksRegionTypeEnum – Тип региона

ksRTInside	1	Полностью внутри
ksRTOutside	2	Снаружи
ksRTCutFrame	3	Секущей рамкой

ksRelativeProjectionTypeEnum – Тип проекции стандартного вида относительно главного вида

ksPtNone	-1	Не определена
ksPtFront	1	Спереди - Фронтальная плоскость
ksPtRear	2	Сзади
ksPtUp	3	Сверху - Горизонтальная плоскость
ksPtDown	4	Снизу
ksPtLeft	5	Слева - Профильная плоскость
ksPtRight	6	Справа
ksPtIsoXYZ	7	Изометрия XYZ

ksRelationTypeEnum – Тип родственных отношений

ksRTUnknown	0	Не определен
ksRTIndifferent	1	Все отношения
ksRTStrong	2	Сильные отношения

ksReportFiltersTypeEnum – Типы фильтров в команде Создать отчет

ksFilterConditionUnknown	0	Неизвестный
ksFilterConditionEqual	1	=
ksFilterConditionSmaller	2	<
ksFilterConditionLarger	3	>
ksFilterConditionEqualOrSmaller	4	=<
ksFilterConditionEqualOrLarger	5	=>
ksFilterConditionContain	6	содержит
ksFilterConditionNotContain	7	не содержит

ksRequestFilesTypeEnum – Тип процесса, запрашивающего файл или список файлов

			Допустимые расширения	Множественный выбор
ksRFUnknown	0	Неизвестный тип		
ksRFSaveBody	1	Сохранение тела в модель	m3d	-

koRFUnitParts	2	Объединение вставок в подсборку	a3d	-
koRFCopyBilletPart	3	Вставка детали заготовки	m3d	-
koRFSavePartAs	4	Сохранение вставки в файл	m3d, a3d	-
koRFAddDetail	5	Вставка детали в сборку	m3d	-
koRFAddAssembly	6	Вставка подсборки в сборку	a3d	-
koRFAddPartFromFile	7	Добавить компонент из файла	m3d, a3d	-
koRFChangeDetailFile	8	Заменить файл источник для вставки детали	m3d	-
koRFChangeAssemblyFile	9	Заменить файл источник для вставки сборки	a3d	-
koRFChangeBilletPartFile	10	Заменить файл источник для детали заготовки	a3d	-
koRFSpcObjAddDocument	11	Подключение документов КОМПАС к объекту спецификации	frw,cdw,m3d,a3d, kdw	+
koRFSpcAssemblyAddDocument	12	Подключение документов КОМПАС к спецификации - Управление сборкой	cdw,a3d	+
koRFAddLocalDetail	13	Добавить локальную деталь	m3d	
koRFAddLayoutGeometry	14	Добавить компоновочную геометрию	m3d, a3d	

ksRibSideEnum – Положение ребра жесткости

ksRibSideLeft	0	Прямое направление в плоскости эскиза. Ребро выдавливается в левую сторону от кривой вдоль плоскости
ksRibSideRight	1	Обратное направление в плоскости эскиза. Ребро выдавливается в правую сторону от кривой вдоль плоскости
ksRibSideUp	2	Прямое направление ортогонально эскизу. Ребро выдавливается в сторону нормали плоскости
ksRibSideDown	3	Обратное направление ортогонально эскизу. Ребро выдавливается в сторону против нормали плоскости

ksRotatedTypeEnum – Способы определения угла вращения

ksRTAngle	0	Угол
ksRTVertex	1	До вершины
ksRTSurface	2	До поверхности

ksRoughSignEnum – Тип значка шероховатости

ksNoProcessingType	0	Без указания типа обработки
ksDeleteMaterial	1	С удалением слоя материала
ksWithoutDeleteMaterial	2	Без удаления слоя материала

ksSemiAxisTypeEnum – Тип полуоси для обозначения центра

ksAxisUnknown	-1	Неизвестный
ksAxisXPlus	0	Полуось по оси X
ksAxisXMinus	1	Полуось против оси X
ksAxisYPlus	2	Полуось по оси Y
ksAxisYMinus	3	Полуось против оси Y

ksSheetsRangeEnum – Тип диапазона страниц

ksAllSheets	0	Все страницы
ksUnevenSheets	1	Нечетные страницы
ksEvenSheets	2	Четные страницы

ksShelfDirectionEnum – Направление полки

ksLSLeft	-1	Влево
ksLSNone	0	Нет полки
ksLSRight	1	Вправо
ksLSUp	2	Вверх
ksLSDown	3	Вниз

ksSketchBendBuildingTypeEnum – Способы построения сгиба по эскизу

ksSBFromSketch	1	От эскиза
ksSBSomeEdges	2	Вдоль всего ребра

ksShoulderBuildingTypeEnum – Способ построения буртика

ksShoulderUnknown	0	Не определено
ksShoulderWidth1	1	Использовать при расчете ширину основания
ksShoulderHeight	2	Использовать при расчете высоту
ksShoulderAngle	4	Использовать при расчете угол
ksShoulderWidth2	8	Использовать при расчете ширину дна
ksShoulderRadius2	16	Использовать при расчете радиус буртика
ksShoulderWidth1Off	-1	Не использовать при расчете ширину основания
ksShoulderHeightOff	-2	Не использовать при расчете высоту
ksShoulderAngleOff	-4	Не использовать при расчете угол
ksShoulderWidth2Off	-8	Не использовать при расчете ширину дна
ksShoulderRadius2Off	-16	Не использовать при расчете радиус буртика
ksShoulderCircleByHeightWidth1	3	Круговая по высоте и ширине основания
ksShoulderCircleByHeightRadius2	18	Круговая по высоте и радиусу буртика
ksShoulderCircleByRadiusWidth1	17	Круговая по радиусу буртика и ширине основания
ksShoulderUByHeighAngleWidth1	7	U-образная по высоте, углу и ширине основания
ksShoulderUByHeighAngleWidthRadius2	23	U-образная по высоте, углу и ширине основания со скруглением дна
ksShoulderUByHeighWidth1Width2	11	U-образная по высоте, ширине основания и ширине дна (две ширины)
ksShoulderUByHeighWidth1Width2Radius2	27	U-образная по высоте, ширине основания и ширине дна (две ширины) со скруглением дна
ksShoulderUByAngleWidth1Width2	13	U-образная по углу, ширине основания и ширине дна (две ширины)
ksShoulderUByAngleWidth1Width2Radius2	29	U-образная по углу, ширине основания и ширине дна (две ширины) со скруглением дна
ksShoulderUByHeighAngleWidth2	14	U-образная по высоте, углу и ширине дна
ksShoulderUByHeighAngleWidth2Radius2	30	U-образная по высоте, углу и ширине дна со скруглением дна
ksShoulderVByHeighAngleWidth1	7	V-образная по высоте, углу и ширине основания
ksShoulderVByHeighAngleRadius2	22	V-образная по высоте, углу и радиусу буртика
ksShoulderVByRadius2AngleWidth1	21	V-образная по радиусу буртика, углу и ширине основания
ksShoulderVByHeighRadius2Width1	19	V-образная по высоте, радиусу буртика и ширине основания

ksShoulderCutingTypeEnum – Форма сечения буртика

ksShoulderCutingCircle	0	Круговая
ksShoulderCutingU	1	U-образная
ksShoulderCutingV	2	V-образная

ksShoulderTypeEnum – Тип буртика. Способ обработки концов буртика

ksShoulderClosed	0	Закрытый
ksShoulderOpened	1	Открытый
ksShoulderChopped	2	Рубленый

ksSlaveDocumentTypeEnum –Типы подчиненных режимов редактирования документов КОМПАС

ksSDSketchMode	1000	Режим редактирования эскиза
ksSDSpecificationSlave	1001	Слейв режим редактирования спецификации

ksSnapTypeEnum – Тип привязки к объектам

ksSTUnknown	-1	Информация отсутствует
ksSTUndefine	0	Нет привязки
ksSTNearestPoint	1	Ближайшая точка
ksSTNearestMiddle	2	Ближайшая середина
ksSTObjectCentre	3	Центр объекта
ksSTIntersect	4	Пересечение
ksSTGrid	5	Привязка по сетке
ksSTXYAlign	6	Выравнивание по X Y
ksSTAngleSnap	7	Угловая привязка
ksSTPointOnCurve	8	Точка на кривой

ksSTNormalToCurve	9	По нормали на кривую
ksSTTangentToCurve	10	По касательной на кривую

ksSortTypeEnum – Типы сортировки

ksSortTypeNone	0	Нет сортировки
ksSortTypeCompositeUp	1	Составная сортировка по возрастанию колонок
ksSortTypeUp	3	Сортировка по возрастанию колонок
ksSortTypeDocument	4	Сортировка раздела документация
ksSortTypeDown	5	Сортировка по убыванию колонок
ksSortTypeCompositeDown	6	Составная сортировка по убыванию колонок

ksSpecificationColumnTypeEnum – Типы колонок спецификации

ksSColumnUnknown	0	Неизвестный тип
ksSColumnFormat	1	Формат
ksSColumnZone	2	Зона
ksSColumnPosition	3	Позиция
ksSColumnMark	4	Обозначение
ksSColumnName	5	Наименование
ksSColumnCount	6	Количество
ksSColumnNote	7	Примечание
ksSColumnMass	8	Масса
ksSColumnMaterial	9	Материал
ksSColumnUser	10	Пользовательская
ksSColumnCode	11	Код
ksSColumnFactory	12	Завод изготовитель
ksSColumnDocumentNumber	13	Номер документа
ksSColumnDocumentName	14	Наименование документа
ksSColumnDocumentCode	15	Код документа
ksSColumnCodeOKP	16	Код ОКП

ksSpecificationObjectStateEnum – Состояние объекта спецификации

ksObjectStateIndependent	0	Самостоятельный объект
ksObjectStateFromInsert	1	Объект из вставки
ksObjectStateEdit	2	Объект редактировался в документе
ksObjectStateUserSetNotEdit	3	Пользователь снял признак редактирования.

ksSpecificationObjectTypeEnum – Типы объектов для спецификации

В API5 соответствует Типам объектов спецификации...

ksSpecificationUnknownObject	0	Неизвестный тип
ksSpecificationBaseObject	1	Базовый объект
ksSpecificationComment	2	Комментарий
ksSpecificationSectionName	3	Имя раздела
ksSpecificationBlock	4	Начало блока
ksSpecificationReserveString	5	Резервная строка
ksSpecificationEmptyString	6	Пустая строка

ksSpecificationVariantEnum – Варианты оформления спецификации

ksSpecificationSimple	0	Простая
ksSpecificationVariantA	1	Вариант А
ksSpecificationVariantB	2	Вариант Б
ksSpecificationVariantV	3	Вариант В
ksSpecificationVariantG	4	Вариант Г

ksSpecRoughPlacementEnum – Размещение неуказанной шероховатости

ksRPTopLeft	0	Вверху слева
ksRPTopRight	1	Вверху справа
ksRPBottomRight	2	Внизу справа

ksSpiral3DHeightTypeEnum – Способ задания высоты спирали 3D

ksSHTByValue	0	По заданному значению высоты
ksSHTByObject	1	По объекту
ksSHTByCurve	2	По длине плоской кривой

ksSplineTangentEnum – Тип направления касательной

ksSTNone	0	Не задано
ksSTByDirection	1	По направлению
ksSTCurveU	2	К изопараметрической кривой по направлению U
ksSTCurveV	3	К изопараметрической кривой по направлению V
ksSTSurfaceCurve	4	К кривой на поверхности

ksSpline3DBuildingTypeEnum – Способ построения спирали 3D

ksSBTByStepAndTurnCount	0	По шагу и количеству витков
ksSBTByStepAndHeight	1	По шагу и высоте
ksSBTByTurnCountAndHeight	2	По количеству витков и высоте

ksSpline3DDiameterTypeEnum – Способ построения спирали 3D

ksSDTByValue	0	По заданному значению диаметра
ksSDTByObject	1	По объекту
ksSDTByGeneratrixTiltAngle	2	По углу наклона образующей

ksStepTypeEnum – Способы вычисления приращения параметра по объекту

ksSpaceStep	0x01	Шаг по стрелке прогиба
ksDeviationStep	0x02	Шаг по углу отклонения
ksMetricStep	0x04	Шаг по длине
ksParamStep	0x08	Шаг для для привязки объектов к параметрам поверхности
ksCollisionStep	0x10	Шаг для определения столкновений элементов модели

ksMipStep	0x20	Шаг для расчета инерционных характеристик

ksSystemControlStartEnum – Результаты передачи управления системе КОМПАС

ksSCStoppedByMenuCommand	1	Выполнена команда меню **Остановить работу библиотеки**
ksSCCloseApplication	0	Идет закрытие системы КОМПАС/Не запущен SystemControlStart.
ksSCStopItself	-1	Вызов функции SystemControlStop из-под библиотеки
ksSCAlreadyStarted	-2	Управление системе КОМПАС уже передано той же библиотекой
ksSCStartedByAnotherLibrary	-3	Управление системе КОМПАС уже передано другой библиотекой
ksSCError	-4	Ошибка

ksTablePointEnum – Тип расположения точки на таблице

ksTPLeftBottom	1	Левый нижний угол
ksTPLeftCenter	2	Середина левой стороны
ksTPLeftUp	3	Левый верхний угол
ksTPUpCenter	4	Середина верхней стороны
ksTPRightUp	5	Правый верхний угол
ksTPRightCenter	6	Середина правой стороны
ksTPRightBottom	7	Правый нижний угол
ksTPBottomCenter	8	Середина нижней стороны
ksTPCenter	9	Середина точка для контрола
		Базовая точка
ksTPUndefined	0	Не задано

ksTableTileLayoutEnum – Расположение заголовка таблицы

ksTTLFirstRow	0	В первой строке
ksTTLFirstColumn	1	В первом столбце
ksTTLNotCreate	2	Не создавать

ksTabulatorFillingEnum – Заполнение табулятора

ksTabulatorFillingNone	0	Без заполнения
ksTabulatorFillingNone	1	Базовая линия

ksTabulatorFillingCenterLine	2	Средняя линия
ksTabulatorFillingBaseDot	3	Базовые точки
ksTabulatorFillingCenterDot	4	Средние точки
ksTabulatorFillingBaseDash	5	Базовый пунктир
ksTabulatorFillingCenterDash	6	Средний пунктир

ksTextAlignEnum – Выравнивание текста для внешнего объекта GDI

ksTALeft	0	Слева
ksTARight	2	Справа
ksTAHCenter	6	Центр горизонтали
ksTATop	0	Вверху
ksTABottom	8	Внизу
ksTABaseline	24	Базовая линия
ksTAVCenter	56	Центр вертикали

ksTextHorizontalFormatEnum – Признак горизонтального форматирования текста на чертеже

ksHFormatNot	0	Перенос правой границы (нет форматирования)
ksHFormatStrNarrowing	1	Сужение текста
ksHFormatDivision	2	Перенос на другую строку

ksTextExportFormEnum – Представление текста при экспорте

ksTEFTextOnly	1	Только текст
ksTEFGeometryOnly	2	Геометрическое представление текста

ksTextItemEnum – Тип компонента текста

ksTItString	0	Строка.
ksTItNumerator	0x1	Числитель.
ksTItDenominator	0x2	Знаменатель.
ksTItFractionEnd	0x3	Конец дроби.
ksTItUpperDeviation	0x4	Верхнее отклонение.

ksTItLowerDeviation	0x5	Нижнее отклонение.
ksTItDeviationEnd	0x6	Конец отклонений.
ksTItSBase	0x7	Основание выражения типа суммы.
ksTItSUpperIndex	0x8	Верхний индекс выражения типа суммы.
ksTItSLowerIndex	0x9	Нижний индекс выражения типа суммы.
ksTItSEnd	0x10	Конец выражения типа суммы.
ksTItSpecialSymbol	0x11	Спецзнак.
ksTItSpecialSymbolEnd	0x12	Конец спецзнаков с текстом.
ksTItSpecialSymbolNext	0x13	Начало для ввода следующих строк в спецзнаке с текстом.
ksTItSpecialSymbolDown	0x14	Для ввода строк снизу в спецзнаке с текстом.
ksTItSpecialSymbolRight	0x15	Для ввода строк справа в спецзнаке с текстом.
ksTItTab	0x16	Табуляция по текущему стилю.
ksTItFontSymbol	0x17	Символ шрифта.
ksTItHyperText	0x2000	Ссылка на текст или положение объекта
ksTItFontSymbolW	0x2017	Символ шрифта Unicode

ksTextLineType – Тип строки текста

ksTLError	Ошибка
ksTLText	Простой текст
ksTLVerticalText	Вертикальный текст
ksTLFragment	Вставка фрагмента
ksTLRaster	Вставка рисунка
ksTLTable	Вставка таблицы

ksTextNumberingEnum – Тип нумерации абзаца

ksTNumbUnknown	-1	Тип не определенный.
ksTNumbNoNumber	0	Срока без нумерации.
ksTNumbNumber	1	Строка с нумерацией уровня level.
ksTNumbNewNumber	2	На строке начинается новая нумерация пунктов.
ksTNumbDisableNumber	3	Строка не должна нумероваться никогда.

ksTextSizeEnum – Размерный коэффициент текста

ksTextDefault	0	Умолчательной высоты.
ksTextNormal	1	Нормальной высоты.
ksTextMiddle	2	Средней высоты.

ksTextSmall	3	Малой высоты.
ksTextBig	2	Большой высоты.

ksTextStyleEnum – Системные стили текста

ksTSDefault	0	Умолчательный стиль для данного типа объекта.
ksTSDrawingAnnotation	1	Текст на чертеже.
ksTSSpecifications	2	Текст для технических требований.
ksTSDimensionText	3	Текст для размерной надписи.
ksTSSurfaceFinish	4	Текст для шероховатости.
ksTSLeader1	5	Текст для линии выноски (позиционной .
ksTSLeader2	6	Текст для линии выноски (над/под полкой).
ksTSLeader3	7	Текст для линии выноски (сбоку).
ksTSShapeDeviations	8	Текст для отклонений формы и базы.
ksTSTableHeader	9	Текст для таблицы (заголовок).
ksTSTableCell	10	Текст для таблицы (ячейка).
ksTSSectionLine	11	Текст для линии разреза/сечения.
ksTSDirectionArrow	12	Текст для стрелки вида.
ksTSUnspecifiedSurfaceFinish	13	Текст для неуказанной шероховатости.
ksTSModificationSymbol	14	Текст для обозначения изменения.
ksTSBrace	15	Текст для фигурной скобки.
ksTSUnitNumber	16	Текст для номера узла.
ksTSMultiTextLeader	17	Текст для выносной надписи.
ksTSUnitMarking	18	Текст для обозначения узла.
ksTSAxisMark	19	Текст для марки координационной оси.
ksTSMarkOnLeader	20	Текст для МПО (марка/позиционное обозначение с линией-выноской).
ksTSMarkOnLine	21	Текст для МПО (марка/позиционное обозначение) на линии.
ksTSMarkInsideForm	22	Текст для МПО (марка/позиционное обозначение) без линии выноски.
ksTSBOMTableName	23	Текст для заголовков спецификации.
ksTSBuildingCutLine	24	Текст для линия разреза/сечения для СПДС.
ksTSRprtTableHeader	25	Текст для таблицы отчета (заголовок).
ksTSRprtTableCell	26	Текст для таблицы отчета (ячейка).
ksTSRprtTableName	27	Текст для таблицы отчета (наименование)
ksTSTableName	28	Текст для таблицы (наименование)
ksTSTextMark	29	Текст текстовой метки

ksThemeEnum – Темы Компас

ksThemeLight	0	Светлая
ksThemeDark	2	Темная

ksTolerancePrefixSignEnum – Знак в обозначении допуска

ksTPSNone	0	Нет
ksTPSRadius	1	Радиус
ksTPSDiametr	2	Диаметр

ksTPSToleranseT	3	Допуск в диаметральном выражении
ksTPSToleranseT2	4	Допуск в радиусном выражении

ksToleranceRecalcsEnum – Способ пересчета размера

ksTRUnknown	0	Неопределенное состояние
ksTRLowerLimit	1	По нижнему пределу
ksTRTopLimit	2	По верхнему пределу
ksTRMiddle	3	В середину поля допуска
ksTRCoefficient	4	С коэффициентом
ksTRUser	5	Пользовательский вариант пересчета (Для вставки)

ksToleranceSuffixSignEnum – Знак в обозначении базы допуска

ksTSNone	0	Нет
ksTSToleranseM	1	Зависимый допуск
ksTSToleranseS	2	Независимый допуск
ksTSToleranseP	3	Выступающее поле допуска

ksUndercutDistanceTypeEnum – Способ задания размера

ksUCDictanceOut	0	Снаружи
ksUCDictanceIn	1	Внутри
ksUCDictanceAll	2	Полный

ksUnfoldTypeEnum – Способы определения длины развертки

ksCoefficient	0	Коэффициент нейтрального слоя
ksValueBend	1	Величина сгиба
ksDecreaseBend	2	Уменьшение сгиба
ksTableBends	3	Таблица сгибов

ksValueTypeEnum – Типы значения атрибута, его колонок и колонок спецификации

ksValueTypeUnknown	0	Неизвестный
ksValueTypeInteger	1	Целое со знаком
ksValueTypeFloat	2	Вещественное
ksValueTypeString	3	Строка
ksValueTypeRecord	4	Запись

ksVector3DParametersTypeEnum – Типы параметров вектора

ksVector3DUnknown	0	Не определен
ksVector3D2Vertex	1	По двум вершинам
ksVector3DCSAngle	2	Угол в плоскости СК (Плоскости XY YZ XZ)
ksVector3DAxis	3	По оси СК
ksVector3DCoefficients	4	По коэффициентам
ksVector3D2Angles	5	По двум углам
ksVector3DEdge	6	По прямолинейному ребру, оси или перпендикулярно плоскости кривой
ksVector3DPlane	7	По оси цилиндра или перпендикулярно плоской грани, плоскости
ksVector3DSurface	8	Перпендикулярно грани в указанной точке
ksVector3DCurve	9	По базисному вектору в точке кривой (кроме прямолинейных объектов)
ksVector3DScreen	10	Перпендикулярно плоскости экрана

ksViewProjectionType – Тип проекции

ksVPNone	-1	Не определена
ksVPNormalTo	0	Нормально к
ksVPFront	1	Спереди - Фронтальная плоскость
ksVPRear	2	Сзади
ksVPUp	3	Сверху - Горизонтальная плоскость
ksVPDown	4	Снизу
ksVPLeft	5	Слева - Профильная плоскость
ksVPRight	6	Справа
ksVPIsometric	7	Изометрия
ksVPDimetric	8	Диметрия
ksVPUnfold	9	Развертка
ksVPUser	10	Пользовательская проекция

ksVisibleStateEnum – Состояние видимости объекта

ksVSUndefined	-1	Неопределенное состояние
ksVSVisible	0	Видимый
ksVSHidden	1	Невидимый
ksVSByLayer	2	По слою

ksZoneDivisionTypeEnum – Способ разбиения зоны

ksZoneDivisionRegular	0	Равномерно по осям
ksZoneDivisionByPlanes	1	По набору плоскостей

ksZoneTypeEnum – Способ создания зоны

ksZoneFree	0	Зона без параметров (Результат операции разбиение зон).
ksZoneByPoints	1	По координатам габаритного прямоугольника
ksZoneByObjects	2	По суммарному габариту объектов

ks3DLineStyle – Стили 3D линий для отрисовки с помощью OpenGL

ksCS3DNoDrawing	0	Линия не отрисовывается.
ksCS3DSolid	1	Сплошная линия.
ksCS3DDashed	2	Штриховая линия.
ksCS3DDotted	3	Пунктирная линия.
ksCS3DDashDot	4	Штрихпунктирная линия.
ksCS3DDashDotLDash2Dots	5	Штрихпунктирная линия (штрих и 2 точки).

LayersGroupWayEnum – Способ группировки слоев

wgLayers	0	Группировать слои
wgLayersCharacteristics	1	Группировать свойства слоев

MateConstraintDirection – Направления сопряжений

-1	объекты разнонаправленные,
0	не учитывать направление,
1	объекты однонаправленные.

MateConstraintFixed – Типы фиксации

0	нет фиксации,
1	фиксировать деталь 1,
2	фиксировать деталь 2.

paramType – Тип редактирования макроэлемента

MP_DBL_CLICK_OFF	0x01	- редактирование по двойному нажатию выключено,
MP_HOTPOINTS	0x02	- интерфейс hot точек включен,
MP_EXTERN_EDIT	0x04	- интерфейс внешнего управления,
MP_PROPERTY_OBJECT	0x08 //>0	- интерфейс внешних свойств объекта.

PropertyControlNameVisibility – Видимость имени элемента управления на Панели свойств

ksNameAlwaysVisible	0	Всегда показывать имя элемента управления на Панели свойств.
ksNameHorizontalVisible	1	Показывать имя элемента управления на горизонтальной Панели.
ksNameVerticalVisible	2	Показывать имя элемента управления на вертикальной Панели.
ksNameNoVisible	3	Не показывать имя элемента управления.

PropertyManagerLayout – Положение панели свойств

pmAlignRight	3	Панель прикреплена справа
pmAlignLeft	4	Панель прикреплена слева
pmAlignRightInGroup	5	Панель прикреплена справа в группе
pmAlignLeftInGroup	6	Панель прикреплена слева в группе

SaveDocumentVersion – Способ записи документа

sdv_Prev	-1	Предыдущую версию
sdv_Current	0	Текущую версию (обычный режим)
sdv_Kompas_5_11_R03	1	Версия KOMPAS_5_11_R03_VERSION
sdv_Last	1	Последняя версия
sdv_Kompas_6_0	2	Версия KOMPAS_6_0_VERSION
sdv_Kompas_6_Plus	3	Версия KOMPAS_6_PLUS_VERSION
sdv_Kompas_7_0	4	Версия KOMPAS_7_0_VERSION
sdv_Kompas_7_Plus	5	Версия KOMPAS_7_PLUS_VERSION
sdv_Kompas_8_0	6	Версия KOMPAS_8_0

SeparatorTypeEnum – Типы элемента управления Панели свойств – "разделитель" (сепаратор)

ksSeparatorDownName	0	Вывод имени снизу под сепаратором.
ksSeparatorUpName	1	Вывод имени сверху над сепаратором.
ksSeparatorWithoutLine	2	Вывод только имени; сепаратор отсутствует.
ksSeparatorBMPLeftName	3	Вывод значка с именем слева.
ksSeparatorBMPRightName	4	Вывод значка с именем справа.

SlideTypeEnum – Тип отображения слайда в окне

ksSlide	-1	Отображение слайда
ksBitmap	1	Отображение растрового изображения
ksGroup	2	Отображение группы
ksKompasDocument	3	Отображение документа КОМПАС
ksKompasText	4	Отображение текста в формате КОМПАС

SpecificationLinkTypeEnum – Режимы связи сборки или чертежа со спецификацией

ksLinkNone	0	Нет.
ksLinkOnlyObjects	1	Только вставка объектов спецификации.
ksLinkWithPositionCalculate	2	Связь с расчетом позиций.

SpecPropertyButtonEnum – Предопределенные кнопки панели свойств

pbEnter	1	Ввод
pbEsc	2	Отказ
pbAutoCreate	3	Автосоздание
pbSaveState	4	Запомнить состояние
pbNewSearch	5	Новый поиск
pbPrevObj	6	Предыдущий объект
pbNextObj	7	Следующий объект
pbHelp	8	Справка
pbCopyProps	9	Копировать свойства

SpecPropertyToolBarEnum – Предопределенные спецпанели для Панели свойств

pnUnknown	0	Неизвестная панель
pnEmpty	1	Пустая панель
pnEscHelp	2	Панель (pbEsc pbHelp)
pnEnterEscHelp	3	Панель (pbEnter pbEsc pbHelp)
pnEnterEscCreateHelp	4	Панель (pbEnter pbEsc pbAutoCreate pbHelp)
pnEnterEscCreateSaveHelp	5	Панель (pbEnter pbEsc pbAutoCreate SaveState pbHelp)
pnEnterEscCreateSaveSearchHelp	6	Панель (pbEnter pbEsc pbAutoCreate SaveState pbNewSearch pbHelp)
pnEnterEscSaveSearchPrevNextHelp	7	Панель (pbEnter pbEsc SaveState pbNewSearch pbPrevObj pbNextObj pbHelp)
pnEnterEscSearchHelp	8	Панель (pbEnter pbEsc pbNewSearch pbHelp)
pnEscSaveSearchHelp	9	Панель (pbEsc SaveState pbNewSearch pbHelp)
pnEnterEscCreateSearchHelp	10	Панель (pbEnter pbEsc pbAutoCreate pbNewSearch pbHelp)
pnEnterEscSaveSearchHelp	11	Панель (pbEnter pbEsc SaveState pbNewSearch pbHelp)
pnEscSaveStateHelp	12	Панель (pbEsc SaveState pbHelp)
pnEnterEscSearchPrevNextHelp	13	Панель (pbEnter pbEsc pbNewSearch pbPrevObj pbNextObj pbHelp)

ZoomTypeEnum – Тип изменения масштаба отображения документа в окне

ksZoomNext	0	Следующий масштаб
ksZoomPrevious	1	Предыдущий масштаб
ksZoomAll	2	Показать весь документ

Перечисления событий

ksDocumentFileNotifyEnum

ksObject2DNotifyEnum

ksKompasObjectNotifyEnum

ksNotifyTypeEnum

ksDocumentFrameNotifyEnum – События для окна документа: клавиатура, мышь, события по отрисовке документа

frBeginPaint	1	Начало отрисовки документа
frClosePaint	2	Конец отрисовки документа
frMouseDown	3	Нажатие кнопки мыши
frMouseUp	4	Отпускание кнопки мыши
frMouseDblClick	5	Двойной щелчок мыши
frBeginPaintGL	6	Начало создания листа в контексте OpenGL
frClosePaintGL	7	Окончание создания листа в контексте OpenGL
frAddGabarit	8	Определение габаритов документа
frActivate	9	Окно активизировалось
frDeactivate	10	Окно деактивизировалось
frCloseFrame	11	Закрытие окна
frMouseMove	12	Перемещение мыши
frShowOcxTree	13	Активизация закладки дерева документа
frBeginPaintTmpObjects	14	Начало отрисовки временных объектов (фантомов)
frClosePaintTmpObjects	15	Конец отрисовки временных объектов (фантомов)

ksPropertyManagerNotifyEnum – События Панели свойств или процессных параметров

prButtonClick	1	Нажата кнопка спецпанели
prChangeControlValue	2	Изменено значение элемента управления
prControlCommand	3	Нажата кнопка элемента управления
prButtonUpdate	4	Задано состояние кнопки спецпанели
prProcessActivate	5	Активизирован процесс
prProcessDeactivate	6	Деактивирован процесс
prCommandHelp	7	Вызвана справка
prSelectItem	8	Выделен элемент списка
prCheckItem	9	Выбран элемент списка
prEditFocus	11	Установка/снятие фокуса на редакторе ввода
prUserMenuCommand	12	Нажатие пункта пользовательского меню

prLayoutChanged	13	Изменение размещения панели свойств
prGetContextMenuType	14	CLLBACK для получения типа контекстного меню
prFillContextPanel	15	CLLBACK для накачки контекстной панели
prFillContextIconMenu	16	CLLBACK для накачки меню с иконками
prEndEditItem	17	Завершение редактирования элемента
prChangeTabExpanded	18	Сворачивание/Разворачивание закладки панели свойств

ksPropertyUserControlNotifyEnum – События пользовательского элемента управления

puCreateOCX	1	Создан элемент управления OCX
puDestroyOCX	2	Удален элемент управления OCX

ksViewsAndLayersManagerNotifyEnum – События для менеджера видов и слоев

vmBeginEdit	1	Начато редактирование
vmEndEdit	2	Завершено редактирование

SystemPropertyType – Номера системных свойств

4	Обозначение
5	Наименование
6	Количество
8	Масса
9	Материал
10	Плотность
11	Автор
12	Организация
13	Комментарий
14	Тип объекта в дереве построения модели
15	Позиция
16	Полное имя файла
17	Короткое имя файла
18	Дата создания
19	Дата последнего изменения
20	Раздел спецификации
21	Разработал
22	Проверил
23	Утвердил
24	Т. контр.
25	Н. контр
26	Класс точности
37	Знак неуказанной шероховатости
38	Параметр неуказанной шероховатости

ksViewProjectionScheme – Схема ориентаций модели

ksVPSUnknown	-1	Не определена
ksVPSUser	0	Пользовательская
ksVPSZAxonometric	1	Z-аксонометрия
ksVPSYAxonometric	2	Y-аксонометрия
ksVPSXAxonometric	3	X-аксонометрия
ksVPSZ90AxonometricISO	4	Z-аксонометрия (ISO 90)
ksVPSY90AxonometricISO	5	Y-аксонометрия (ISO 90)
ksVPSX90AxonometricISO	6	X-аксонометрия (ISO 90)

KompasObject – Интерфейс API КОМПАС

Интерфейс событий...

Примечание:

Получить интерфейс KompasObject можно:

- для обычных *.rtw библиотек - применив экспортную функцию IDispatch* CreateKompasObject();
- для ActiveX библиотек - после вызова команды библиотеки и выполнения функции ExternalRunCommand;
- при работе под управлением внешнего приложения (контроллера) - после вызова стандартной системной функции.

KompasObject – свойства

currentDirectory – Текущий каталог

Интерфейс...

Тип данных: BOOL

Синтаксис:

currentDirectory = Object.currentDirectoryVisible	=	Получить свойство (*)
Object.currentDirectory = currentDirectory	=	Установить свойство (*)
currentDirectory = Object.GetCurrentDirectory()	=	Получить свойство (**)
Object.SetCurrentDirectory(currentDirectory)		Установить свойство (**)

lookStyle – Тип отрисовки визуальной части приложения КОМПАС

Интерфейс...

Тип данных: long

Синтаксис Automation:

lookStyle = iKompasObject.lookStyle	Получить свойство (*)
iKompasObject.lookStyle = lookStyle	Установить свойство (*)

lookStyle = iKompasObject.GetLookStyle()	Получить свойство (**)
iKompasObject.SetLookStyle(lookStyle)	Установить свойство (**)

Примечание:

Свойство lookStyle может принимать или возвращать значения из ksTypeLookStyle.

visible – Свойство видимости приложения

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

visible = iKompasObject.Visible	Получить свойство (*)
iKompasObject.Visible = visible	Установить свойство (*)
visible = iKompasObject.GetVisible( )	Получить свойство (**)
iKompasObject.SetVisible (visible)	Установить свойство (**)

Примечание:

Позволяет получить и установить свойство видимости приложения КОМПАС-3D.

KompasObject – методы

ActiveDocument3D – Получить указатель на интерфейс текущего документа трехмерной модели

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetActive3dDocument.

Синтаксис Automation:

LPDISPATCH ActiveDocument3D();

Возвращаемое значение:

- указатель на интерфейс документа трехмерной модели ksDocument3D.

Синтаксис COM:

Чтобы получить интерфейс текущего документа трехмерной модели при программировании на COM, нужно использовать экспортную функцию LPDOCUMENT3D ksGetActive3dDocument();

Возвращаемое значение:

- указатель на интерфейс документа трехмерной модели IDocument3D.

Примечание:

Если документ трехмерной модели не активен, функция возвращает NULL.

ActiveDocument2D – Получить указатель на интерфейс текущего графического документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH ActiveDocument2D();

Возвращаемое значение:

- указатель на интерфейс графического документа ksDocument2D.

Примечание:

Если документ не активен, функция возвращает NULL.

ActiveDocumentTxt – Получить указатель на интерфейс текущего текстового документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH ActiveDocumentTxt();

Входные параметры:

- Указатель на интерфейс текстового документа ksDocumentTxt

Примечание:

Если документ не активен, функция возвращает NULL.

DataBaseObject – Получить указатель на интерфейс ksDataBaseObject для работы с базами данных

Интерфейс...

Синтаксис Automation:

LPDISPATCH DataBaseObject();

Возвращаемое значение:

- указатель на интерфейс ksDataBaseObjec для работы с базами данных.

Document2D – Получить указатель на интерфейс графического документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH Document2D();

Возвращаемое значение:

- указатель на интерфейс графического документа ksDocument2D.

Document3D - Получить указатель на интерфейс документа трехмерной модели

Интерфейс...

Синтаксис Automation:

LPDISPATCH Document3D();

Возвращаемое значение:

- указатель на интерфейс документа трехмерной модели ksDocument3D.

Синтаксис COM:

Чтобы получить интерфейс документа трехмерной модели при программировании на COM, нужно использовать экспортную функцию ksGet3dDocument.

Возвращаемое значение:

- указатель на интерфейс документа трехмерной модели IIDocument3D.

DocumentTxt – Получить указатель на интерфейс текстового документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH DocumentTxt();

Входные параметры:

- Указатель на интерфейс текстового документа ksDocumentTxt

GetAttributeObject – Получить указатель на интерфейс для работы с атрибутами

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetAttributeObject();

Возвращаемое значение:

- указатель на интерфейс ksAttributeObject.

GetDynamicArray – Получить указатель на интерфейс динамического массива ksDynamicArray

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetDynamicArray (long type);

Входной параметр:

type - тип динамического массива.

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray.

BOOL preview,

long typeDir);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - фильтр формируется автоматически),
preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна,
typeDir	- стартовая папка.

Возвращаемое значение:

- строка с именем файла.

Примечание:

Параметр typeDir может иметь значения sptSYSTEM_FILES и sptLIBS_FILES; во всех остальных случаях открывается текущая папка.

GetFragmentLibrary – Получить указатель на интерфейс библиотеки фрагментов

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetFragmentLibrary();

Возвращаемое значение:

- указатель на интерфейс ksFragmentLibrary.

GetIterator – Получить указатель на интерфейс ksIterator для навигации по объектам

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetIterator();

Возвращаемое значение:

- указатель на интерфейс ksIterator.

GetMathematic2D – Получить указатель на интерфейс для работы с математическими функциями

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMathematic2D();

Возвращаемое значение:

- указатель на интерфейс ksMathematic2D.

GetModelLibrary – Получить указатель на интерфейс библиотеки моделей

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksGetModelLibrary.

Синтаксис Automation:

LPDISPATCH GetModelLibrary();

Возвращаемое значение:

- указатель на интерфейс ksModelLibrary.

GetParamStruct – Получить указатель на интерфейс структуры параметров объекта нужного типа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetParamStruct (short structType);

Входной параметр:

structType - тип интерфейса параметров

Возвращаемое значение:

- указатель на интерфейс указанного типа из StructType2D.

ksAttachKompasLibrary – Подключить библиотеку

Интерфейс...

Синтаксис Automation:

long ksAttachKompasLibrary (BSTR libname);

Входные параметры:

libname	- полное имя файла библиотеки.

Возвращаемое значение:

уникальный номер библиотеки	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

После успешного выполнения метода имя библиотеки появится в меню "Библиотеки".

ksCalculate – Подсчитать значение выражения

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/490_Glava57_Zadanie_zavisimoste.htm

Аналог данного метода при использовании API экспортных функций - ksCalculate.

Синтаксис Automation:

long ksCalculate (BSTR s, double* res);

Входные параметры:

s	- строка с выражением,
res	- результат расчета.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Поддерживаются функции и переменные:

- SIN, COS, TAN, ATAN - тригонометрические функции (аргумент в радианах),
- SIND, COSD, TAND, ATAND - тригонометрические функции (аргумент в градусах),

- SQRT, EXP, LN, ABS - корень квадратный, экспонента, натуральный логарифм, абсолютное значение.

2. Если s = "A1 = 100", то будет заведена переменная A1 со значением 100; переменных может быть неограниченное количество.

ksCalculateReset – Очистить массив переменных калькулятора

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCalculateReset.

Синтаксис Automation:

long ksCalculateReset();

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksClearFileCache – Очистить кеш поиска файлов

Интерфейс...

Синтаксис Automation:

BOOL ksClearFileCache();

ksConvertLangMenu – Конвертировать меню в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangMenu.

Синтаксис Automation:

long ksConvertLangMenu (long hMenu);

Входной параметр:

hMenu	- дескриптор исходного меню.

Возвращаемое значение:

	- дескриптор переведённого меню.

ksConvertLangStr – Конвертировать строку src в dst в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangStr.

Синтаксис Automation:

BSTR ksConvertLangStr (BSTR src);

Входной параметр:

src	- исходная строка.

Возвращаемое значение:

- переведенная строка.

Примечание.

Словарь сохраняется в файле с расширением dic. Строки файла должны иметь следующий формат:

"Вращение"="Rotation"

"Параметры"="Parameters"

...

...

"Перемещение"="Moving"

Строки файла должны быть отсортированы по алфавиту.

ksConvertLangStrEx – Конвертировать строку с идентификатором в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangStrEx.

Синтаксис Automation:

BSTR ksConvertLangStrEx (long hInstance, long strID);

Входной параметр:

hInstance	- HINSTANCE модуля, в котором находится переводимая строка,
strID	- идентификатор строки.

Возвращаемое значение:

- переведенная строка.

ksConvertLangStrEx2 – Конвертировать строку в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangStrEx.

Синтаксис Automation:

BSTR ksConvertLangStrEx2(VARIANT hInstance, long strID);

Входные параметры:

hInstance	- HINSTANCE модуля, в котором находится переводимая строка,
strID	- идентификатор строки.

Возвращаемое значение:

- переведенная строка.

Примечание:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо KompasObject::ksConvertLangStrEx.

ksConvertLangWindow – Конвертировать окно с входящими дочерними окнами в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangWindow.

Синтаксис Automation:

BOOL ksConvertLangWindow (long hWnd);

Входной параметр:

hWnd	- дескриптор окна,

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

ksConvertLangWindowEx – Конвертировать окно с входящими дочерними окнами в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangWindowEx.

Синтаксис Automation:

BOOL ksConvertLangWindowEx (long hWnd,

long hInstance,

BSTR dlgID);

Входной параметр:

hWnd	- дескриптор окна,
hInstance	- HINSTANCE модуля, в котором находится переводимый диалог,
dlgID	- имя ресурса диалога.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64. Рекомендуется использовать функцию KompasObject::ksConvertLangWindowEx2.

ksConvertLangWindowEx2 – Конвертировать окно в соответствии с текущим словарем

Интерфейс...

Аналог данного метода при использовании экспортных функций ksConvertLangWindowEx.

Синтаксис Automation:

BOOL ksConvertLangWindowEx2(long hWnd, VARIANT hInstance, BSTR dlgID);

Входной параметр:

hWnd	- дескриптор окна,

hInstance	- HINSTANCE модуля, в котором находится переводимый диалог,
dlgID	- имя ресурса диалога.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо KompasObject::ksConvertLangWindowEx.

ksCreateInsertionFragment – Создать вставку фрагмента

Интерфейс...

Синтаксис Automation:

int LIB_FUNC ksCreateInsertionFragment(char * fileName);

Входные параметры:

fileName	- полное имя файла фрагмента.

Возвращаемое значение:

1	- в случае удачи.

Примечание:

Допускается fileName следующего вида: "c:_f\gr\lib1.lfr|детали|литье|фланец" где:

- c:_f\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье|- разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

Функция запускает процесс вставки фрагмента из файла. Функция не ждет завершения процесса вставки.

ksDetachKompasLibrary – Отключить библиотеку

Интерфейс...

Синтаксис Automation:

long ksDetachKompasLibrary (long libId);

Входные параметры:

libId	- уникальный номер библиотеки, полученный при выполнении ksAttachKompasLibrary.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksDrawBitmapEx2 – Отрисовать BMP с идентификатором bmpID в заданном окне(hWindow)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawBitmapEx.

Синтаксис Automation:

long ksDrawBitmapEx2(long HWindow, long bmpID, VARIANT hInst);

Входные параметры:

Hwindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,
sldID	- идентификатор BITMAP-слайда в файле ресурсов,
hInst	- hInstance библиотеки, в которой расположен слайд.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64. Для правильного формирования _variant_t HINSTANCE нужно передавать его через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо KompasObject::ksDrawBitmapEx.

ksDrawKompasDocument – Отрисовать документ системы КОМПАС как слайд в присланном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawKompasDocument.

Синтаксис Automation:

long ksDrawKompasDocument (long HWindow,

BSTR docFileName);

Входные параметры:

Hwindow	- несущее окно,
docFileName	- полное имя файла документа.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Действие метода распространяется также на фрагменты и модели из библиотек. При этом имя файла должно иметь вид "c:\gr\lib1.l3d|детали|литье|фланец",
где
c:\gr\lib1.l3d - имя файла библиотеки,

|детали|литье| - разделы, подразделы внутри библиотеки,

фланец - имя фрагмента или модели.

ksDrawKompasDocumentByReference – Отрисовать КОМПАС–документ как слайд в присланном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawKompasDocumentByReference.

Синтаксис Automation:

long ksDrawKompasDocumentByReference (long HWindow, long pDoc);

Входной параметр:

Hwindow	- несущее окно,
pDoc	- указатель на документ.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если pDoc = 0, отрисовывается текущий документ.

ksDrawKompasText – Отрисовать текст в формате КОМПАС в присланном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawKompasText.

Синтаксис Automation:

long ksDrawKompasText(long HWindow, BSTR text);

Входные параметры:

HWindow	- дескриптор окна для отрисовки слайда,
text	- текст.

ksEditTextLine – Вызвать диалог редактирования сложноструктурированного текста

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEditTextLine.

Синтаксис Automation:

BSTR ksEditTextLine (long HWindow, long* res, BSTR str);

Входные параметры:

Hwindow	- дескриптор окна,
str	- входная строка текста.

Выходные параметры:

res	- строка текста.

Возвращаемое значение:

1	- выход из диалога по кнопке **OK**,
0	- выход из диалога по кнопке **Отмена**.

ksExecDialPredefinedText – Получить предопределенный текст из файла текстовых шаблонов (с расширением tdp)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/538_65_9_6_Tekstovye_shablony.htm

Аналог данного метода при использовании API экспортных функций - ksExecDialPredefinedText.

Синтаксис Automation:

BSTR ksExecDialPredefinedText (long HWindow, long FAR* res);

Входной параметр:

Hwindow	- несущее окно.

Выходной параметр:

res	- результат работы функции: 1 - в случае успеха, 0 - в случае неудачи.

Возвращаемое значение:

указатель на интерфейс ksDynamicArray массива строк, тип массива - TEXT_LINE_ARR.	- в случае успеха,
NULL	- в случае неудачи.

Примечание:

Метод возвращает все строки. Они складываются в одну с переводом строки "\n"

ksExecDialPredefinedTextEx – Получить предопределенный текст из файла текстовых шаблонов (с расширением tdp)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/538_65_9_6_Tekstovye_shablony.htm

Аналог данного метода при использовании API экспортных функций - ksExecDialPredefinedTextEx.

Синтаксис Automation:

LPDISPATCH ksExecDialPredefinedTextEx (long HWindow);

Входной параметр:

Hwindow	- несущее окно.

Возвращаемое значение:

указатель на интерфейс ksDynamicArray массива строк, тип массива - TEXT_LINE_ARR.	- в случае успеха,
NULL	- в случае неудачи.

ksExecDialSymbol – Вызов диалога Вставка символа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksExecDialSymbol.

Синтаксис Automation:

BSTR ksExecDialSymbol(long HWindow, long * symb, BSTR font);

Входные параметры:

HWindow	- дескриптор окна,
symb	- номер символа, на котором будет стоять курсор,
font	- имя шрифта в диалоге.

Выходные параметры:

symb	- номер выбранного символа.

Возвращаемое значение:

Имя шрифта	- в случае успешного завершения,
NULL	- в случае неудачи или закрытии диалога по отмене.

ksExecDialSpecialSymbol – Вызов диалога Вставка спецзнака

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksExecDialSpecialSymbol.

Синтаксис Automation:

int ksExecDialSpecialSymbol(long HWindow);

Входные параметры:

HWindow	- дескриптор окна.

Возвращаемое значение:

Номер спецзнака	- в случае успешного завершения,
-1	- в случае неудачи или закрытии диалога по отмене.

ksExecuteKompasCommand – Выполнить команду системы КОМПАС

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksExecuteKompasCommand.

Синтаксис Automation:

BOOL ksExecuteKompasCommand (long commandID, BOOL post);

Входные параметры:

commandID	- константа из перечисления ProcessTypeEnum или ksKompasCommandEnum ,
post	- true - запуск команды через PostMessage, - false - через SendMessage.

Возвращаемое значение:

TRUE	- в случае удачи,

FALSE	- в случае ошибки.

Примечание:

1. Проверить доступность команды можно с помощью функции KompasObject::ksIsKompasCommandEnable.
2. Проверить, нажата ли в данный момент кнопка команды, можно с помощью KompasObject::ksIsKompasCommandCheck.

ksExecuteKompasLibraryCommand – Выполнить команду библиотеки

Интерфейс...

Синтаксис Automation:

long ksExecuteKompasLibraryCommand (long libId, long command);

Входные параметры:

libId	- уникальный номер библиотеки, полученный при выполнении ksAttachKompasLibrary,
command	- номер команды из библиотеки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

В отличии от функции ksExecuteLibraryCommand выполняется команда у подключенной ранее библиотеки. ksExecuteLibraryCommand - подключает библиотеку, выполняет команду, если защита позволяет и отключает библиотеку.

ksExecuteKompasLibraryCommandEx – Выполнить команду библиотеки

Интерфейс..

Синтаксис Automation:

long ksExecuteKompasLibraryCommandEx (long libId, long command, LPDISPATCH external);

Входные параметры:

libId	- уникальный номер библиотеки, полученный при выполнении ksAttachKompasLibrary,
command	- номер команды из библиотеки,
external	- указатель на интерфейс связи между библиотеками (приложениями).

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksExecuteLibraryCommand – Выполнить команду другой библиотеки

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksExecuteLibraryCommand.

Синтаксис Automation:

long ksExecuteLibraryCommand (BSTR fileName, long command);

Входные параметры:

fileName	- имя файла прикладной библиотеки,
command	- номер команды из библиотеки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если задано короткое имя файла прикладной библиотеки (без пути), то эта библиотека ищется в подпапке ..\Libs главной папки системы КОМПАС.

ksExecQualityDialog – Вызов диалога Выбор квалитета

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksExecQualityDialog.

Синтаксис Automation:

BOOL ksExecQualityDialog(long HWindow, BSTR curQual, double * dimValue, long inMM, LPDISPATCH param);

Входные параметры:

HWindow	- дескриптор окна,
curQual	- текущее значение квалитета,
dimValue	- значение для которого требуется выбрать квалитет или NULL,
inMM	- размерность параметров dimValue, high, low : 1 - миллиметры, 0 - единицы измерения текущего документа.

Выходной параметр:

param	- указатель на интерфейс структуры параметров квалитета ksQualityContensParam.

Возвращаемое значение:

TRUE	- в случае выбора квалитета,
FALSE	- отмены выбора или ошибки.

ksGetApplication7 – Получить указатель интерфейса приложения API версии 7

Интерфейс...

Аналог данного метода при использовании экспортных функций CreateKompasApplication.

Синтаксис Automation:

LPDISPATCH ksGetApplication7();

Возвращаемое значение:

	Указатель на интерфейс API версии 7 IApplication.

ksGetDocOptions – Получить настройки текущего документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SID_SETUP_HELP_ID.htm

Аналог данного метода при использовании API экспортных функций - GetDocOptions.

Синтаксис Automation:

long ksGetDocOptions (long optionsType, LPDISPATCH param);

Входной параметр:

optionsType	- тип настройки.

Выходной параметр:

param	- указатель на интерфейс параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

ksGetDocumentByReference – Получить интерфейс документа по указателю на документ

Интерфейс...

Синтаксис Automation:

LPDISPATCH ksGetDocumentByReference (long docRef);

Входной параметр:

docRef	- указатель на документ.

Возвращаемое значение:

Указатель на интерфейс документа	

Примечание:

Если docRef = 0, возвращается указатель на интерфейс IDispatch активного документа.

ksGetDocumentType – Получить тип документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetDocumentType.

Синтаксис Automation:

long ksGetDocumentType (long doc);

Входной параметр:

doc	- указатель на документ.

Возвращаемое значение:

тип документа DocType	- при успешном завершении,
0	- в случае неудачи.

Примечание:

Если указатель doc = 0, возвращается тип активного документа.

ksGetDocumentTypeByNameEx – Получить тип и специализацию документа по имени файла

Интерфейс...

Синтаксис Automation:

BSTR ksGetDocumentTypeByNameEx(BSTR fileName, long * docType, long * errorId);

Входные параметры:

fileName	- полное имя файла документа.

Выходные параметры:

docType	- тип документа DocType "Idefin2d.h",
errorId	- номер ошибки, если не удалось получить тип документа.

Возвращаемое значение:

	Специализация документа.

Примечание:

Возможные ошибки получения типа документа:

- etError64 64 - Документ не найден или неверная структура файла,
- etError217 217 - Документ создан более поздней версией,
- etError218 218 - Файл защищен с помощью приложения КОМПАС-Защита.

ksGetDocumentTypeByName – Получить тип документа DocType

Интерфейс...

Получить тип документа DocType

Аналог данного метода при использовании API экспортных функций - ksGetDocumentTypeByName.

Синтаксис Automation:

long ksGetDocumentTypeByName (BSTR fileName);

Входной параметр:

(BSTR) fileName	- полное имя файла документа.

Возвращаемое значение:

тип документа DocType "Idefin2d.h" 0	- при успешном завершении, - в случае неудачи.

Примечание:

1. Документ при вызове этой функции не создается.
2. Независимо от используемого оформления возвращается:

для чертежа	lt_DocSheetStandart
для спецификации	lt_DocSpc
для текстового документа	lt_DocTxtStandart

ksGetSelectedEmbodimentMarking – Вернуть обозначение исполнения, выбранное в диалоге выбора файла (ksSelectD3Model)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSelectedEmbodimentMarking.

Синтаксис Automation:

BSTR ksGetSelectedEmbodimentMarking();

Возвращаемое значение:

- обозначение исполнения.

Примечание:

1. В диалоге выбора файла KompasObject::ksSelectD3Model могут быть выбраны также дополнительные номера, если они были созданы в файле модели.
2. Получить выбранный дополнительный номер исполнения можно с помощью функции KompasObject::ksGetSelectedEmbodimentAdditionalNumber.

ksGetSelectedEmbodimentAdditionalNumber – Вернуть дополнительный номер исполнения, выбранный в диалоге выбора файла (ksSelectD3Model)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSelectedEmbodimentAdditionalNumber.

Синтаксис Automation:

BSTR ksGetSelectedEmbodimentAdditionalNumber();

Возвращаемое значение:

- дополнительный номер исполнения.

Примечание:

1. В диалоге выбора файла KompasObject::ksSelectD3Model могут быть выбраны также обозначения исполнения, если они были созданы в файле модели.
2. Получить выбранное обозначение исполнения можно с помощью функции KompasObject::ksGetSelectedEmbodimentMarking.

ksGetExternalInterface – Получить указатель внешнего интерфейса

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksGetExternalInterface.

Синтаксис Automation:

LPDISPATCH ksGetExternalInterface();

Возвращаемое значение:

- указатель на интерфейс IDispatch связи между библиотеками (приложениями).

Примечание:

1. Метод не выполняет Addref на выдаваемый интерфейс.
2. Внешний интерфейс можно использовать для обмена данными между двумя приложениями. Первое приложение передает интерфейс при вызове метода KompasObject::ksExecuteKompasLibraryCommandEx.
3. После выполнения KompasObject::ksExecuteKompasLibraryCommandEx метод ksGetExternalInterface возвращает NULL.

ksGetLibraryStylesArray – Получить указатель на интерфейс динамического массива стилей заданного типа, находящихся в заданной библиотеке стилей ksDynamicArray типа LIBRARY_STYLE_ARR

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WORK_WITH_COLLECTION_AND_LIB.htm

Аналог данного метода при использовании API экспортных функций - ksGetLibraryStylesArray,

Синтаксис Automation:

LPDISPATCH ksGetLibraryStylesArray (BSTR libraryName,

short libraryType);

Входные параметры:

libraryName	- полное имя библиотеки стилей,
libraryType	- тип библиотеки стиля.

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа LIBRARY_STYLE_ARR.

ksGetLibraryTreeStruct – Получить структуру дерева библиотеки документов и библиотеки атрибутов

Интерфейс...

Аналог данного метода при использовании экспортных функций ksGetLibraryTreeStruct.

Синтаксис Automation:

BOOL ksGetLibraryTreeStruct (BSTR libName, LPDISPATCH root);

Входной параметр:

libName	- полное имя файла библиотеки моделей, фрагментов, атрибутов,
root	- указатель на интерфейс ksTreeNodeParam параметров корневого узла дерева.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksGetObjectsFilter3D – Получить интерфейс фильтрации объектов-моделей

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/683_82_7_2_Filqtry_obwektov.htm

Аналог данного метода при использовании API экспортных функций - ksGetObjectsFilter3D.

Синтаксис Automation:

IObjectsFilter3D * GetObjectsFilter3D();

Возвращаемое значение:

- указатель на интерфейс ksObjectsFilter3D.

ksGetQualityDefects – Получить отклонения

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данного метода при использовании API экспортных функций - ksGetQualityDefects.

Синтаксис Automation:

long ksGetQualityDefects (BSTR name,

double dimValue,

double* high,

double* low,

short inMM);

Входные параметры:

name	- поле допуска,
dimValue	- значение размера,
inMM	- размерность параметров dimValue, high, low: 1 - миллиметры, 0 - единицы измерения текущего документа.

Выходные параметры:

high	- верхнее отклонение,
low	- нижнее отклонение.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksGetQualityNames – Получить указатель на динамический массив полей допусков ksDynamicArray,

которые поддерживают размер dimValue и не превышают указанных отклонений

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данного метода при использовании API экспортных функций - ksGetQualityNames.

Синтаксис Automation:

long ksGetQualityNames (LPDISPATCH names,

double dimValue,

double high,

double low,

short system,

short withLimitation);

Входные параметры:

names	- указатель на динамический массив ksDynamicArray типа CHAR_STR_ARR,
dimValue	- номинал,
high	- верхнее отклонение,
low	- нижнее отклонение,
system	- система: 1 - отверстия, 0 - вала,
withLimitation	- признак учета ограничений: 0 - без учёта ограничений, 1 - с учётом ограничений, наложенных в системе.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksGetQualityContensParam – Получить параметры квалитета

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данного метода при использовании API экспортных функций - ksGetQualityContensParam.

Синтаксис Automation:

long ksGetQualityContensParam (BSTR name, LPDISPATCH param, short inMM);

Входные параметры:

name	- поле допуска,
inMM	- размерность параметров minLimit, maxLimit, high, low: 1 - миллиметры, 0 - единицы измерения текущего документа.

Выходной параметр:

param	- указатель на интерфейс структуры параметров квалитета ksQualityContensParam.

Возвращаемое значение:

TRUE	- в случае выбора квалитета,
FALSE	- отмены выбора или ошибки.

ksGetSystemControlStartResult – Проверить запущен SystemControlStart или нет

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSystemControlStartResult.

Синтаксис Automation:

long ksGetSystemControlStartResult();

Входные параметры:

- не используется.

Возвращаемое значение:

тип выхода из режима работы под управлением системы КОМПАС.

ksGetSysOptions – Получить системные настройки

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_SAVECONFIG.htm

Аналог данного метода при использовании API экспортных функций - ksGetSysOptions.

Синтаксис Automation:

long ksGetSysOptions (long optionsType, LPDISPATCH param);

Входной параметр:

optionsType - тип настройки.

Выходные параметры:

param - указатель на интерфейс параметров, соответствующий настройке.

Возвращаемое значение:

1 - в случае удачного завершения,

0 - в случае неудачи.

Соответствие входных и выходных параметров...

ksGetWorkWindowColor – Получить цвет фона рабочего окна КОМПАС-ГРАФИК

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WINDOWCOLOR_SETUP.htm

Аналог данного метода при использовании API экспортных функций - ksGetWorkWindowColor.

Синтаксис Automation:

long ksGetWorkWindowColor();

Возвращаемое значение:

	- цвет фона рабочего окна.

ksGet3dDocumentFromRef – Получить указатель на интерфейс ksDocument3D, соответствующий присланному указателю на документ

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksGet3dDocumentFromReference.

Синтаксис Automation:

LPDISPATCH ksGet3dDocumentFromRef(long doc);

Входной параметр:

doc	- указатель на документ.

Возвращаемое значение:

указатель на интерфейс ksDocument3D	- в случае удачного завершения,
NULL	- если такого документа нет.

ksIsKompasCommandCheck – Проверить, нажата ли кнопка команды

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsKompasCommandCheck.

Синтаксис Automation:

int LIB_FUNC ksIsKompasCommandCheck (long commandID);

Входные параметры:

commandID	- константа из перечисления ProcessTypeEnum или ksKompasCommandEnum .

Возвращаемое значение:

1	- кнопка нажата,
0	- кнопка отжата.

ksIsKompasCommandEnable – Проверить доступность выполнения команды

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsKompasCommandEnable.

Синтаксис Automation:

BOOL ksIsKompasCommandEnable (long commandID);

Входные параметры:

commandID	- константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

TRUE	- команда доступна,
FALSE	- команда недоступна.

ksIsLibraryEnabled – Проверить защиту библиотеки фрагментов и моделей

Интерфейс..

Справка системы КОМПАС: Библиотеки элементов...

`KOMPAS.chm::/Application.htm#elem_lib`

Аналог данного метода при использовании API экспортных функций - ksIsLibraryEnabled.

Синтаксис Automation:

long ksIsLibraryEnabled(BSTR libName);

Входные параметры:

libName	- имя библиотеки либо полное, либо относительно папки LIBS.

Возвращаемое значение:

1	- выполнить библиотеку можно,
0	- выполнить библиотеку нельзя.

Примечание:

Функция применима для библиотек rtw, библиотек фрагментов lfr, библиотек моделей l3d

ksIsModule3DActive – Проверить, разрешена ли работа с модулем 3D

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsModule3DActive.

Синтаксис Automation:

long ksIsModule3DActive();

Возвращаемое значение:

1	- работа с модулем 3D разрешена,
0	- работа с модулем 3D не разрешена.

ksIsModuleSpecificationActive – Проверить, разрешена ли работа со спецификацией

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksIsModuleSpecificationActive.

Синтаксис Automation:

long ksIsModuleSpecificationActive();

Возвращаемое значение:

1	- работа со спецификацией разрешена,
0	- работа со спецификацией запрещена.

Примечание:

Если ksIsModuleSpecificationActive == FALSE, работа со спецификацией запрещена.

Для включения работы со спецификацией используйте метод KompasObject::ksModuleSpecification. Этот же метод позволяет отключить работу со спецификацией.

ksLockFileCache – Выключить/включить кеширование поиска файлов

Интерфейс...

Синтаксис Automation:

BOOL ksLockFileCache(BOOL lock);

Входные параметры:

lock	- TRUE - запретить кеширование, - FALSE - разрешить кеширование.

Возвращаемое значение:

Предыдущее значение флага

Примечание:

Кеширование включается при запуске библиотечной команды.

ksMaterialDlg – Проверить на получение и получить материал и его плотность из справочника материалов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMaterialDlg.

Синтаксис Automation:

BSTR ksMaterialDlg (long HWindow,

long* res,

double* plt,

double* kod_size_1,

double* kod_size_2,

double* kod_size_3,

double* kod_size_4,

BSTR kod_tip);

Входные параметры:

Hwindow	- дескриптор окна,

kod_tip	- строка кодов типов сортаментов для отображения (через запятую) (0 - отображается все).

Выходные параметры:

res	- результат работы функции: - 1 - справочник материалов не подключился, 0 - выход из диалога справочника материалов по отмене, 1- успешное завершение,
plt	- плотность выбранного материала,
kod_size_1	- код вида типоразмера: 1 - толщина, 2 - диаметр, 0 - вид не определен,
kod_size_2	- значение размера вида толщина, диаметр, диаметр вписанной окружности, значение A типоразмеров вида AxB или AxBxC,
kod_size_3	- значение B типоразмеров вида AxB или AxBxC,
kod_size_4	- значение C типоразмеров вида AxBxC.

Возвращаемое значение:

строка с обозначением материала	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Метод устарел и не используется. Предназначался для работы со справочником материалов, который в настоящее время не входит в комплект поставки.

ksModuleSpecification – Управление возможностью работы со спецификацией

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksModuleSpecification.

Синтаксис Automation:

long ksModuleSpecification (BOOL attach);

Входной параметр:

Attach	- признак выполняемого действия: 1 - включить работу со спецификацией, 0 - выключить работу со спецификацией.

Возвращаемое значение:

1	- если был инициирован процесс включения или отключения возможности работы со спецификацией,
0	- в случае неудачного завершения.

Примечание:

Проверить, включена или отключена возможность работы со спецификацией, можно с помощью метода KompasObject.

ksModule3D – Подключить 3D модуль для режима сетевой работы системы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksModule3D.

Синтаксис Automation:

long ksModule3D (BOOL attach);

Входные параметры:

attach	TRUE - включить работу с 3D модулем, FALSE - отключить.

Возвращаемое значение:

1	- если был инициирован процесс подключения или отключения 3D модуля,
0	- в случае неудачи.

ksPrintKompasDocument – Печать КОМПАС-документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/ID_FILE_PRINT.htm

Аналог данного метода при использовании API экспортных функций - ksPrintKompasDocument.

Синтаксис Automation:

long ksPrintKompasDocument (BSTR fileName,

BSTR toFile,

double scale);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
scale	- масштаб вывода.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksPrintKompasDocumentEx – Печать КОМПАС-документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/ID_FILE_PRINT.htm

Аналог данного метода при использовании API экспортных функций - ksPrintKompasDocumentEx.

Синтаксис Automation:

long ksPrintKompasDocument (BSTR fileName,
BSTR toFile,
double scale,
BOOL fKompasPrinter);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
scale	- масштаб вывода,
fKompasPrint	- TRUE используем принтер Компас, - FALSE - умолчательный принтер Windows.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Развитие метода KompasObject::ksPrintKompasDocument.

ksPrintPreviewWindow – Открыть окно предварительного просмотра документа перед печатью

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/1746_199_1_perehod_v_pred_prosmotr.htm

Аналог данного метода при использовании API экспортных функций - ksPrintPreviewWindow.

Синтаксис Automation:

long ksPrintPreviewWindow (LPDISPATCH docsArr, long inquiry);

Входные параметры:

docsArr	- указатель на интерфейс динамического массива ksDynamicArray типа CHAR_STR_ARR,
inquiry	- признак запроса документов: 1 - если docsArr = NULL или массив пуст, запросить документы у пользователя, 0 - показать документы без запроса.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksReadDouble – Запросить ввод вещественного числа с контролем попадания значения в заданный интервал

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ReadDouble.

Синтаксис Automation:

long ksReadDouble (BSTR mess,

double defValue,

double min,

double max,

double* value);

Входные параметры:

mess	- строка приглашения,
defValue	- предлагаемое значение по умолчанию,
min, max	- интервал возможных значений.

Выходной параметр:

value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения.

ksReadInt – Запросить ввод целого числа с контролем попадания значения в заданный интервал

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ReadInt.

Синтаксис Automation:

long ksReadInt (BSTR mess,

long defValue,

long min,

long max,

long* value);

Входные параметры:

mess - строка приглашения,
defValue - предлагаемое значение по умолчанию,
min, max - интервал возможных значений.

Выходной параметр:

value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения.

ksReadString – Запросить ввод строки

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ReadString.

Синтаксис Automation:

BSTR ksReadString (BSTR mess,

BSTR defValue);

Входные параметры:

mess	- строка приглашения,
defValue	- предлагаемое значение по умолчанию.

Возвращаемое значение:

- результат ввода.

ksRefreshActiveWindow – Обновить активное окно документа

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksRefreshActiveWindow.

Синтаксис Automation:

long ksRefreshActiveWindow();

Возвращаемое значение:

1	- в случае удачного завершения.
0	- в случае неудачи.

ksSetDebugMessagesMode – Включить/выключить режим автоматического вывода сообщений о результатах работы библиотеки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetDebugMessagesMode.

Синтаксис Automation:

BOOL ksSetDebugMessagesMode(BOOL debugMode);

Возвращаемое значение:

- предыдущее состояние флага режима.

ksSelectD3Model – Выбрать из списка открытых или из файла модели

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSelectD3Model.

Синтаксис Automation:

BSTR ksSelectD3Model(BOOL onlyDetail, BOOL showAddNum);

Возвращаемое значение:

- имя файла,

Входные параметры:

onlyDetail	- только файлы деталей,
showAddNum	- отображать комбобокс выбора дополнительного номера.

Примечание:

1. В диалоге выбора файла могут быть выбраны также обозначение исполнения и дополнительный номер исполнения, если они были созданы в файле модели.
2. Получить выбранное обозначение исполнения можно с помощью метода KompasObject::ksGetSelectedEmbodimentMarking.
3. Получить выбранный дополнительный номер исполнения можно с помощью метода KompasObject::ksGetSelectedEmbodimentAdditionalNumber.

ksSetDocOptions – Задать тип настройки текущего документа

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SID_SETUP_HELP_ID.htm`

Аналог данного метода при использовании API экспортных функций - SetDocOptions.

Синтаксис Automation:

long ksSetDocOptions (long optionsType, LPDISPATCH param);

Входной параметр:

optionsType	- тип настройки.

Выходной параметр:

param	- указатель на интерфейс параметров.

Возвращаемое значение:

1	- в случае удачного завершения,

0	- в случае неудачи.

Соответствие входных и выходных параметров...

ksSetSysOptions – Установить системные настройки

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_SAVECONFIG.htm

Аналог данного метода при использовании API экспортных функций - ksSetSysOptions.

Синтаксис Automation:

long ksSetSysOptions (long optionsType, LPDISPATCH param);

Входные параметры:

optionsType	- тип настройки.

Выходные параметры:

param	- указатель на интерфейс параметров, соответствующий настройке.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

ksViewGetDensity – Выбрать из диалога плотность

Интерфейс..

Справка системы КОМПАС...

KOMPAS.chm::/dlg_mix_density.htm

Аналог данного метода при использовании API экспортных функций - ksViewGetDensity.

Синтаксис Automation:

double ksViewGetDensity (long HWindow);

Входной параметр:

HWindow	- дескриптор "родительского" окна или NULL.

Возвращаемое значение:

плотность (г/куб.мм)	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Плотность выбирается из системного файла плотностей graphic.dns.

ksViewGetDensityAndMaterial – Выбрать из диалога плотность и наименование материала

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/dlg_mix_density.htm`

Аналог данного метода при использовании API экспортных функций - ksViewGetDensityAndMaterial.

Синтаксис Automation:

BSTR ksViewGetDensityAndMaterial (double *density, long HWindow);

Входные параметры:

Hwindow	- дескриптор окна.

Выходные параметры:

density	- плотность (г/куб.мм)

Возвращаемое значение:

Строка с обозначением материала	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Плотность выбирается из системного файла плотностей graphic.dns.

LoadDSK – Загрузить dsk КОМПАС

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1061_127_1_3_Fajly_konfiguracii.htm

Синтаксис Automation:

BOOL LoadDSK();

Возвращаемое значение:

TRUE	- в случае удачного открытия dsk,
FALSE	dsk не открыт, возможно открыт ранее.

Примечание:

При открытии dsk будут подгружены документы и библиотеки, которые были подключены в последнем сеансе работы КОМПАС.

Quit – Закрыть приложение

Интерфейс...

Синтаксис Automation:

void Quit();

Входные параметры:

- не используется.

Возвращаемое значение:

- не используется.

SpcActiveDocument – Получить указатель на интерфейс для текущего документа-спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH SpcActiveDocument();

Возвращаемое значение:

- указатель на интерфейс ksSpcDocument.

Примечание:

Если документ спецификации не активен, функция возвращает NULL.

SpcDocument – Получить указатель на интерфейс документа-спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH SpcDocument();

Возвращаемое значение:

- указатель на интерфейс ksSpcDocument.

Методы вывода на экран

ksDrawBitmap – Отрисовать растровый слайд

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DrawBitmap.

Синтаксис Automation:

long ksDrawBitmap (long HWindow,

long slideId);

Входные параметры:

HWindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,
slideId	- идентификатор BITMAP-слайда в файле ресурсов.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksDrawBitmapEx – Отрисовать растровый слайд в заданном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawBitmapEx.

Синтаксис Automation:

long ksDrawBitmapEx (long HWindow, long sldID, long hInst);

Входные параметры:

Hwindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,
sldID	- идентификатор BITMAP-слайда в файле ресурсов,
hInst	- hInstance текущей библиотеки или NULL, если она одна.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64.

Рекомендуется использовать функцию KompasObject::ksDrawBitmapEx2.

ksDrawSlide – Отрисовать слайд в заданном окне

Интерфейс..

Аналог данного метода при использовании API экспортных функций - DrawSlide.

Синтаксис Automation:

long ksDrawSlide (long HWindow, long slideId);

Входные параметры:

HWindow	- дескриптор окна,
slideId	- идентификатор слайда.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

Слайд должен быть описан в ресурсном файле библиотеки при помощи объекта RCDATA.

ksDrawSlideEx – Отрисовать указанный слайд в заданном окне

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksDrawSlideEx.

Синтаксис Automation:

long ksDrawSlideEx (long HWindow,

long slideId,

long hInst);

Входные параметры:

HWindow	- дескриптор окна,
slideId	- идентификатор слайда,
hInst	- hInstance текущей библиотеки или NULL, если она одна.

Возвращаемое значение:

1	- в случае успешного завершения.

Примечание:

Слайд должен быть описан в ресурсном файле библиотеки при помощи объекта RCDATA.

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64. Рекомендуется использовать функцию KompasObject::ksDrawSlideEx2.

ksDrawSlideEx2 – Отрисовать указанный слайд в заданном окне

Интерфейс..

Аналог данного метода при использовании API экспортных функций ksDrawSlideEx.

Синтаксис Automation:

long ksDrawSlideEx2(long HWindow, long sldID, VARIANT hInst);

Входные параметры:

HWindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,

sldID	- идентификатор BITMAP-слайда в файле ресурсов,
hInst	- hInstance библиотеки, в которой расположен слайд.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Слайд должен быть описан в ресурсном файле библиотеки при помощи объекта RCDATA. HINSTANCE библиотеки нужно передать через VARIANT как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо KompasObject::ksDrawSlideEx.

ksDrawSlideFromFile – Отрисовать слайд в окне из текстового файла, содержащего блок RCDATA

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksDrawSlideFromFile.

Синтаксис Automation:

long ksDrawSlideFromFile (long HWindow,

BSTR fileName);

Входные параметры:

HWindow	- дескриптор окна,
fileName	- полное имя файла.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание :

Функция предназначена для отладки слайдов. Позволяет просмотреть отредактированный слайд в окне диалога без перетрансляции тестовой библиотеки.

ksError – Выдать сообщение об ошибке

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Error.

Синтаксис Automation:

BOOL ksError(BSTR message);

Входной параметр:

message	- строка с текстом сообщения об ошибке.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksGetHWindow – Получить дескриптор главного окна КОМПАС-ГРАФИК

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetHWindow.

Синтаксис Automation:

long ksGetHWindow();

Возвращаемое значение:

- дескриптор главного окна, который используется функциями WINDOWS для работы с окнами.

ksMessage – Выдать сообщение

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Message.

Синтаксис Automation:

BOOL ksMessage (BSTR message);

Входной параметр:

message	- строка с текстом сообщения.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksMessageBoxResult – Вывести сообщение, соответствующее результату работы библиотеки (с кодом ошибки)

Интерфейс..

Вывести сообщение, соответствующее результату работы библиотеки (с кодом ошибки).

Аналог данного метода при использовании API экспортных функций - MessageBoxResult.

Синтаксис Automation:

BOOL ksMessageBoxResult();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция сбрасывает флаг ошибки в случае, если эта ошибка не является фатальной.

ksReturnResult – Получить результат работы библиотеки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ReturnResult.

Синтаксис Automation:

long ksReturnResult();

Возвращаемое значение:

	- Код ошибки в зависимости от типа документа: графического или документа-модели при выполнении библиотечной программы.

Примечание:

1. Коды ошибок документа-модели и графического документа частично совпадают, поэтому их нужно обрабатывать в зависимости от выполняемых операций.
2. Ошибка с номером >0 не является фатальной. Отрицательный номер ошибки приводит к завершению программы.
3. Текст ошибок можно получить методом KompasObject::ksStrResult.
4. Сообщение с текстом ошибки можно получить, используя метод KompasObject_ksMessageBoxResult.
5. Сбросить ошибку, если она не фатальная, можно, используя метод KompasObject_ksResultNULL.

ksSlideBackground – Установить цвет фона по умолчанию для отрисовки слайда

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksSlideBackground.

Синтаксис Automation:

BOOL ksSlideBackground (long color);

Входной параметр:

color	- цвет фона слайда.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

ksStrResult – Получить строку сообщения, соответствующую результату работы библиотеки (с кодом ошибки)

Интерфейс..

Аналог данного метода при использовании API экспортных функций - StrResult.

Синтаксис Automation:

BSTR ksStrResult();

Возвращаемое значение:

- строка сообщения, соответствующая результату работы библиотеки.

Примечание:

Функция сбрасывает флаг ошибки в случае, если эта ошибка не является фатальной.

ksWriteSlide – Записать выделенные объекты чертежа в формате векторного слайда КОМПАС

Интерфейс..

Аналог данного метода при использовании API экспортных функций - WriteSlide.

Синтаксис Automation:

long ksWriteSlide (BSTR fileName,

long slideId,

double x,

double y);

Входные параметры:

filename	- имя файла для записи,
slideId	- идентификатор слайда в файле ресурсов,
x, y	- координаты базовой точки.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание :

Функция выводит выделенные объекты изображения чертежа в формат векторного слайда КОМПАС.

Документ должен быть открыт с отображением на экране.

ksYesNo – Подтвердить действие или отказаться от него

Интерфейс..

Аналог данного метода при использовании API экспортных функций - YesNo.

Синтаксис Automation:

long ksYesNo (BSTR s);

Входной параметр:

s	- строка сообщения.

Возвращаемое значение:

1	- при подтверждении действия,
0	- при отказе,
-1	- при отмене действия.

Примечание:

Подтверждение означает нажатие в диалоге кнопки **Да** (**Yes**, **OK)**; отказ - нажатие кнопки **Нет** (**No**); отмена действия - нажатие кнопки **Отмена** (**Cansel**).

Работа с файлами

ksChoiceFile – Показать диалог и выбрать в нем имя файла для чтения

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksChoiceFile.

Синтаксис Automation:

BSTR ksChoiceFile (BSTR ext,

BSTR filter,

BOOL preview);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - формируется автоматически),
preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Возвращаемое значение:

строка с именем файла	- в случае удачного завершения,
NULL	- в случае отказа от выбора.

ksChoiceFileAppointedDir – Выдать папку и выбрать имя файла для открытия

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksChoiceFileAppointedDir.

Синтаксис Automation:

BSTR ksChoiceFileAppointedDir (BSTR ext,

BSTR filter,

BOOL preview,

long typeDir);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - формируется автоматически),
preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна,
typeDir	стартовая папка.

Возвращаемое значение:

- строка с именем файла.

Примечание.

Параметр typeDir может иметь значения sptSYSTEM_FILES и sptLIBS_FILES; во всех остальных случаях открывается текущая папка.

ksChoiceFiles – Выдать диалог выбора файлов для открытия

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksChoiceFiles.

Синтаксис Automation:

long ksChoiceFiles (BSTR ext,

BSTR filter,

LPDISPATCH p,

BOOL preview);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - формируется автоматически),
preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходной параметр:

p	- указатель на интерфейс динамического массива ksDynamicArray типа CHAR_STR_ARR.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksChoiceFilesW – Выдать диалог выбора файлов для чтения (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFiles.

Синтаксис:

int LIB_FUNK ksChoiceFileW (LPWSTR ext,

LPWSTR filter,

reference p,

unsigned char preview);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - формируется автоматически),

preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходной параметр:

p	- указатель на динамический массив строк CHAR_STR_ARR или CHAR_STR_ARR_W.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksChoiceFiles.

ksSaveFile – Выдать диалог выбора файла для сохранения

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksSaveFile.

Синтаксис Automation:

BSTR ksSaveFile (BSTR ext,

BSTR oldName,

BSTR filter,

BOOL preview);

Входные параметры:

ext	- расширение имени файла,
oldName	- имя файла по умолчанию,
filter	- фильтр поиска (0 - формируется автоматически),

preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Возвращаемое значение:

- строка с именем файла.

Сервисные функции

ksEnableTaskAccess – Разрешить или запретить доступ к задаче со стороны пользователя

Интерфейс...

Аналог данного метода при использовании API экспортных функций - EnableTaskAccess.

Синтаксис Automation:

BOOL ksEnableTaskAccess(long enable);

Входной параметр:

enable	- признак разрешения доступа к задаче со стороны пользователя: 1 - доступ разрешен, 0 - доступ запрещен.

Возвращаемое значение:

TRUE

Примечание:

См. также KompasObject::ksIsEnableTaskAccess.

ksFullFileName – Проверить имя файла и, если оно не полное, добавить к нему имя текущей папки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - FullFileName.

Синтаксис Automation:

BSTR ksFullFileName (BSTR oldName);

Входной параметр:

oldName	- старое имя файла.

Возвращаемое значение:

- полное имя файла.

ksGetFullPathFromRelativePath – Сформировать полный путь к файлу из заданного пути к задающему файлу и относительного пути к файлу

Интерфейс..

Аналог данного метода при использовании API экспортных функций - GetFullPathFromRelativePath.

Синтаксис Automation:

BSTR ksGetFullPathFromRelativePath (BSTR mainFilePath, BSTR relativePath);

Входные параметры:

mainFilePath	- полный путь к задающему файлу,
relativePath	- относительный путь к требуемому файлу (без общей с задающим файлом части пути).

Возвращаемое значение:

- полный путь к файлу.

ksGetFullPathFromSystemPath – Сформировать полный путь к файлу из заданного относительного пути к файлу и системного пути установленного типа

Интерфейс..

Аналог данного метода при использовании API экспортных функций - ksGetFullPathFromSystemPath.

Синтаксис Automation:

BSTR ksGetFullPathFromSystemPath (BSTR relativePath, long pathType);

Входные параметры:

relativePath	- относительный путь к файлу,
pathType	- тип системной папки.

Возвращаемое значение:

- полный путь к файлу.

ksGetRelativePathFromFullPath – Сформировать относительный путь к файлу из полного пути к задающему файлу и полного пути к файлу

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetRelativePathFromFullPath.

Синтаксис Automation:

BSTR ksGetRelativePathFromFullPath (BSTR mainFilePath,

BSTR sourcePath);

Входные параметры:

mainFilePath	- полный путь к задающему файлу,
sourcePath	- полный путь к требуемому файлу.

Возвращаемое значение:

- относительный путь к файлу.

ksGetRelativePathFromSystemPath – Сформировать относительный путь к файлу из заданного полного пути к файлу и системного пути установленного типа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetRelativePathFromSystemPath.

Синтаксис Automation:

BSTR ksGetRelativePathFromSystemPath (BSTR sourcePath, long pathType);

Входные параметры:

sourcePath	- полный путь к файлу,
pathType	- тип системной папки.

Возвращаемое значение:

- относительный путь к файлу.

ksGetSystemVersion – Получить номер версии системы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSystemVersion.

Синтаксис Automation:

long ksGetSystemVersion (long* iMajor,

long* iMinor,

long* iRelease,

long* iBuild);

Выходные параметры:

iMajor	- старшее слово версии,
iMinor	- младшее слово версии,
iRelease	- номер выпуска внутри одной версии,
iBuild	- номер сборки внутри одного выпуска.

Возвращаемое значение:

1	- в случае удачного завершения.

Пример:

Для версии 5.4 Release 2 Build 1

iMajor = 5

iMinor = 4

iRelease = 2

iBuild = 1

Примечание:

Любой из указателей может быть равен 0, тогда соответствующее ему значение не выдается.

ksGetSystemProfileString – Получить строку из файла инициализации системы или из реестра

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSystemProfileString.

Синтаксис Automation:

BSTR ksGetSystemProfileString (BSTR lpSection,

BSTR lpKey);

Входные параметры:

lpSection	- имя секции,
lpKey	- имя ключа.

Возвращаемое значение:

строка из файла инициализации системы (kompasw.ini) или из реестра (Registry)	- в случае успеха,
NULL	- если секция или ключ не найдены.

Пример:

lpSection = "Directories"

lpKey = "Sys"

Результат: lpReturnedString = "c:\Program Files\Kompas54\Sys"

ksIsEnableTaskAccess – Определить, разрешен ли доступ к задаче со стороны пользователя

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IsEnableTaskAccess.

Синтаксис Automation:

long ksIsEnableTaskAccess();

Возвращаемое значение:

0	- доступ к задаче со стороны пользователя запрещен,
1	- доступ к задаче со стороны пользователя разрешен.

Примечание:

См. также KompasObject::ksEnableTaskAccess.

ksOpenHelpFile – Открыть файл справки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksOpenHelpFile.

Синтаксис Automation:

BOOL ksOpenHelpFile (BSTR file, long command, long id);

Входные параметры:

file	- полное имя файла Справки,
command	- тип Справки (контекстная, по команде и т.д.),
id	- идентификатор страницы справки.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание :

1. Синтаксис соответствует WinHelp.
2. Функция открывает файл Справки КОМПАС.
3. При закрытии системы окно Справки закроется автоматически.

ksPumpWaitingMessages – Обработать все сообщения, имеющиеся в очереди сообщений задачи

Интерфейс..

Аналог данного метода при использовании API экспортных функций - PumpWaitingMessages.

Синтаксис Automation:

BOOL ksPumpWaitingMessages();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksRemoveUniqueFile – Удалить уникальный служебный файл

Интерфейс..

Аналог данного метода при использовании API экспортных функций - RemoveUniqueFile.

Синтаксис Automation:

BOOL ksRemoveUniqueFile (BSTR fileName);

Входной параметр:

fileName	- имя служебного файла.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Удалить служебный файл с уникальным именем, предварительно полученным с помощью функции ksUniqueFileName.

ksResultNULL – Обнулить результат работы библиотеки, если ошибка не фатальная

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ResultNULL.

Синтаксис Automation:

long ksResultNULL();

Возвращаемое значение:

0	- если ошибок не обнаружено,
1	- если ошибка была обнулена.

ksSetCriticalProcess – Установить критический процесс

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetCriticalProcess.

Синтаксис Automation:

long ksSetCriticalProcess();

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

До завершения ksCursor, ksPlacement или конца библиотечной команды системе не разрешается принудительно завершать работу библиотеки.

ksSystemControlStart – Перейти под управление КОМПАС-ГРАФИК

Интерфейс..

Аналог данного метода при использовании API экспортных функций - SystemControlStart.

Синтаксис Automation:

long ksSystemControlStart (BSTR menuCommand);

Входной параметр:

menuCommand	- строка, помещаемая в меню для возврата в библиотеку.

Возвращаемое значение:

	тип выхода из режима работы под управлением системы КОМПАС.

Примечание:

В результате вызова этого метода управление передается системе КОМПАС. В случае выполнения в системе КОМПАС одного из действий, описываемых возвращаемым значением, управление вновь передается библиотеке, и она должна соответствующим образом отреагировать на внешнее воздействие. Например, если пользователь вызовет команду закрытия КОМПАС, библиотека должна прекратить свои действия.

ksSystemControlStop – Отдать управление библиотеке

Интерфейс..

Аналог данного метода при использовании API экспортных функций - SystemControlStop

Синтаксис Automation:

BOOL ksSystemControlStop();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Функция принудительно передает управление библиотечной функции, запущенной в режиме немодального диалога (в этом случае управление может переходить от библиотеки к системе и наоборот).

ksSystemPath – Получить системный путь установленного типа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_PATHVIEW.htm

Аналог данного метода при использовании API экспортных функций - ksSystemPath.

Синтаксис Automation:

BSTR ksSystemPath(long folderType);

Входной параметр:

folderType	- тип запрашиваемой системной папки.

Возвращаемое значение:

- строка с запрашиваемым путем.

ksUniqueFileName – Получить уникальное имя файла

Интерфейс...

Аналог данного метода при использовании API экспортных функций - UniqueFileName.

Синтаксис Automation:

BSTR ksUniqueFileName();

Возвращаемое значение:

- строка с именем файла.

Примечание:

Получить уникальное имя для создания служебного файла в процессе работы библиотеки. Имя регистрируется и запоминается системой. По окончании работы в случае, если файл с таким именем существует, он будет автоматически удален.

TransferInterface – Преобразовать интерфейсный объект одного типа API в интерфейсный объект API другого типа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTransferInterface.

Тип данных: LPUNKNOWN

Синтаксис Automation:

LPUNKNOWN TransferInterface (LPUNKNOWN obj,
long newApiType,
long objNewType);

Входные параметры:

obj	- интерфейс 3D COM, API5 Auto или API7,
apiNewType	- тип API, к которому преобразуется исходный интерфейс, может принимать значения из ksAPITypeEnum,
objNewType	- тип объекта в интерфейсе, к которому преобразуется исходный объект (может быть задан 0).

Возвращаемое значение:

	- указатель на интерфейс объекта в API заданного типа.

Описание:

Метод позволяет получить интерфейс объекта заданного типа objNewType в API заданного типа apiNewType по присланному объекту obj.

Примечание:

1. Исходным объектом может быть интерфейс 3D COM, API5 Auto или API7.
2. Если новый тип API совпадает с API присланного объекта, возвращается присланный объект.
3. Если задан новый тип API как неопределенный (0), возвращается присланный объект.
4. Если задан новый тип API значением вне допустимого множества значений, возвращается NULL.
5. Если задан новый тип объекта значением вне допустимого множества значений, возвращается NULL.
6. Если задан новый тип объекта неопределенным значением (0), возвращается интерфейс базового объекта, совпадающий по типу с требуемым объектом, либо требуемый объект может быть получен от базового через QueryInterface; рекомендуется явно указывать требуемый тип объекта.

TransferReference – Преобразовать объект по reference из API5 в интерфейсный объект API7

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTransferReference.

Тип данных LPUNKNOWN

Синтаксис Automation:

LPUNKNOWN ksTransferReference(long obj, long doc);

Входные параметры:

obj	- указатель на объект в API5,
doc	- указатель на документ, где находится объект.

Возвращаемое значение:

- указатель на интерфейс объекта в API7 (Тип данных:LPUNKNOWN).

Описание:

Функция позволяет получить интерфейс объекта в API7 по присланному указателю obj в API5.

Примечание:

1. Функция реализована пока для документов и объектов вида графического документа. В остальных случаях возвращается NULL.
2. Если obj - указатель на документ, параметр doc игнорируется.
3. Если параметр doc = 0, берется текущий документ.

Документ – модель (Интерфейсы – ksDocument3D, IDocument3D)

Интерфейс документа-модели.

Интерфейс событий документа-модели; работа с файлом...

ksDocument3D	- интерфейс Automation
IDocument3D	- интерфейс COM

Примечание:

1. Данный интерфейс в автоматизации может быть получен от интерфейса API Kompas KompasObject с помощью методов KompasObject::Document3D, KompasObject::ActiveDocument3D;
2. Данный интерфейс в COM может быть получен с помощью экспортных функций: ksGet3dDocument, ksGetActive3dDocument.

ksDocument3D, IDocument3D – свойства

author – Имя автора документа

Интерфейс...

Справка системы КОМПАС

KOMPAS.chm::/DOCUMENT_INFO_DIALOG.htm

Тип данных: строка

Синтаксис Automation:

author = document.author	Получить свойство (*)
document.author = author	Установить свойство (*)
author = document.GetAuthor ()	Получить свойство (**)
document.SetAuthor(author)	Установить свойство (**)

Синтаксис COM:

author = document->GetAuthor ()	Получить свойство
document->SetAuthor(author)	Установить свойство

comment – Комментарий документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DOCUMENT_INFO_DIALOG.htm

Тип данных: строка

Синтаксис Automation:

comment = document.comment	Получить свойство (*)
document.comment = comment	Установить свойство (*)
comment = document.GetComment ()	Получить свойство (**)
document.SetComment (comment)	Установить свойство (**)

Синтаксис COM:

comment = document->GetComment ()	Получить свойство
document->SetComment (comment)	Установить свойство

dis mantleMode – Разнесенный вид

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_3D_REPLACE_COMPONENTS_PARAMETERS.htm

Тип данных: BOOL

Значения свойства:

TRUE	- разнесенный вид включен,
FALSE	- разнесенный вид выключен,

Синтаксис:

dismantleMode = iDocument3D.dismantleMode	Получить свойство(*)
iDocument3D.dismantleMode = dismantleMode	Установить свойство (*)
dismantleMode = iDocument3D.GetDismantleMode()	Получить свойство (**)
iDocument3D.SetDismantleMode (dismantleMode)	Установить свойство (**)

drawMode – Тип отображения модели

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/679_82_5_Otobrazhenie_modeli.htm

Тип данных: long

Значения свойства:

Из перечисления ViewMode

Синтаксис Automation:

drawMode = iDocument3D.drawMode	Получить свойство (*)
iDocument3D.drawMode = drawMode	Установить свойство (*)
drawMode = iDocument3D.GetDrawMode()	Получить свойство(**)
iDocument3D.SetDrawMode(drawMode)	Установить свойство (**)

enableRollBackFeaturesInCollections – Выдавать в коллекциях объекты, исключенные указателем в дереве

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/667_81_3_6_Ukazatelq_okonchanij.htm

Тип данных: BOOL

Синтаксис Automation:

enable = iDocument3D.enableRollBackFeaturesInCollections	Получить свойство (*)
iDocument3D.enableRollBackFeaturesInCollections = enable	Установить свойство (*)
enable = document.GetenableRollBackFeaturesInCollections()	Получить свойство(**)
document.SetenableRollBackFeaturesInCollections(enable)	Установить свойство(**)

Примечание:

По умолчанию объекты, исключенные с помощью указателя (зеленая полоска) дерева, не выдаются. Свойство используется для получения исключенных объектов дерева, которые можно использовать для изменения положения указателя в дереве.

См. функции ksDocument3D::GetRollBackFeature и ksDocument3D::SetRollBackFeature.

fileName – Имя файла документа (трехмерной модели)

Интерфейс...

Тип данных: строка

Синтаксис Automation:

fileName = document.fileName	Получить свойство (*)
document.fileName = fileName	Установить свойство (*)
fileName = document.GetFileName()	Получить свойство (**)
document.SetFileName(fileName)	Установить свойство (**)

Синтаксис COM:

fileName = document->GetFileName()	Получить свойство
document->SetFileName(fileName)	Установить свойство

invisibleMode – Режим редактирования документа

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- невидимый режим,
FALSE	- видимый режим.

Синтаксис Automation:

InvisibleMode = document.invisibleMode	Получить свойство (*)
InvisibleMode = document.GetInvisibleMode ()	Получить свойство (**)

Синтаксис COM:

InvisibleMode = document->GetInvisibleMode ()		Получить свойство

Примечание:

Данное свойство предназначено только для чтения.

hideAllAuxiliaryGeom – Скрыть / показать все вспомогательные объекты

Интерфейс...

Тип данных: BOOL

Синтаксис:

hideAllAuxiliaryGeom = Object.hideAllAuxiliaryGeom	=	Получить свойство (*)
Object.hideAllAuxiliaryGeom = hideAllAuxiliaryGeom	=	Установить свойство (*)
hideAllAuxiliaryGeom = Object.GetHideAllAuxiliaryGeom( )	=	Получить свойство (**)
Object.SetHideAllAuxiliaryGeom(hideAllAuxiliaryGeom)		Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideAllAxis – Скрыть\показать конструктивные оси

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.htm

Тип данных: BOOL

Значения свойства:

TRUE	- скрыть конструктивные оси,
FALSE	- показать конструктивные оси.

Синтаксис Automation:

hideAllAxis = iDocument3D.hideAllAxis	Получить свойство(*)
iDocument3D.hideAllAxis = hideAllAxis	Установить свойство(*)
hideAllAxis = iDocument3D.GetHideAllAxis()	Получить свойство(**)
iDocument3D.SetHideAllAxis(hideAllAxis)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

HideAllControlPoints – Скрыть / показать контрольные точки

Интерфейс...

Справка системы КОМПАС

`KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.htm`

Тип данных: BOOL

Значения свойства:

TRUE	- контрольные точки скрыты,
FALSE	- контрольные очки показаны.

Синтаксис COM:

BOOL HideAllControlPoints = iObject->GetHideAllControlPoints()	Получить свойство
iObject->SetHideAllControlPoints (HideAllControlPoints)	Установить свойство

Примечание.

Свойство только для COM-интерфейса IDocument3D.

HideAllCurves – Скрыть / показать пространственные кривые

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.htm

Тип данных: BOOL

Значения свойства:

TRUE	- пространственные кривые скрыты,
FALSE	- пространственные кривые показаны.

Синтаксис:

BOOL HideAllCurves = iObject->GetHideAllCurves()	Получить свойство
iObject->SetHideAllCurves (HideAllCurves)	Установить свойство

Примечание.

Свойство только для COM-интерфейса IDocument3D.

hideAllDimensions – Скрыть / показать размеры

Интерфейс...

Тип данных: BOOL

Синтаксис:

hideAllDimensions = Object.hideAllDimensions	Получить свойство (*)
Object.hideAllDimensions = hideAllDimensions	Установить свойство (*)
hideAllDimensions = Object.GetHideAllDimensions()	Получить свойство (**)
Object.SetHideAllDimensions(hideAllDimensions)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать размеры в документе или, если документ является сборкой, во вставках.

hideAllDesignations – Скрыть / показать условные обозначения

Интерфейс...

Тип данных: BOOL

Синтаксис:

hideAllDesignations = Object.hideAllDesignations	Получить свойство (*)
Object.hideAllDesignations = hideAllDesignations	Установить свойство(*)
hideAllDesignations = Object.GetHideAllDesignations()	Получить свойство (**)
Object.SetHideAllDesignations(hideAllDesignations)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать условные обозначения в документе или, если документ является сборкой, во вставках.

hideAllPlaces – Скрыть\показать начала координат

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.htm`

Тип данных: BOOL

Значения свойства:

TRUE	- скрыть начала координат,
FALSE	- показать начала координат.

Синтаксис Automation:

iDocument3D.hideAllPlaces	Получить свойство (*)
iDocument3D.hideAllPlaces = hideAllPlaces	Установить свойство (*)
hideAllPlaces = iDocument3D.GetHideAllPlaces()	Получить свойство (**)

iDocument3D.SetHideAllPlaces(hideAllPlaces)	Установить свойство(**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideAllPlanes – Скрыть\показать конструктивные плоскости

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.HTM`

Тип данных: BOOL

Значения свойства:

TRUE	- скрыть конструктивные плоскости,
FALSE	- показать конструктивные плоскости.

Синтаксис Automation:

hideAllPlanes = iDocument3D.hideAllPlanes	Получить свойство (*)
iDocument3D.hideAllPlanes = hideAllPlanes	Установить свойство (*)
hideAllPlanes = iDocument3D.GetHideAllPlanes()	Получить свойство (**)
iDocument3D.SetHideAllPlanes(hideAllPlanes)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideAllSketches – Скрыть\показать эскизы

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.HTM

Тип данных: BOOL

Значения свойства:

TRUE	- скрыть конструктивные эскизы,
FALSE	- показать конструктивные эскизы.

Синтаксис Automation:

iDocument3D.hideAllSketches = hideAllSketches		Получить свойство (*)
iDocument3D.hideAllSketches		Установить свойство (*)
hideAllSketches = iDocument3D.GetHideAllSketches()		Получить свойство (**)
iDocument3D.SetHideAllSketches(hideAllSketches)		Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideAllSurfaces – Скрыть\показать поверхности

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.HTM

Тип данных: BOOL

Значения свойства:

TRUE	- скрыть поверхности,
FALSE	- показать поверхности.

Синтаксис Automation:

hideAllSurfaces = iDocument3D.hideAllSurfaces	Получить свойство (*)
iDocument3D.hideAllSurfaces = hideAllSurfaces	Установить свойство (*)

hideAllSurfaces = iDocument3D.GetHideAllSurfaces()	Получить свойство (**)
iDocument3D.SetHideAllSurfaces(hideAllSurfaces)	Установить свойство(**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideAllThreads – Скрыть\показать обозначения резьбы

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/IDD_SETUP_LOD_OTHERS.HTM

Тип данных; BOOL

Значения свойства:

TRUE	- скрыть поверхности,
FALSE	- показать поверхности.

Синтаксис Automation:

hideAllThreads = iDocument3D.hideAllThreads	Получить свойство (*)
iDocument3D.hideAllThreads = hideAllThreads	Установить свойство (*)
hideAllThreads = iDocument3D.GetHideAllThreads()	Получить свойство (**)
iDocument3D.SetHideAllThreads(hideAllThreads)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать все вспомогательные объекты в документе или, если документ является сборкой, во вставках.

hideInComponentsMode – Режим скрытия вспомогательной геометрии в компонентах

Интерфейс...

Тип данных: BOOL

Синтаксис:

hideInComponentsMode = Object.hideInComponentsMode	Получить свойство (*)
Object.hideInComponentsMode = hideInComponentsMode	Установить свойство (*)
hideInComponentsMode = Object.GetHideInComponentsMode()	Получить свойство(**)
Object.SetHideInComponentsMode(hideInComponentsMode)	Установить свойство(**)

Примечание:

Флаг позволяет для сборки включить режим работы с флагами скрытия вспомогательной геометрии во вставках или в документе.

Если флаг включен, то свойства: hideAllPlanes, hideAllAxis, hideAllSketches, hideAllPlaces, hideAllSurfaces, hideAllThreads, hideAllCurves, hideAllControlPoints, hideAllDimensions, hideAllDesignations, hideAllAuxiliaryGeom управляют признаками скрытия вспомогательной геометрии во вставках.

hideLayoutGeometry – Скрыть / показать компоновочную геометрию

Интерфейс...

Тип данных: BOOL

Синтаксис:

hideLayoutGeometry = Object.hideLayoutGeometry	Получить свойство (*)
Object.hideLayoutGeometry = hideLayoutGeometry	Установить свойство (*)
hideLayoutGeometry = Object.GetHideLayoutGeometry()	Получить свойство (**)
Object.SetHideLayoutGeometry(hideLayoutGeometry)	Установить свойство (**)

Примечание:

В зависимости от флага hideInComponentsMode позволяет скрыть или показать компоновочную геометрию в документе или, если документ является сборкой, во вставке.

perspective – Признак отображения модели в перспективной проекции

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/679_82_5_Otobrazhenie_modeli.htm

Тип данных: BOOL

Значения свойства:

TRUE	- перспективная проекция включена,
FALSE	- перспективная проекция выключена.

Синтаксис Automation:

perspective = iDocument3D.perspective	Получить свойство (*)
iDocument3D.perspective = perspective	Установить свойство (*)
perspective = iDocument3D.GetPerspective()	Получить свойство (**)
iDocument3D.SetPerspective(perspective)	Установить свойство (**)

reference – Указатель документа-модели (детали, сборки)

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iDocument3D.reference	Получить свойство (*)
iDocument3D.reference = ref	Установить свойство(*)
ref = document.GetReference ()	Получить свойство (**)
document.SetReference (ref)	Установить свойство (**)

Примечание:

Свойство определяет указатель на трехмерную модель в стиле API графических документов. Позволяет использовать функционал атрибутов и итераторов.

shadedWireframe – Полутоновое изображение с каркасом объектов активного окна документа-модели

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/679_82_5_Otobrazhenie_modeli.htm

Тип данных: BOOL

Значения свойства:

TRUE	- полутоновое изображение с каркасом объектов включено,
FALSE	- полутоновое изображение с каркасом объектов выключено.

Синтаксис:

shadedWireframe = iDocument3D.shadedWireframe	Получить свойство (*)
iDocument3D.shadedWireframe = shadedWireframe	Установить свойство(*)
shadedWireframe = iDocument3D.GetShadedWireframe()	Получить свойство (**)
iDocument3D.SetShadedWireframe(shadedWireframe)	Установить свойство (**)

treeNeedRebuild – Необходимость перестроения дерева при изменении его состава

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/964_117_12_Preduprezhdenija_o_n.htm

Тип данных: BOOL

Значения свойства:

TRUE	- разрешить перестраивать дерево,

FALSE	- запретить перестраивать дерево.

Синтаксис:

treeNeedRebuild = iDocument3D.treeNeedRebuild	Получить свойство(*)
iDocument3D.treeNeedRebuild = treeNeedRebuild	Установить свойство (*)
treeNeedRebuild = iDocument3D.GetTreeNeedRebuild()	Получить свойство (**)
iDocument3D.SetTreeNeedRebuild (treeNeedRebuild)	Установить свойство (**)

Примечания:

1. Так как после вызова ksEntity::Update у любого объекта модель будет перестраиваться, то для ускорения при перестроении группы объектов, кроме закрытия самого перестроения ksPart::needRebuild, можно также закрыть отрисовку дерева, если она не нужна.
2. При использовании данного свойства нужно запоминать его значение и после окончания работы с ним возвращать его в первоначальное состояние.

windowNeedRebuild – Свойство необходимости перерисовки окна документа

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- надо перерисовывать,
FALSE	- не надо перерисовывать.

Синтаксис:

windowNeedRebuild = object.windowNeedRebuild	=	Получить свойство (*)
object.windowNeedRebuild = windowNeedRebuild	=	Установить свойство (*)
windowNeedRebuild = object.GetWindowNeedRebuild()	=	Получить свойство (**)
object.SetWindowNeedRebuild(windowNeedRebuild)		Установить свойство (**)

Примечание:

Свойство позволяет временно отключить перерисовку окна документа.

Используется для ускорения создания, удаления или изменения группы объектов.

ksDocument3D, IDocument3D – методы

AdditionFormatParam – Получить указатель на интерфейс параметров сохранения модели в дополнительном формате

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/990_123_2_Ehksport.htm

Синтаксис Automation:

LPDISPATCH AdditionFormatParam();

Синтаксис COM:

LPADDITIONFORMATPARAM AdditionFormatParam();

Возвращаемое значение:

- указатель на интерфейс ksAdditionFormatParam или IAdditionFormatParam.

AddImportedSurfaces – Добавить импортированные поверхности

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

AddImportedSurfaces (BSTR fileName, BOOL together)

Синтаксис COM:

LPENTITYCOLLECTION AddImportedSurfaces (LPOLESTR fileName, BOOL together)

Входные параметры:

fileName	- имя файла (*.igs, *.sat) импортированной поверхности,
together	- Если параметр имеет значение TRUE, то импортированные поверхности будут сшиваться; если параметр имеет значение FALSE, то будет импортирован массив импортированных поверхностей.

Возвращаемое значение:

Указатель на интерфейс ksEntityCollection или IEntityCollection,	- в случае успеха,
NULL	- в случае неудачи.

Примечания:

В документ можно вставить импортированную поверхность из файла, причем в общем случае может импортироваться массив импортированных поверхностей, если together = FALSE.

AddMateConstraint – Добавить сопряжение в сборку

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/1024_112_1_Obshchije_sved_sopr.htm`

Синтаксис Automation:

BOOL AddMateConstraint (long constraintType,

LPDISPATCH object1,

LPDISPATCH object2,

short direction,

short fixed,

double value);

Синтаксис COM:

BOOL AddMateConstraint (long constraintType,

LPENTITY obj1,

LPENTITY obj2,

short direction,

short fixed,

double val);

Входные параметры:

constraintType	- тип сопряжения из перечисления MateConstraintType,
object1	- указатель на интерфейс первого объекта, на который накладывается сопряжение (ksEntity или IEntity),
object2	- указатель на интерфейс второго объекта, на который накладывается сопряжение (ksEntity или iEntity),

direction	- ориентация (1 - объекты однонаправленные, 0 - направление не учитывается, -1 - объекты разнонаправленные),
fixed	- признак фиксации деталей перед выполнением (0 - детали не фиксируются, 1 - фиксируется первая деталь, 2 - фиксируется вторая деталь),
val	- параметр для ограничений (расстояние или угол между объектами).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Направление задаётся знаком параметра val.

AttributeCollection – Получить массив атрибутов

Интерфейс...

Справка системы КОМПАС: атрибуты объекта...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Справка системы КОМПАС:атрибуты документа...

`KOMPAS.chm::/514_63_1_3_Atribut_dokumenta.htm`

Синтаксис Automation:

ksAttribute3DCollection* AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb,
LPDISPATCH pObj);

Синтаксис COM:

LPATTRIBUTE3DCOLLECTION AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb,
LPUNKNOWN pObj);

Входные параметры:

key1, key2, key3, key4	- ключи для поиска по ключам либо 0,

numb	- номер типа атрибута для поиска по номеру либо 0,
pObj	- указатель на ksDocument3D - атрибуты документа, ksFeatureCollection - атрибуты для данной группы объектов, ksFeature - атрибуты определенного объекта.

Возвращаемое значение:

	- указатель на массив атрибутов ksAttribute3DCollection.

Примечания:

Атрибут - это дополнительная неграфическая информация, связанная с объектом или несколькими объектами чертежа. Такая информация может быть представлена в виде числа, строки текста, а также таблицы с фиксированным или переменным числом строк.

В дальнейшем значения атрибутов могут использоваться для поиска объектов, а также обрабатываться различными приложениями (например, системой проектирования спецификаций, различными расчетными программами и т.п.).

Типы (описания структуры) атрибутов могут храниться как непосредственно в документе, так и в специальных файлах (библиотеках типов атрибутов). Эти файлы имеют расширение lat.

ChangeObjectInLibRequest – Изменить фантом или компоненты команд

Интерфейс...

Синтаксис Automation:

BOOL ChangeObjectInLibRequest();

Синтаксис COM:

BOOL ChangeObjectInLibRequest();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Вызов этого метода позволяет принудительно, не из CallBack-функции изменить параметры текущего процесса. Предварительно, используя интерфейс ksRequestInfo3d, можно изменить:

- состав команд командного окна,
- строку-приглашение,

- идентификатор или имя курсора,
- заголовок командного окна.

Все сделанные изменения будут активизированы сразу после вызова данной функции. Процесс должен быть предварительно запущен функцией UserGetPlacementAndEntity.

Close – Закрыть документ-модель (деталь или сборку)

Интерфейс...

Синтаксис Automation:

BOOL Close();

Синтаксис COM:

BOOL Close();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ComponentPositioner – Получить указатель на интерфейс управления положением компонентов в сборке

Интерфейс...

Синтаксис Automation:

ksComponentPositioner * ComponentPositioner();

Синтаксис COM:

LPCOMPONENTPOSITIONER ComponentPositioner();

Возвращаемое значение:

	- указатель на интерфейс ksComponentPositioner.

CopyPart – Создать копию заданного компонента с заданным положением

Интерфейс...

Синтаксис Automation:

LPDISPATCH CopyPart (LPDISPATCH sourcePart, LPDISPATCH newPlacement);

Синтаксис COM:

LPPART CopyPart (LPPART sourcePart, LPPLACEMENT newPlacement);

Входные параметры:

sourcePart	- указатель на интерфейс исходного компонента ksPart- или IPart, подлежащего копированию,
newPlacement	- указатель на интерфейс положения создаваемой копии ksPlacement- или IPlacement.

Возвращаемое значение:

	- указатель на интерфейс компонента ksPart или IPart.

Примечание:

Если не задано положение копии компонента, она создается в начале координат.

Create – Создать документ-модель (деталь или сборку)

Интерфейс...

Синтаксис Automation:

BOOL Create (BOOL invisible, BOOL typeDoc);

Синтаксис COM:

BOOL Create (BOOL invisible, BOOL _typeDoc);

Входные параметры:

invisible	- признак режима редактирования документа (TRUE - невидимый режим, FALSE - видимый режим),
typeDoc	- тип документа (TRUE - деталь, FALSE - сборка).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

CreatePartInAssembly – Получить указатель на интерфейс детали, создаваемой в сборке

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_3D_ADD_COMPONENT_PART.htm

Синтаксис Automation:

LPDISPATCH CreatePartInAssembly (BSTR fileName,

LPDISPATCH plane);

Синтаксис COM:

LPPART CreatePartInAssembly (LPOLESTR fileName,

LPENTITY plane);

Входные параметры:

fileName	- имя файла детали создаваемой в сборке,
plane	- указатель на интерфейс плоского объекта ksEntity или IEntity, к которому приклеивается деталь.

Возвращаемое значение:

- указатель на интерфейс детали, создаваемой в сборке ksPart или IPart.

DefaultPlacement – Получить указатель на интерфейс умолчательной системы координат

Интерфейс...

Синтаксис Automation:

ksPlacement * DefaultPlacement();

Синтаксис COM:

LPPLACEMENT DefaultPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement.

DeleteObject – Удалить объект: деталь, операцию, сопряжение

Интерфейс...

Синтаксис Automation:

BOOL DeleteObject (LPDISPATCH obj);

Синтаксис COM:

BOOL DeleteObject (LPUNKNOWN obj);

Входной параметр:

obj	- указатель на интерфейс IDispatch или IUnknown удаляемого объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечания:

1. Удалить можно любой трехмерный объект.
2. Если в качестве входного параметра переданы:

- поверхность ksFaceDefinition,
- ребро ksEdgeDefinition,
- вершина ksVertexDefinition,

   то будет удален весь формообразующий элемент, которому принадлежит удаляемый объект.

3. Если в качестве входного параметра передано описание объекта, полученное методом ksEntity::GetDefinition, то после удаления недопустимо использование интерфейса описания удаленного объекта, породившего его интерфейса ksEntity и других интерфейсов удаленного объекта.

EntityCollection – Получить указатель на интерфейс динамического массива объектов заданного типа, выбранных в документе

Интерфейс...

Синтаксис Automation:

LPDISPATCH EntityCollection (short objType, BOOL checkEntity);

Синтаксис COM:

LPENTITYCOLLECTION EntityCollection (short objType,BOOL checkEntity);

Входные параметры:

objType	- тип объектов, содержащихся в массиве,
checkEntity	- признак проверки для вновь добавляемых объектов на NULL (TRUE - проверка включена, FALSE - проверка выключена и в массив можно добавить NULL).

Возвращаемое значение:

- указатель на интерфейс динамического массива объектов ksEntityCollection или IEntityCollection.

Примечание:

Массив создается пустым. Метод Refresh для полученного массива не работает.

ExcludeFeaturesAfter– Исключить\включить объекты после заданного

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/CM_EXCLUDE_AFTER_SEL_OBJECT.htm`

Синтаксис Automation:

BOOL ExcludeFeaturesAfter(LPDISPATCH obj,
BOOL exclude);

Синтаксис COM:

BOOL ExcludeFeaturesAfter(LPFEATURE obj,
BOOL exclude);

Входные параметры:

obj	- указатель на интерфейс объекта дерева ksFeature или IFeature после которого нужно исключить из расчета объекты,
exclude	- TRUE - выключить объекты, - FALSE - включить объекты.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetChooseMng – Получить указатель на интерфейс менеджера выбора (подсветки) объектов

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetChooseMng();

Синтаксис COM:

LPCHOOSEMNG GetChooseMng();

Возвращаемое значение:

- указатель на интерфейс ksChooseMng или IChooseMng.

GetDocument3DNotify – Получить указатель на интерфейс источника событий для документа-модели

Интерфейс...

Синтаксис Automation:

Document3DNotify* GetDocument3DNotify();

Возвращаемое значение:

Указатель на источник событий Document3DNotify - в случае успеха.

GetDocument3DNotifyResult – Дополнительные параметры для событий документа 3D

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDocument3DNotifyResult();

Синтаксис COM:

IDocument3DNotifyResult * GetDocument3DNotifyResult();

Возвращаемое значение:

Указатель на интерфейс дополнительных параметров для события 3D документа IDocument3DNotifyResult, ksDocument3DNotifyResult

Примечание:

Интерфейс позволяет получить указатель на объект для которого посылается данное событие. Например, выбор материала и обозначения доступны для верхнего компонента, вставки и тела.

GetEditMacroObject – Получить редактируемый макроэлемент 3D

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetEditMacroObject();

Синтаксис COM:

LPENTITY GetEditMacroObject();

Возвращаемое значение:

- Указатель на интерфейс редактируемого макроэлемента ksEntity.

GetInterface – Получить интерфейс 3D объекта по типу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetInterface (long o3dType);

Синтаксис COM:

LPUNKNOWN GetInterface (long o3dType);

Входные параметры:

o3dType	- тип интерфейса.

Возвращаемое значение:

Указатель на заданный интерфейс	- в случае успеха,
NULL	- в случае неудачи.

Список интерфейсов, возвращаемых функцией GetInterface и функцией ksDocument3D::GetInterface.

GetLastFeature – Получить последний объект в дереве

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetLastFeature()

Синтаксис COM :

LPFEATURE GetLastFeature()

Возвращаемое значение:

Указатель на объект дерева ksFeature или IFeature	- в случае успеха,
NULL	- в случае неудачи.

GetMateConstraint – Получить указатель на интерфейс нового временного сопряжения

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMateConstraint();

Синтаксис COM:

LPMATECONSTRAINT GetMateConstraint();

Возвращаемое значение:

- указатель на интерфейс нового сопряжения ksMateConstraint или IMateConstraint.

Примечания:

1. Сопряжения бывают постоянными и временными. Временные сопряжения, в отличие от постоянных, существуют только в течении процесса указания местоположения или объектов. Процесс запускается методом ksDocument3D::UserGetPlacementAndEntity. Постоянные сопряжения создаются с помощью метода ksDocument3D::AddMateConstraint.
2. Для создания временного сопряжения в модели после получения указателя на его интерфейс с помощью данного метода и изменения параметров создаваемого сопряжения используется метод ksMateConstraint::Create .
3. Пример использования методов создания временных и постоянных сопряжений приведен в демонстрационной библиотеке построения модели стандартного изделия (шпильки), которая сохранена в файле STUDS3D.rtw.

GetObjParam – Получить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjParam.

Синтаксис Automation:

long ksGetObjParam (long reference, LPDISPATCH param, long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров,
param	- указатель на интерфейс параметров ksUserParam.

Возвращаемое значение:

тип объекта	- в случае удачного завершения,

0	- в случае неудачи.

GetObjectType – Получить тип объекта

Интерфейс...

Синтаксис Automation:

long GetObjectType (LPDISPATCH obj);

Синтаксис COM:

int GetObjectType (LPUNKNOWN obj);

Входной параметр:

obj	- указатель на интерфейс IDispatch или IUnknown трехмерного объекта.

Возвращаемое значение:

	- тип объекта Obj3dType.

Примечания:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному трехмерному интерфейсу. Зная тип объекта, можно привести его к соответствующему интерфейсу.

GetPart – Получить указатель на интерфейс компонента в соответствии с заданным типом

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPart (long type);

Синтаксис COM:

LPPART GetPart(int type);

Входной параметр:

type	- тип компонента из перечисления Типы компонентов.

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Примечание:

Тип задается для нового, редактируемого или главного компонента либо тип равен номеру компонента в документе.

Функция используется, чтобы получить доступ к компоненту документа.

Деталь или сборка являются компонентами. Сборка, в свою очередь, состоит из компонентов - деталей и подсборок.

GetRequestInfo – Получить указатель интерфейса параметров запроса к системе ksRequestInfo3D

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetRequestInfo (LPDISPATCH part);

Синтаксис COM:

LPREQUESTINFO GetRequestInfo (LPPART part);

Входной параметр:

part - указатель на интерфейс компонента (детали или подсборки) ksPart или IPart.

Возвращаемое значение:

- указатель на интерфейс параметров запроса к системе ksRequestInfo3D или IRequestInfo.

GetRollBackFeature – Получить положение указателя в дереве

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/667_81_3_6_Ukazatelq_okonchanij.htm`

Синтаксис Automation:

LPDISPATCH GetRollBackFeature( );

Синтаксис COM:

LPFEATURE GetRollBackFeature();

Возвращаемое значение:

- указатель на объект дерева ksFeature или IFeature, после которого установлен указатель.	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetSelectionMng – Получить указатель на интерфейс менеджера выделенных объектов

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSelectionMng();

Синтаксис COM:

LPSELECTIONMNG GetSelectionMng();

Возвращаемое значение:

указатель на интерфейс ksSelectionMng или ISelectionMng.

GetSpecification – Получить указатель на интерфейс для работы с объектами спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSpecification();

Возвращаемое значение:

- указатель на интерфейс для работы с объектами спецификации ksSpecification.

GetViewProjectionCollection – Получить указатель на интерфейс массива проекций отображения модели в окне

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/674_82_4_Orientacija_modeli.htm

Синтаксис Automation:

LPDISPATCH GetViewProjectionCollection();

Синтаксис COM:

LPVIEWPROJECTIONCOLLECTION GetViewProjectionCollection();

Возвращаемое значение:

	- указатель на интерфейс ksViewProjectionCollection или IViewProjectionCollection.

IsActive – Получить состояние активности документа

Интерфейс...

Синтаксис Automation:

BOOL IsActive();

Синтаксис COM:

BOOL IsActive();

Возвращаемое значение:

TRUE	- документ является активным.

IsDetail – Определить, является ли модель деталью

Интерфейс...

Синтаксис Automation:

BOOL IsDetail();

Синтаксис COM:

BOOL IsDetail();

Возвращаемое значение:

TRUE	- деталь,
FALSE	- сборка.

IsEditMode – Определить, запущен ли какой-либо из компонентов на редактирование через библиотеку

Интерфейс...

Синтаксис Automation:

BOOL IsEditMode();

Синтаксис COM:

BOOL IsEditMode();

Возвращаемое значение:

TRUE	- редактирование компонента через библиотеку,
FALSE	- редактируемых компонентов нет.

Примечание:

Редактируемый компонент можно получить методом GetPart (pEdit_Part).

ksDeleteObj – Удалить объект

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteObj.

Синтаксис Automation:

long ksDeleteObj (long ref);

Синтаксис COM:

long ksDeleteObj (long ref);

Входные параметры:

ref	- указатель на удаляемый объект.

Возвращаемое значение:

1	- в случае успешного завершения.

ksGetObjParam – Получить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjParam.

Синтаксис Automation:

long ksGetObjParam (long reference, LPDISPATCH param, long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров объекта или индекс строки в массиве для объектов TECHNICALDEMAND_OBJ и TEXT_OBJ или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ),
Param	- указатель на интерфейс параметров ksUserParam.

Возвращаемое значение:

Тип объекта	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Вызов метода с нулевыми значениями параметров param и parType позволяет получить тип объекта по его reference.

ksIsSlaveSpcOpened – Открыто ли окно спецификации в Slave режиме

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1155_132_3_Podchinennyj_rezhim.htm

Синтаксис Automation:

int ksIsSlaveSpcOpened();

Синтаксис COM:

int ksIsSlaveSpcOpened();

Возвращаемое значение:

0	- окно спецификации в Slave режиме не открыто,
1	- окно спецификации в Slave режиме открыто,
2	- окно спецификации в Slave режиме открыто и активно.

ksSetObjParam – Установить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetObjParam.

Синтаксис Automation:

long ksSetObjParam (long reference, LPDISPATCH param, long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров объекта или индекс строки в массиве для объектов TECHNICALDEMAND_OBJ и TEXT_OBJ или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ),
Param	- указатель на интерфейс параметров ksUserParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

LoadFromAdditionFormat – Загрузить документ из дополнительных форматов jgs, sat, xt, x_b, step, stl, jt, C3D

Интерфейс...

Синтаксис Automation:

BOOL LoadFromAdditionFormat(BSTR fileName, LPDISPATCH additionPar);

Синтаксис COM:

BOOL LoadFromAdditionFormat(LPOLESTR fileName, LPADDITIONFORMATPARAM additionPar);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

fileName	-имя файла,
additionPar	- указатель на интерфейс ksAdditionFormatParam или IAdditionFormatParam, определяющий параметры записи в дополнительный формат,

MateConstraintCollection – Получить интерфейс параметров массива сопряжений компонента сборки

Интерфейс...

Синтаксис Automation:

LPDISPATCH MateConstraintCollection();

Синтаксис COM:

LPMATECONSTRAINTCOLLECTION MateConstraintCollection();

Возвращаемое значение:

Указатель на интерфейс коллекции сопряжений текущей редактируемой сборки или подсборки
IMateConstraintCollection
ksMateConstraintCollection

Open – Открыть на редактирование документ-модель (деталь или сборку)

Интерфейс...

Синтаксис Automation:

BOOL Open (BSTR fileName, BOOL invisible);

Синтаксис COM:

BOOL Open (LPOLESTR fileName, BOOL invisible);

Входные параметры:

fileName	- имя файла открываемого документа,
invisible	- признак режима редактирования документа (TRUE – невидимый режим, FALSE - видимый режим).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

PartCollection – Получить указатель на интерфейс динамического массива компонентов, вставленных в сборку

Интерфейс...

Синтаксис Automation:

LPDISPATCH PartCollection (BOOL refresh)

Синтаксис COM:

LPPARTCOLLECTION PartCollection (BOOL refresh);

Входные параметры:

refresh	- TRUE - полученный массив будет заполнен компонентами данной сборки или детали, FALSE - полученный массив будет пустым.

Возвращаемое значение:

	- указатель на интерфейс динамического массива компонентов, вставленных в сборку ksPartCollection или IPartCollection.

PlaceFeatureAfter – Поставить объект дерева после другого объекта дерева

Интерфейс...

Синтаксис Automation:

BOOL PlaceFeatureAfter(LPDISPATCH obj, LPDISPATCH afterObj);

Синтаксис COM:

BOOL PlaceFeatureAfter(LPFEATURE obj, LPFEATURE afterObj);

Входные параметры:

obj	- указатель на перемещаемый объект дерева ksFeature или IFeature,

afterObj	- указатель на объект дерева, после которого нужно установить заданный объект ksFeature или IFeature,

Возвращаемое значение:

TRUE FALSE	- в случае успешного завершения, - в случае неудачи.

RasterFormatParam – Получить указатель на интерфейс параметров сохранения документа в растровом формате

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm`

Синтаксис Automation:

LPDISPATCH RasterFormatParam();

Синтаксис COM:

LPRASTERFORMATPARAM RasterFormatParam();

Возвращаемое значение:

- указатель на интерфейс ksRasterFormatParam или IRasterFormatParam.

RebuildDocument – Перестроить документ-модель

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_3D_REBUILD_ALL.htm`

Синтаксис Automation:

BOOL RebuildDocument();

Возвращаемое значение:

TRUE FALSE	- в случае успешного завершения, - в случае неудачи.

RemoveMateConstraint – Удалить сопряжение из сборки

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1024_112_1_Obshchije_sved_sopr.htm

Синтаксис Automation:

BOOL RemoveMateConstraint (long constraintType,

LPDISPATCH object1,

LPDISPATCH object2);

Синтаксис COM:

BOOL RemoveMateConstraint (unsigned int constraintType,

LPENTITY obj1,

LPENTITY obj2);

Входные параметры:

constraintType	- тип сопряжения из перечисления MateConstraintType,
object1	- указатель на интерфейс первого объекта, на который наложено сопряжение (ksEntity или IEntity),
object2	- указатель на интерфейс второго объекта, на который наложено сопряжение (ksEntity или ksEntity).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

RunTakeCreateObjectProc – Запустить подчиненный режим создания объектов

Интерфейс...

Синтаксис Automation:

BOOL RunTakeCreateObjectProc(long processType, LPDISPATCH takeObject, BOOL needCreateTakeObj, BOOL lostTakeObj)

Синтаксис COM:

BOOL RunTakeCreateObjectProc(long processType, LPUNKNOWN takeObject, BOOL needCreateTakeObj, BOOL lostTakeObj);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Выходные параметры:

ProcessType	- тип подпроцесса,
TakeObject	- указатель на интерфейс редактируемого в подпроцессе объекта.
NeedCreateTakeObj	- необходимость создания объекта в подпроцессе
LostTakeObj-	- TRUE - фиксировать параметры создаваемого объекта.

Функция позволяет запустить следующие подпроцессы

prPoint3D	Точка 3D
prAxisByDirection	Ось через вершину по объекту
prCAxis2Points	Ось по двум точкам
prCAxis2Planes	Ось по двум плоскостям
prCAxisConeface	Ось конической грани
prCAxisEdge	Ось, проходящая через ребро
prContour3D	Контур 3D
prLocalCoordinateSystem	Локальная система координат
prIsoparamCurve	Изопараметрическая кривая
prEquidistant3D	Эквидистанта 3D
prCPlaneOffset	Смещённая плоскость
prCPlane3Points	Плоскость по 3-м точкам
prCPlaneAngle	Плоскость под углом
prCPlaneEdgePoint	Плоскость через ребро и вершину
prCPlaneParallel	Плоскость через вершину параллельно другой плоскости
prCPlanePerpendicular	Плоскость через вершину перпендикулярно ребру
prCPlaneNormalToSurface	Нормальная плоскость
prCPlaneTangentToSurface	Касательная плоскость
prCPlaneLineToEdge	Плоскость через ребро параллельно/ перпендикулярно другому ребру
prCPlaneLineToFlat	Плоскость через ребро параллельно/ перпендикулярно грани
prCPlaneMiddle	Средняя плоскость
prCPlaneTangentAtPoint	Плоскость, касательная к грани в точке
prCPlaneAtCurve	Плоскость через плоскую кривую
prMateParallel	Сопряжения компонентов - Параллельность
prMatePerpendicular	Сопряжения компонентов - Перпендикулярность
prMateOnDistance	Сопряжения компонентов - На расстоянии
prMateOnAngle	Сопряжения компонентов - Под углом
prMateTangent	Сопряжения компонентов - Касание
prMateConcentric	Сопряжения компонентов - Соосность
prMateCoincident	Сопряжения компонентов - Совпадение
prMateSymmetry	Сопряжения компонентов - Симметрия
prMateDependent	Сопряжения компонентов - Зависимое положение

Save – Сохранить документ-модель (деталь или сборку) в файл с заданным ранее именем

Интерфейс...

Синтаксис Automation:

BOOL Save();

Синтаксис COM:

BOOL Save();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Имя файла документа задается свойством fileName.

SaveAs – Сохранить документ-модель (деталь или сборку) с новым именем

Интерфейс...

Синтаксис Automation:

BOOL SaveAs (BSTR fileName);

Синтаксис COM:

BOOL SaveAs (LPOLESTR fileName);

Входной параметр:

fileName	- имя файла документа.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SaveAsEx – Сохранить документ с новым именем файла

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm`

Синтаксис Automation:

BOOL SaveAsEx (BSTR fileName, long saveMode);

Синтаксис COM:

BOOL SaveAsEx (LPOLESTR fileName, long saveMode);

Входные параметры:

fileName	- новое имя файла документа,
saveMode	- версия для сохранения: -1 - в предыдущую версию, 0 - в текущую версию, 1 - в версию 5.11.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Если fileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.

SaveAsToAdditionFormat – Сохранить модель в файле формата SAT, XT, STEP, IGES, VRML, STL

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/990_123_2_Ehksport.htm`

Синтаксис Automation:

BOOL SaveAsToAdditionFormat (BSTR fileName,

LPDISPATCH additionPar);

Синтаксис COM:

BOOL SaveAsToAdditionFormat (LPOLESTR fileName,

LPADDITIONFORMATPARAM additionPar);

Входные параметры:

fileName	- полное имя файла, в который сохраняется модель,
additionPar	- указатель на интерфейс ksAdditionFormatParam или IAdditionFormatParam, определяющий параметры записи в дополнительный формат.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SaveAsToRasterFormat – Сохранить документ в растровом файле

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Синтаксис Automation:

BOOL SaveAsToRasterFormat (BSTR fileName,

LPDISPATCH rasterPar);

Синтаксис COM:

BOOL SaveAsToRasterFormat (LPOLESTR fileName,

LPRASTERFORMATPARAM rasterPar);

Входные параметры:

fileName	- полное имя файла, в который сохраняется модель,
rasterPar	- указатель на интерфейс ksRasterFormatParam или IRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

SaveAsToUncompressedRasterFormat – Сохранить документ в растровый формат

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Синтаксис Automation:

BOOL SaveAsToUncompressedRasterFormat (BSTR fileName, LPDISPATCH rasterPar);

Синтаксис COM:

BOOL SaveAsToUncompressedRasterFormat (LPOLESTR fileName, LPRASTERFORMATPARAM rasterPar);

Выходные параметры:

fileName	- Имя файла документа. Задается полное имя файла,
rasterPar	- Указатель на интерфейс ksRasterFormatParam или IRasterFormatParam, определяющий параметры конвертации в растровый формат.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи

Примечания:

1. Метод позволяет сохранить документ в растровый формат (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) без сжатия, с заданными свойствами.
   Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

   Подробнее см. описание интерфейса ksRasterFormatParam. Для API экспортных функций - метод ksSaveAsToUncompressedRasterFormat.

SetActive – Сделать документ текущим

Интерфейс...

Синтаксис Automation:

BOOL SetActive();

Синтаксис COM:

BOOL SetActive();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetPartFromFile – Вставить в модель компонент ссылкой на внешний файл или телом

Интерфейс...

Синтаксис Automation:

BOOL SetPartFromFile (BSTR fileName,

LPDISPATCH part,

BOOL externalFile);

Синтаксис COM:

BOOL SetPartFromFile (LPOLESTR fileName,

LPPART part,

BOOL externalFile);

Входные параметры:

fileName	- имя файла, из которого будет вставлен компонент,
part	- указатель на интерфейс компонента, который будет вставлен в документ (ksPart или IPart),
externalFile	- признак сохранения связи с файлом-источником (TRUE - вставка со ссылкой на внешний файл, FALSE - вставка "телом", без сохранения ссылки на источник).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

При вставке из файла указывается полный путь к файлу. При вставке из библиотеки моделей указывается полный путь к файлу библиотеки и путь внутри библиотеки моделей (например, "C:\\standard.l3d|Крепежные элементы|Болты|Болт_1").

SetPartFromFileEx – Вставить в модель компонент из файла или из библиотеки моделей

Интерфейс...

Синтаксис Automation:

BOOL SetPartFromFileEx (BSTR fileName, LPDISPATCH part, BOOL externalFile, BOOL redraw);

Синтаксис COM:

BOOL SetPartFromFileEx (LPOLESTR fileName, LPPART part, BOOL externalFile, BOOL redraw);

Входные параметры:

fileName	- имя файла, из которого будет вставлен компонент,
part	- указатель на интерфейс компонента, который будет вставлен в документ (ksPart или IPart),
externalFile	- признак сохранения связи с файлом-источником: (TRUE - вставка со ссылкой на внешний файл, FALSE - вставка "телом", без сохранения ссылки на источник),
redraw	- признак перестроения документа после вставки компонента: (TRUE - перестроить документ, FALSE - не перестраивать).

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечания:

При вставке из файла указывается полный путь к файлу. При вставке из библиотеки моделей указывается полный путь к файлу библиотеки и путь внутри библиотеки моделей (например, "C:\\standard.l3d|Крепежные элементы|Болты|Болт_1").

SetRollBackFeature – Установить положение указателя в дереве

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/667_81_3_6_Ukazatelq_okonchanij.htm`

Синтаксис Automation:

BOOL SetRollBackFeature(LPDISPATCH obj);

Синтаксис COM:

BOOL SetRollBackFeature(LPFEATURE obj);

Входные параметры:

obj	- указатель на объект дерева ksFeature или IFeature, после которого нужно установить указатель.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

StopLibRequest – Остановить текущий процесс

Интерфейс...

Синтаксис Automation:

BOOL StopLibRequest();

Синтаксис COM:

BOOL StopLibRequest();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Процесс должен быть предварительно запущен функцией UserGetPlacementAndEntity.

UpdateDocumentParam – Активизировать измененные параметры документа

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_3D_REBUILD_ALL.htm`

Синтаксис Automation:

BOOL UpdateDocumentParam();

Синтаксис COM:

BOOL UpdateDocumentParam();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

UserGetCursor – Запустить процесс указания местоположения точки

Интерфейс...

Синтаксис Automation:

BOOL UserGetCursor (BSTR prompt, double * x, double * y, double * z);

Синтаксис COM:

BOOL UserGetCursor (LPOLESTR prompt, double * x, double * y, double * z);

Входные параметры:

promt	- строка подсказки для пользователя.

Выходные параметры:

x	- координата x точки, указанной пользователем,
y	- координата y точки, указанной пользователем,
z	- координата z точки, указанной пользователем.

Возвращаемое значение:

TRUE	- в случае успешного завершения (пользователь указал точку).

UserGetPlacementAndEntity – Запустить процесс указания местоположения или объектов

Интерфейс...

Синтаксис Automation:

BOOL UserGetPlacementAndEntity(long entityCount);

Синтаксис COM:

BOOL UserGetPlacementAndEntity(long entityCount);

Входной параметр:

entityCount	- максимальное количество объектов, которое может быть одновременно указано в процессе.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Параметры настройки процесса задаются с помощью интерфейса ksRequestInfo3D.

Массив объектов необходимо заполнить вручную в CallBack-функции.

В интерфейсе ksRequestInfo3D определен доступ к массиву объектов, который заполняется в процессе.

Для процесса можно задать строку приглашения и функции обратной связи для фильтрации объектов и выбора объекта.

Алгоритм работы процесса:

Разработчик определяет значение entityCount, текст строки приглашения, функции обратной связи и запускает процесс.

При захвате курсором объекта модели этот объект передается в функцию фильтрации, где можно проверить его тип и сообщить процессу, подходит объект или нет.

Если объект подходит, то он подсвечивается на экране.

После указания объекта он передается в функцию выбора, где его можно добавить в массив объектов и изменить строку приглашения.

Объекты, попавшие в массив, выделяются на экране.

После выхода из процесса массив остается доступным до очередного вызова функции GetRequestInfo.

UserSelectEntity – Запустить процесс выбора объекта

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/682_82_7_1_Vybor_obwektov_v_okn.htm`

Синтаксис Automation:

```
LPDISPATCH UserSelectEntity (LPDISPATCH defObject,
LPCTSTR methodName,
LPCTSTR prompt,
long hInstance,
LPDISPATCH dispatchOCX);
```

Входные параметры:

defObject	- указатель на объект по умолчанию, который будет подсвечен при запуске процесса,
methodName	- название функции фильтрации или NULL,
prompt	- строка подсказки или NULL,
hInstance	- hInstance модуля, в котором находится функция фильтрации,
dispatchOCX	- интерфейс, в котором находится функция фильтрации, если hInstance не доступен.

Тип функции фильтрации:

bool WINAPI SELECTFILTERPROC (LPDISPATCH _entity)

Пользователь может указать процессу, подходит ему по каким-то соображениям присланный объект или нет. Если функция фильтрации возвращает TRUE, значит, объект подходит, и он будет подсвечен процессом.

Возвращаемое значение:

указатель на интерфейс ksEntity объекта, который выделили в процессе выбора	- в случае успешного завершения,
NULL	- в случае отказа.

Синтаксис COM:

LPENTITY UserSelectEntity (LPENTITY defObject,
USERSELECTFILTERPROC methodName,
LPOLESTR prompt);

Входные параметры:

defObject	- указатель на объект по умолчанию, который будет подсвечен при запуске процесса,
methodName	- название функции фильтрации или NULL,
prompt	- строка приглашения или NULL.

Тип функции фильтрации:

typedef BOOL (__stdcall * USERSELECTFILTERPROC)(LPENTITY);

Пользователь может указать процессу, подходит ему по каким-то соображениям присланный объект или нет. Если функция фильтрации возвращает TRUE, значит, объект подходит, и он будет подсвечен процессом.

Возвращаемое значение:

указатель на интерфейс IEntity объекта, который выделили в процессе выбора	- в случае успешного завершения,
NULL	- в случае отказа.

UserSelectEntityEx – Запустить процесс выбора объекта (позволяет использовать Панель свойств)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/682_82_7_1_Vybor_obwektov_v_okn.htm

Синтаксис Automation:

LPDISPATCH UserSelectEntityEx(LPDISPATCH filterObject,
BSTR methodName,
BSTR prompt,
long hInst,
LPDISPATCH val,
LPUNKNOWN processParam);

Входные параметры:

filterObject	- указатель на объект по умолчанию, который будет подсвечен при запуске процесса,
methodName	- название функции фильтрации или NULL,
prompt	- строка подсказки или NULL,
hInst	- hInstance модуля, в котором находится функция фильтрации,
val	- интерфейс, в котором находится функция фильтрации, если hInstance не доступен,
processParam	- Параметры процесса IProcessParam.

Тип функции фильтрации:

bool WINAPI SELECTFILTERPROC (LPDISPATCH _entity)

Пользователь может указать процессу, подходит ему по каким-то соображениям присланный объект или нет. Если функция фильтрации возвращает TRUE, значит, объект подходит, и он будет подсвечен процессом.

Возвращаемое значение:

указатель на интерфейс ksEntity объекта, который выделили в процессе выбора	- в случае успешного завершения,

NULL	- в случае отказа.

Синтаксис COM:

LPENTITY UserSelectEntityEx(LPENTITY defSelectObject,
void* fnFilter,
LPOLESTR prompt,
LPUNKNOWN processParam);

Входные параметры:

defSelectObject	- указатель на объект по умолчанию, который будет подсвечен при запуске процесса,
fnFilter	- название функции фильтрации или NULL,
prompt	- строка приглашения или NULL,
processParam	- Параметры процесса IProcessParam.

Тип функции фильтрации:

typedef BOOL (__stdcall * USERSELECTFILTERPROC)(LPENTITY);

Пользователь может указать процессу, подходит ему по каким-то соображениям присланный объект или нет. Если функция фильтрации возвращает TRUE, значит, объект подходит, и он будет подсвечен процессом.

Возвращаемое значение:

указатель на интерфейс IEntity объекта, который выделили в процессе выбора	- в случае успешного завершения,
NULL	- в случае отказа.

Примечание:

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64.

Рекомендуется использовать функцию ksDocument3D::UserSelectEntityEx2.

UserSelectEntityEx2 – Запустить процесс выбора объекта (позволяет использовать Панель свойств)

Интерфейс...

Синтаксис Automation:

LPDISPATCH UserSelectEntityEx2(LPDISPATCH filterObject,
BSTR methodName, BSTR prompt,
VARIANT hInst, [defaultvalue(0)]LPDISPATCH val, [defaultvalue(0)]LPUNKNOWN processParam);

Входные параметры:

filterObject	- указатель на объект по умолчанию, который будет подсвечен при запуске процесса,
methodName	- название функции фильтрации или NULL,
prompt	- строка подсказки или NULL,
hInst	- hInstance модуля, в котором находится функция фильтрации,
val	- интерфейс, в котором находится функция фильтрации, если hInstance не доступен,
processParam	- параметры процесса IProcessParam.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Аналог функции в COM см. IRequestInfo3D::SetCallBack.

Тип функции фильтрации:

bool WINAPI SELECTFILTERPROC (LPDISPATCH _entity)

Пользователь может указать процессу, подходит ему по каким-то соображениям присланный объект или нет. Если функция фильтрации возвращает TRUE, значит объект подходит, и он будет подсвечен процессом.

Возвращаемое значение:

указатель на интерфейс IEntity объекта, который выделили в процессе выбора	- в случае успешного завершения,
NULL	- в случае отказа.

Примечание:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksDocument3D::UserSelectEntityEx.

ZoomPrevNextOrAll – Показать модель в активном окне полностью или установить предыдущий или последующий масштаб изображения

Интерфейс...

Синтаксис Automation:

BOOL ZoomPrevNextOrAll (short type);

Синтаксис COM:

BOOL ZoomPrevNextOrAll (short type);

Входные параметры:

type	- признак устанавливаемого масштаба (0 - следующий масштаб изображения, 1 - предыдущий масштаб изображения, 2 - показать модель полностью).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

В настоящий момент метод работает только при type=2.

Массив компонентов сборки (Интерфейсы ksPartCollection и IPartCollection)

Интерфейс динамического массива компонентов сборки.

ksPartCollection	- интерфейс Automation
IPartCollection	- интерфейс COM

Примечание:

Данный интерфейс можно получить от интерфейса ksDocument3D или IDocument3D.

ksPartCollection – методы

Add – Добавить компонент в массив

Интерфейс...

Синтаксис Automation:

BOOL Add (LPDISPATCH part);

Синтаксис COM:

BOOL Add (LPPART part);

Входной параметр:

part	- указатель на интерфейс компонента ksPart или IPart.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

AddAt – Добавить в массив компонент с заданным индексом

Интерфейс...

Синтаксис Automation:

BOOL AddAt (LPDISPATCH part,

long index);

Синтаксис COM:

BOOL AddAt (LPPART part,

unsigned long index);

Входные параметры:

part	- указатель на интерфейс компонента ksPart или IPart,

index - индекс в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

AddBefore – Добавить в массив компонент перед указанным компонентом

Интерфейс...

Синтаксис Automation:

BOOL AddBefore (LPDISPATCH entity,

LPDISPATCH base);

Синтаксис COM:

BOOL AddBefore (LPPART entity,

LPPART base);

Входные параметры:

part - указатель на интерфейс добавляемого компонента ksPart илиIPart,

base - указатель на интерфейс компонента базового объекта ksPart или IPart.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Clear – Очистить массив

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Элементы массива из модели не удаляются.

DetachByBody – отсоединить компонент от массива

Интерфейс...

Синтаксис Automation:

BOOL DetachByBody (LPDISPATCH part);

Синтаксис COM:

BOOL DetachByBody (LPPART part);

Входной параметр:

part - указатель на интерфейс отсоединяемого компонента ksPart или IPart.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Элемент массива из модели не удаляется.

DetachByIndex – Отсоединить от массива компонент с указанным индексом

Интерфейс...

Синтаксис Automation:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (unsigned long index);

Входной параметр:

index - индекс отсоединяемого компонента в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Элемент массива из модели не удаляется.

GetByIndex – Получить указатель на интерфейс компонента по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPPART GetByIndex (unsigned long index);

Входной параметр:

index - индекс компонента в массиве.

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

GetByName – Получить указатель на интерфейс компонента по имени

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByName (BSTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Синтаксис COM:

LPPART GetByName (LPOLESTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Входные параметры:

name	- имя объекта,
testFullName	- признак полного имени (TRUE - полное имя),
testIgnoreCase	- признак учета регистра: TRUE - игнорировать регистр, FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

GetCount – Получить количество элементов массива компонентов сборки

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

unsigned long GetCount();

Возвращаемое значение:

- количество элементов массива.

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt(LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPPART entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity	- указатель на интерфейс компонента ksPart или IPart.

First – Получить указатель на интерфейс первого компонента

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPPART First();

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Last – Получить указатель на интерфейс последнего компонента

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPPART Last();

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Next – Получить указатель на интерфейс следующего компонента

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPPART Next();

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Prev – Получить указатель на интерфейс предыдущего компонента

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPPART Prev();

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Refresh – Обновить массив компонентов сборки

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом компонентов и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить компонент в массиве по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetByIndex (LPDISPATCH part,

long index);

Синтаксис COM:

BOOL SetByIndex (LPPART part,

unsigned long index);

Входные параметры:

part - указатель на интерфейс компонента ksPart или IPart,
index - индекс заменяемого компонента в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Компонент сборки (деталь или подсборка). Интерфейсы ksPart и IPart

Интерфейс детали или подсборки в составе сборки.

ksPart	- интерфейс Automation
IPart	- интерфейс COM

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- интерфейс динамического массива компонентов сборки ksPartCollection или IPartCollection,
- интерфейс документа-модели IDocument3D, или ksDocument3D.

IPart – свойства

DoubleClickEditOff – Получить признак внешнего редактирования детали

Интерфейс...

Тип данных:BOOL.

Синтаксис Automation:

off = iPart.DoubleClickEditOff	Получить свойство (*)
iPart.DoubleClickEditOff = off	Установить свойство (*)
off = iPart.GetDoubleClickEditOff()	Получить свойство (**)
iPart.SetDoubleClickEditOff (off)	Установить свойство (**)

Примечание:

Свойство позволяет для библиотечных элементов, имеющих макропараметры, оставить стандартное поведение компонента и объектов, входящих в состав компонента. Для этого нужно установить значение свойства равным TRUE.

fileName – Имя файла компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

fileName = iPart.fileName	Получить свойство (*)
iPart.fileName = fileName	Установить свойство (*)
fileName = iPart.GetFileName()	Получить свойство (**)
iPart.SetFileName(fileName)	Установить свойство (**)

fixedComponent – Состояние фиксации компонента

Интерфейс...

Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- компонент зафиксирован,
FALSE	- компонент не зафиксирован.

Синтаксис Automation:

fixedComponent = iPart.fixedComponent	Получить свойство (*)
iPart.fixedComponent = fixedComponent	Установить свойство (*)
fixedComponent = iPart.GetFixedComponent()	Получить свойство (**)
iPart.SetFixedComponent(fixedComponent)	Установить свойство (**)

Примечание:

Если компонент зафиксирован, то нельзя изменить его местоположение.

hidden – Состояние видимости объекта

Интерфейс...

Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- объект скрытый,
FALSE	- объект видимый.

Синтаксис Automation:

hidden = iPart.hidden	Получить свойство (*)
iPart.hidden = hidden	Установить свойство (*)
hidden = iPart.GetHidden()	Получить свойство (**)
iPart.SetHidden (hidden)	Установить свойство (**)

excluded – Исключить/включить в расчет деталь

Интерфейс...

Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- исключить деталь из расчета,
FALSE	- включить деталь в расчет.

Синтаксис Automation:

excluded = iEntity.excluded		Получить свойство (*)
iEntity.excluded = excluded		Установить свойство (*)
excluded = iEntity.GetExcluded()		Получить свойство (**)
iEntity.SetExcluded(excluded)		Установить свойство (**)

Примечание:

1. После установки признака Excluded в True исключение объекта из расчета происходит сразу, без вызова Update. Поэтому изменение объектов (и вызов Update) нужно делать до переключения Excluded в True.
2. Начиная с КОМПАС-3D V12, исключенные из расчета объекты считаются не валидными.

marking – Обозначение компонента

Интерфейс...

Тип данных: строка

Синтаксис Automation:

marking = iPart.marking	Получить свойство (*)
iPart.marking = marking	Установить свойство (*)
marking = iPart.GetMarking()	Получить свойство (**)
iPart.SetMarking(marking)	Установить свойство (**)

material – Материал детали

Интерфейс...

Справка системы КОМПАС...

Тип данных: строка

Синтаксис Automation:

material = iPart.material	Получить свойство (*)
material = iPart.GetMaterial()	Получить свойство (**)

Примечания:

1. Данное свойство доступно только для чтения.
2. Обозначение материала можно получить только у детали.

MultiBodyParts – Признак того, что компонент состоит из нескольких частей

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- компонент состоит из нескольких частей,
FALSE	- компонент не состоит из нескольких частей.

Синтаксис Automation:

MultiBodyParts = iPart.MultiBodyParts		Получить свойство (*)
MultiBodyParts = iPart.GetMultiBodyParts()		Получить свойство (**)

Примечания:

Свойство позволяет узнать состоит ли компонент их нескольких частей.

В результате работы свойства можно получить компонент (деталь), состоящую из нескольких частей. С точки зрения конструктора, деталь должна представлять нечто целое. Частей быть не должно. Следовательно, нужно отыскать в компоненте операции, порождающие несколько частей, и выключить из расчета не нужные, чтобы образовалось целое тело.

name – Имя детали или подсборки в составе сборки

Интерфейс...

Тип данных: строка

Синтаксис Automation:

name = iPart.name	Получить свойство (*)
iPart.name = name	Установить свойство (*)
name = iPart.GetName()	Получить свойство (**)
iPart.SetName(name)	Установить свойство (**)

needRebuild – Необходимость перестроения объектов при изменении их свойств

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- При вызове метода ksEntity после установки модельному объекту новых свойств модель будет перестроена,
FALSE	- модель перестраиваться не будет.

Синтаксис Automation:

needRebuild iPart.needRebuild	=	Получить свойство (*)
iPart.needRebuild needRebuild	=	Установить свойство (*)
needRebuild iPart.GetNeedRebuild()	=	Получить свойство (**)
iPart.SetNeedRebuild (needRebuild)		Установить свойство (**)

Синтаксис COM:

needRebuild = iPart->GetNeedRebuild()		Получить свойство
iPart->SetNeedRebuild (needRebuild)		Установить свойство

Примечания:

Если необходимо изменить свойства нескольких объектов, то достаточно одного перестроения модели после установки новых свойств всем модельным объектам. Для этого перед изменением свойств объектов нужно установить значение этого свойства равным FALSE, поменять свойства объектов и вызвать у каждого из них метод ksEntity::Update, а затем вызвать метод ksPart::RebuildModelEx с параметром redraw равным TRUE.

Если значению данного свойства установлено FALSE, то после каждого вызова метода ksEntity::Update у объекта модель будет перестраиваться.

PropertyObjectEditable – Поддерживается интерфейс внешних свойств объекта

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

PropertyObjectEditable Object.PropertyObjectEditable	=	Получить свойство (*)
Object.PropertyObjectEditable PropertyObjectEditable	=	Установить свойство (*)
PropertyObjectEditable Object.GetPropertyObjectEditable()	=	Получить свойство (**)

Object.SetPropertyObjectEditable(PropertyObjectEditable)	Установить свойство (**)

standardComponent

Интерфейс...

Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- компонент стандартный,
FALSE	- компонент нестандартный.

Синтаксис Automation:

standardComponent = iPart.standardComponent	=	Получить свойство (*)
iPart.standardComponent = standardComponent	=	Установить свойство (*)
standardComponent = iPart.GetStandardComponent()	=	Получить свойство (**)
iPart.SetStandardComponent(standardComponent)		Установить свойство (**)

Примечание:

Стандартный компонент не имеет Дерева построения и редактируется только через прикладную библиотеку (если он был создан библиотекой, и в нем сохранены имя библиотеки и номер команды редактирования методом SetUserParam).

useColor – Используемый цвет (цвет источника, цвет хозяина, собственный цвет)

Интерфейс...

Справка системы КОМПАС...

Тип данных: long

Значения свойства...**Синтаксис Automation**:

useColor = iPart.useColor	Получить свойство (*)
iPart.useColor = useColor	Установить свойство (*)
useColor = iPart.GetUseColor()	Получить свойство (**)
iPart.SetUseColor(useColor)	Установить свойство (**)

Примечания:

1. Собственный цвет можно получить или установить, используя методы: SetAdvancedColor, GetAdvancedColor и ColorParam.
2. Свойство применимо для деталей и подсборок в сборке.

IPart – методы

BeginEdit – Запустить режим редактирования на месте для данного компонента

Интерфейс...

Синтаксис Automation:

IDispatch * BeginEdit();

Синтаксис COM:

LPDOCUMENT3D BeginEdit();

Возвращаемое значение:

- указатель на интерфейс документа ksDocument3D или IDocument3D.

Примечание:

1. Данный метод работает только для компонента, вставленного в сборку. Для детали и для "верхнего" компонента сборки возвращает NULL.
2. Метод работает только для визуального документа.

BodyCollection – Получить указатель на интерфейс массива тел

Интерфейс...

Синтаксис Automation:

LPDISPATCH BodyCollection();

Синтаксис COM:

LPBODYCOLLECTION BodyCollection();

Возвращаемое значение:

- указатель на интерфейс документа ksBodyCollection или IBodyCollection.

ClearAllObj – Удалить все вспомогательные объекты, сохранённые в детали

Интерфейс...

Синтаксис Automation:

BOOL ClearAllObj();

Синтаксис COM:

BOOL ClearAllObj();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например, отверстие - цилиндрическая поверхность) и запомнить это отверстие в макро болта, то при следующем редактировании можно получить сохранённый указатель на отверстие.

ColorParam – Получить указатель на интерфейс параметров цвета и визуальных свойств компонента

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ColorParam();

Синтаксис COM:

LPCOLORPARAM ColorParam();

Возвращаемое значение:

- указатель на интерфейс ksColorParam или IColorParam.

CreateOrEditObject – Запустить процесс создания или редактирования объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH CreateOrEditObject (short objType, LPDISPATCH editObj);

Синтаксис COM:

LPENTITY CreateOrEditObject (short objType, LPENTITY editObj);

Входные параметры:

objType - тип создаваемого объекта, выбирается из Obj3dType,
editObj - редактируемый объект.

Типы компонентов...

Возвращаемое значение:

- Указатель на интерфейс объекта ksEntity или IEntity.

Примечание:

1. Для создания нового объекта необходимо передать методу тип objType и нулевой указатель editObj.
2. Для редактирования уже существующего объекта необходимо передать методу указатель editObj на этот объект, переданный тип objType при этом игнорируется.

CurveIntersection – Рассчитать пересечения с кривой

Интерфейс...

Синтаксис Automation:

BOOL CurveIntersection (LPDISPATCH curve,
LPDISPATCH parts,
LPDISPATCH fases,
LPDISPATCH points);

Синтаксис COM:

BOOL CurveIntersection (LPCURVE3D curve,
LPPARTCOLLECTION parts,
LPFACECOLLECTION faces,
LPCOORDINATE3DCOLLECTION points);

Входные параметры:

curve - Указатель на интерфейс 3D кривой ICurve3D

Выходные параметры:

parts - коллекция деталей, пересекаемых кривой IPartCollection,
fases - коллекция граней, пересекаемых кривой IFaceCollection,
points - коллекция координат точек пересечений ICoordinate3dCollection.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Координаты точек сортируются по параметру кривой t.
2. Параметры parts, faces, points являются не обязательными.
3. Заполняются только присланные коллекции.
4. Интерфейсы коллекций для заполнения нужно получить с помощью функций:

- ksPartCollection
- ksDocument3D::PartCollection с параметром refresh = FALSE;
- ksFaceCollection
- ksDocument3D::GetInterface с параметром o3dType = o3d_faceCollection;
- ksCoordinate3dCollection
- ksDocument3D::GetInterface с параметром o3dType = o3d_coordinate3dCollection.

EndEdit – Закрыть режим редактирования на месте для данного компонента

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL EndEdit (VARIANT_BOOL rebuild);

Синтаксис COM:

BOOL EndEdit (BOOL rebuild);

Входные параметры:

rebuild - перестроить компонент.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Данный метод работает только для компонента, вставленного в сборку.

EntityCollection – Формирует массив объектов и возвращает указатель на его интерфейс

Интерфейс...

Синтаксис Automation:

LPDISPATCH EntityCollection (short objType);

Синтаксис COM:

LPENTITYCOLLECTION EntityCollection (short objType);

Входной параметр:

objType - тип объектов, содержащихся в массиве.

Возвращаемое значение:

- указатель на интерфейс ksEntityCollection или IEntityCollection.

Примечания:

1. При создании массив заполняется объектами указанного типа, содержащимися в компоненте.
2. В возвращаемом массиве включен контроль, не позволяющий добавить в него нулевой указатель.
3. Параметр objType необходим для создания массива и для правильной работы метода Refresh.

   Типы объектов...

GetAdvancedColor – Возвращает параметры цвета и визуальных свойств компонента

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetAdvancedColor (long* color,
double* ambient,
double* diffuse,
double* specularity,
double* shininess,
double* transparency,
double* emission);

Синтаксис COM:

BOOL GetAdvancedColor (COLORREF* color,
double* ambient,
double* diffuse,
double* specularity,
double* shininess,

double* transparency,

double* emission);

Выходные параметры:

color	- цвет,
ambient	- общий свет,
diffuse	- диффузия,
specularity	- зеркальность,
shininess	- блеск,
transparency	- прозрачность,
emission	- излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetCountObj – Получить количество вспомогательных объектов, сохранённых в макро

Интерфейс...

Синтаксис Automation:

BOOL GetCountObj();

Синтаксис COM:

long GetCountObj();

Возвращаемое значение:

- Количество вспомогательных объектов, сохранённых в макро.

Примечания:

Объект - это вспомогательная информация и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например, отверстие - цилиндрическая поверхность) и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

GetDefaultEntity – Получить указатель на интерфейс объекта, создаваемого системой по умолчанию

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDefaultEntity (short objType);

Синтаксис COM:

LPENTITY GetDefaultEntity (short objType);

Входной параметр :

objType - тип объекта.

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Типы объектов (objType):

o3d_planeXOY	1	- плоскость XOY
o3d_planeXOZ	2	- плоскость XOZ
o3d_planeYOZ	3	- плоскость YOZ
o3d_pointCS	4	- точка начала системы координат
o3d_axisOX	71	- ось OX
o3d_axisOY	72	- ось OY
o3d_axisOZ	73	- ось OZ

GetDensity – Получить плотность детали

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

double GetDensity()

Синтаксис COM:

double GetDensity();

Возвращаемое значение:

плотность (г/куб.мм) - в случае успешного завершения,

0 - в случае неудачи (если компонент - не деталь).

GetFeature – Вернуть указатель на интерфейс – объект дерева, связанный с данным объектом

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFeature();

Синтаксис COM:

LPFEATURE GetFeature();

Возвращаемое значение:

- указатель на интерфейс документа ksFeature или IFeature.

GetGabarit – Получить габарит

Интерфейс...

Синтаксис Automation:

BOOL GetGabarit(BOOL full
BOOL customizable
double * x1
double * y1,
double * z1,
double * x2,
double * y2,
double * z2);

Синтаксис COM:

BOOL GetGabarit(BOOL full,
BOOL customizable
double * x1,
double * y1,
double * z1,
double * x2,
double * y2,
double * z2);

Входные параметры:

full	- TRUE - полный,
	- FALSE - только тела,
customizable	- TRUE - с учетом настроек видимости,
	- FALSE - без учета; при full == false игнорируется,
x1, y1, z1	- координаты первой точки габаритного параллелепипеда,
x2, y2, z2	- координаты второй точки габаритного параллелепипеда.

Возвращаемое значение:

TRUE - в случае успеха.

GetMainBody – Получить указатель на интерфейс результирующего тела

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMainBody();

Синтаксис COM:

LPBODY GetMainBody();

Возвращаемое значение:

- указатель на интерфейс ksBody или IBody.

Пример:

Деталь имеет массив интерфейсов тел IBody. Используя этот массив интерфейсов, можно получить доступ к математической модели детали (компонента).

GetMass – Получить массу детали

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

double GetMass();

Синтаксис COM:

double GetMass();

Возвращаемое значение:

масса	- в случае успешного завершения,
0	- в случае неудачи.

CalcMassInertiaProperties – Определить массово-центровочные характеристики

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH CalcMassInertiaProperties (long bitVector);

Синтаксис COM:

LPMASSINERTIAPARAM CalcMassInertiaProperties (unsigned int bitVector);

Входной параметр:

bitVector	- определяет размерность длины, размерность массы.

Возвращаемое значение:

указатель на интерфейс ksMassInertiaParam или IMassInertiaParam.

Примечание:

Варианты значений для задания значений bitVector находятся в интервале [ST_MIX_MM..ST_MIX_KG]

Пример:

(метры|кг - ST_MIX_MIST_MIX_KG)

GetMateConstraintObjects – Получить массив зависимых объектов для повторного редактирования детали

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetMateConstraintObjects();

Синтаксис COM:

LPENTITYCOLLECTION GetMateConstraintObjects();

Возвращаемое значение:

- указатель на интерфейс ksEntityCollection или IEntityCollection.

Примечание:

Функция работает в сборке. Массив объектов формируется в результате работы функции UserGetPlacementAndEntity для данной детали. Как правило, на указанные в процессе объекты накладываются сопряжения с объектами данной детали. Если массив сохранить, то при повторном входе в процесс связанные объекты будут выделены.

GetMathematic3D – Получить интерфейс математических измерений

Интерфейс...

Синтаксис COM:

LPUNKNOWN GetMathematic3D(int type);

Входные параметры:

type - тип запрашиваемого интерфейса, допустимыми значениями являются следующие:

- o3d_sTrackingPointsMeasurer,
- o3d_measurer.

Возвращаемое значение:

указатель на интерфейс IUnknown - в случае успеха,
NULL - в случае неудачи.

Примечание:

Метод позволяет получать интерфейс математических измерений по типу Obj3dType в виде интерфейса IUnknown. Полученный интерфейс необходимо привести к интерфейсу, соответствующему типу type с помощью функции IUnknown::QeryInterface.

GetMeasurer – Получить указатель на интерфейс измерений

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMeasurer();

Синтаксис COM:

LPMEASURER GetMeasurer();

Возвращаемое значение:

- указатель на интерфейс ksMeasurer или IMeasurer.

GetObject – Получить указатель на вспомогательный объект, сохраненный в деталь по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObject(long index);

Синтаксис COM:

LPUNKNOWN GetObject(long index);

Входные параметры:

index - номер объекта.

Возвращаемое значение:

Указатель на интерфейс IDispatch или IUnknown сохранённого объекта.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактировании можно получить сохранённый указатель на отверстие.

См. также: IPart::SetObject

GetObjectByName – Получить компонент по имени

Интерфейс...

Синтаксис Automation:

```
LPDISPATCH GetObjectByName( BSTR name,
short objType,
BOOL testFullName,
BOOL testIgnoreCase );
```

Синтаксис COM:

```
LPUNKNOWN GetObjectByName( LPOLESTR name,
short  objType,
BOOL testFullName,
BOOL testIgnoreCase );
```

Входные параметры:

name	- имя компонента,
objType	- тип компонента,
testFullName	- полное имя компонента: TRUE - использовать полное имя, FALSE - частичное совпадение,
testIgnoreCase	- игнорировать регистр.

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

См. также: IPart::SetObject

GetObject3DNotify – Получить объект обработки событий для объектов 3D документов

Интерфейс...

Синтаксис Automation:

ksObject3DNotify * GetObject3DNotify (long objType, LPDISPATCH obj);

Входные параметры:

Obj	- указатель на объект,
objType	- тип объекта (o3d_unknown...o3d_sketch, o3d_axis2Planes...o3d_thread, o3d_part, o3d_feature).

Возвращаемое значение:

- указатель на интерфейс источника событий Object3DNotify.

Примечание:

Если задан указатель на объект, события генерируются только для этого объекта. Если задан тип объекта, события генерируются для всех объектов этого типа. Если задан тип o3d_unknown, события генерируются для всех объектов.

GetObject3DNotifyResult – Получить интерфейс результатов редактирования объекта документа-модели

Интерфейс...

Синтаксис Automation:

ksObject3DNotifyResult * GetObject3DNotifyResult();

Синтаксис COM:

IObject3DNotifyResult * GetObject3DNotifyResult();

Возвращаемое значение:

- указатель на интерфейс источника событий ksObject3DNotifyResult или IObject3DNotifyResult.

GetPart – Получить указатель на интерфейс компонента в соответствии с заданным типом

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPart (long type);

Синтаксис COM:

LPPART GetPart(int type);

Входные параметры:

type - тип компонента.

Типы компонентов...

Возвращаемое значение:

- указатель на интерфейс компонента ksPart или IPart.

Примечание:

Тип задается для нового, редактируемого или главного компонента либо тип равен номеру компонента в документе. Функция используется, чтобы получить доступ к компоненту документа. Деталь или сборка являются компонентами. Сборка, в свою очередь, состоит из компонентов - деталей и подсборок.

GetPlacement – Получить указатель на интерфейс местоположения компонента

Пример ...Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

GetSummMatrix – Получить суммарную матрицу преобразования координат

Интерфейс...

Синтаксис Automation:

VARIANT GetSummMatrix(LPDISPATCH part1);

Синтаксис COM:

BOOL GetSummMatrix(VARIANT * matrix,
LPPART part1);

Входные параметры:

part1 - указатель на интерфейс ksPart или IPart детали из которой нужно сделать пересчет координат.

Возвращаемое значение:

массив SafeArray типа VT_ARRAY | VT_R8

Примечания:

1. Элементы матрицы возвращаются в виде одномерного массива из шестнадцати элементов.
2. Матрица имеет размер 4x4.

GetUserLibraryCommand – Получить номер команды пользовательской библиотеки, при помощи которой можно редактировать компонент

Интерфейс...

Аналог данного метода при использовании Automation - ksUserParam::number.

Пример...

Синтаксис COM:

long GetUserLibraryCommand();

Возвращаемое значение:

Номер библиотечной команды - в случае успешного завершения,
-1 - если библиотеки нет.

GetUserLibraryFileName – Получить имя файла пользовательской библиотеки, при помощи которой можно редактировать компонент

Интерфейс...

Аналог данного метода при использовании Automation - ksUserParam::fileName. Пример....

Синтаксис COM:

LPOLESTR GetUserLibraryFileName();

Возвращаемое значение:

Имя файла пользовательской библиотеки	- в случае успешного завершения,
NULL	- если библиотеки нет.

GetUserLibraryName – Получить имя пользовательской библиотеки, при помощи которой можно редактировать компонент

Интерфейс...

Аналог данного метода при использовании Automation - ksUserParam::libName.

Пример...

Синтаксис COM:

LPOLESTR GetUserLibraryName();

Возвращаемое значение:

Имя пользовательской библиотеки	- в случае успешного завершения,
NULL	- если библиотеки нет.

GetUserParam – Возвращает пользовательские параметры, записанные в компоненте

Интерфейс...

Синтаксис Automation:

BOOL GetUserParam (LPDISPATCH userPars);

Выходной параметр:

userPars	- указатель на интерфейс пользовательских параметров ksUserParam.

Синтаксис COM:

BOOL GetUserParam (void *value,

unsigned int size);

Входной параметр:

Size	- размер структуры параметров.

Выходной параметр :

Value	- пользовательская структура параметров.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Способ получения пользовательских данных (userPars) должен совпадать со способом их сохранения (void*, SAFEARRAY, или DynamicArray) через ksPart::SetUserParam-.

- Если пользовательские данные были сохранены через SAFEARRAY (userPars), то перед их получением нужно создать SAFEARRAY соответствующего размера (размер данных можно определить при помощи ksPart::GetUserParamSize).

Если пользовательские данные были сохранены через ksUserParam::SetUserArray, то перед их получением нужно создать UserArray, аналогичный по структуре используемому при сохранении, и передать его в ksUserParam::SetUserArray.

GetUserParamSize – Возвращает размер (в байтах) пользовательских параметров, записанных в компоненте

Интерфейс...

Синтаксис Automation:

long GetUserParamSize();

Синтаксис COM:

long GetUserParamSize();

IsDetail – Определить, является ли компонент деталью

Интерфейс...

Синтаксис Automation:

BOOL IsDetail();

Синтаксис COM:

BOOL IsDetail();

Возвращаемое значение:

TRUE - деталь,
FALSE - сборка.

NewEntity – Создать новый интерфейс объекта и получить указатель на него

Интерфейс...

Синтаксис Automation:

LPDISPATCH NewEntity (short objType);

Синтаксис COM:

LPENTITY NewEntity (short objType);

Входной параметр:

objType - тип объекта.

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечание:

Реальный объект создается в модели после вызова метода Create.

PutStorage – Вставить деталь-заготовку

Интерфейс...

Справка системы КОМПАС...

ADD_COMPONENT_FROM_FILE.htm

Синтаксис Automation:

BOOL PutStorage (LPOLESTR fileName,

BOOL type,

BOOL mirror);

Синтаксис COM:

BOOL PutStorage (const char* fileName,

BOOL type,

BOOL mirror);

Входные параметры:

fileName - имя файла детали-заготовки,
type - тип вставки детали-заготовки
(TRUE - внешней ссылкой,
FALSE - без истории),
mirror - признак вставки "зеркальной" детали
(TRUE - "зеркальная" деталь,
FALSE - деталь в том виде, в котором она существует в файле-источнике).

Возвращаемое значение:

TRUE - в случае успешного завершения.

RebuildModel – Перестроить модель в соответствии с новыми значениями свойств объектов и внешних переменных

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL RebuildModel();

Синтаксис COM:

BOOL RebuildModel();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Если изменены значения внешних переменных компонента, то они будут переданы в модель. Если компонент является фантомом, то он будет перестроен.

RebuildModelEx – Перестроить модель в соответствии с новыми значениями свойств объектов и внешних переменных

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL RebuildModelEx (BOOL redraw);

Синтаксис COM:

BOOL RebuildModelEx (BOOL redraw);

Входные параметры:

redraw - признак перестроения документа:
TRUE - перестроить документ,
FALSE - не перестраивать.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

1. Если изменены значения внешних переменных компонента, то они будут переданы в модель. Если параметру redraw установлено значение TRUE или компонент является фантомом, то он будет перестроен.
2. Значение redraw, равное FALSE, предполагается использовать для ускорения работы при многократном редактировании одной и той же детали. Перерисовка документа в этом случае не выполняется. Однако следует учитывать, что после завершения редактирования необходимо вызвать метод с параметром redraw, равным TRUE.

SetAdvancedColor – Установить параметры цвета и визуальных свойств объекта

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAdvancedColor (long color,

double ambient,

double diffuse,

double specularity,

double shininess,

double transparency,

double emission);

Синтаксис COM:

BOOL SetAdvancedColor (COLORREF color,

double ambient,

double diffuse,

double specularity,

double shininess,

double transparency,

double emission);

Входные параметры:

color	- цвет,
ambient	- общий свет,
diffuse	- диффузия,
specularity	- зеркальность,
shininess	- блеск,
transparency	- прозрачность,
emission	- излучение.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetFileName – Установить имя файла детали

Интерфейс...

Синтаксис Automation:

BOOL SetFileName (BSTR name);

Синтаксис COM:

BOOL SetFileName (LPOLESTR name);

Входной параметр:

name - полное имя файла детали.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

1. Метод используется для компонентов, вставленных в сборку.
2. Документ с указанным именем должен существовать.
3. Компонент не должен быть деталью из библиотеки моделей или стандартным элементом.
4. Изменение вступает в силу после вызова метода ksPart::Update.

SetMateConstraintObjects – Сохранить в детали зависимые объекты для повторного редактирования детали

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetMateConstraintObjects (LPDISPATCH collection);

Синтаксис COM:

BOOL SetMateConstraintObjects (LPENTITYCOLLECTION collection);

Входной параметр:

collection - указатель на интерфейс ksEntityCollection или IEntityCollection.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Функция работает в сборке. Массив объектов формируется в результате работы процесса UserGetPlacementAndEntity для данной детали. Как правило, на указанные в процессе объекты накладываются сопряжения с объектами данной детали. Если массив сохранить, то при повторном входе в процесс связанные объекты будут выделены. Если collection = NULL, массив будет взят из документа, который получился после работы процесса UserGetPlacementAndEntity для данной детали.

SetMaterial – Установить материал детали

Интерфейс...

Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetMaterial (BSTR name,

double density);

Синтаксис COM:

BOOL SetMaterial (LPOLESTR name,

double density);

Входные параметры:

name - название материала,
density - плотность материала (г/куб.см).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

1. Компонент, у которого меняется материал, должен быть деталью.
2. Компонент не должен быть деталью из библиотеки моделей или стандартным элементом.
3. Метод используется в детали для верхнего компонента, в сборках - для вставленных деталей.
4. Изменение материала вступает в силу после вызова метода ksPart::Update.

SetObject – Сохранить указатель на вспомогательный объект в деталь по индексу

Интерфейс...

Синтаксис Automation:

BOOL SetObject(long index, LPDISPATCH obj);

Синтаксис COM:

BOOL SetObject(long index, LPUNKNOWN obj);

Входные параметры:

index - номер объекта,
obj - указатель на интерфейс объекта,

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечания:

Объект - это вспомогательная информация и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактировании можно получить сохранённый указатель на отверстие.

См. также: IPart::GetObject

SetPlacement – Установить новое местоположение компонента

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement (LPDISPATCH placement);

Синтаксис COM:

BOOL SetPlacement (LPPLACEMENT placement);

Входной параметр:

placement - указатель на интерфейс ksPlacement или IPlacement.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Данный метод вступает в силу после вызова метода UpdatePlacement.

SetSourceVariables – Установить для вставки значения переменных из источника

Интерфейс...

Синтаксис Automation:

BOOL SetSourceVariables(BOOL rebuild);

Синтаксис COM:

BOOL SetSourceVariables(BOOL rebuild);

Входной параметр:

rebuild - перестроить вставку.

Примечание:

Функция позволяет сбросить изменения значений переменных во вставке и установить значения, заданные в источнике.

SetUserParam – Установить параметры компонента, указанные пользователем

Интерфейс...

Синтаксис Automation:

BOOL SetUserParam (LPDISPATCH userPars);

Входной параметр:

userPars - указатель на интерфейс пользовательских параметров ksUserParam.

Синтаксис COM:

BOOL SetUserParam (void *value,

unsigned int size,

LPOLESTR nameFile,

LPOLESTR nameLib,

int number);

Входные параметры:

value - указатель на пользовательскую структуру параметров,
size - размер структуры параметров,

nameFile	- имя файла прикладной библиотеки для последующего редактирования через библиотеку (0 - запоминается текущая библиотека),
nameLib	- имя прикладной библиотеки для последующего редактирования через библиотеку (0 - запоминается текущая библиотека),
number	- номер команды в прикладной библиотеке для последующего редактирования через библиотеку (-1 - запоминается текущая команда).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Данный метод позволяет сохранить параметры пользователя для последующего редактирования с помощью библиотеки.

TransformPoint – Перевести координаты точки присланной детали part1 в систему координат детали

Интерфейс...

Синтаксис Automation:

BOOL TransformPoint (double* x, double* y, double* z, LPDISPATCH part1);

Синтаксис COM:

BOOL TransformPoint (double* x, double* y, double* z, LPPART part1);

Входные параметры:

x, y, z	- координаты точки в системе координат детали part1,
part1	указатель на интерфейс детали ksPart, из системы координат которой нужно перевести точку.

Выходные параметры:

x, y, z	- координаты точки в системе координат детали.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

TransformPoints – Перевести координаты точек присланной детали part1 в систему координат детали

Интерфейс...

Синтаксис Automation:

BOOL TransformPoints(VARIANT * points,
LPDISPATCH part1);

Синтаксис COM:

BOOL TransformPoints(VARIANT * points,
LPPART part1);

Входные параметры:

points	- массив SAFEARRAY координат точек,
part1	- указатель на интерфейс детали в системе координат которой заданы точки.

Выходные параметры:

points	- массив SAFEARRAY координат точек.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечания:

1. Значения координат изменяются непосредственно в присланном массиве SAFEARRAY.
2. Тип варианта должен быть VT_ARRAY | VT_R8.
3. Значения в массиве должны лежать в последовательности x1, y1, z1, x2, y2, z2 ... xn, yn, zn.

VariableCollection – Получить указатель на интерфейс массива внешних переменных

Интерфейс...

Синтаксис Automation:

LPDISPATCH VariableCollection();

Синтаксис COM:

LPVARIABLECOLLECTION VariableCollection();

Возвращаемое значение:

- указатель на интерфейс ksVariableCollection или IVariableCollection .

Примечание:

Изменения в полученном массиве не отображаются в модели немедленно. Чтобы изменения вступили в силу, необходимо вызвать метод RebuildModel.

Update – Изменить свойства компонента (используя ранее установленные свойства)

Интерфейс...

Синтаксис Automation:

BOOL Update();

Синтаксис COM:

BOOL Update();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

После вызова этого метода все изменения отображаются в компоненте.

UpdatePlacement – Изменить местоположение компонента, заданное методом SetPlacement

Пример...Интерфейс...

Синтаксис Automation:

BOOL UpdatePlacement();

Синтаксис COM:

BOOL UpdatePlacement();

Возвращаемое значение:

TRUE - в случае успешного завершения.

UpdatePlacementEx – Изменить местоположение компонента, заданное методом SetPlacement

Интерфейс...

Синтаксис Automation:

BOOL UpdatePlacementEx (BOOL redraw);

Синтаксис COM:

BOOL UpdatePlacementEx (BOOL redraw);

Входные параметры:

redraw - признак перестроения документа после изменения местоположения компонента:
(TRUE - перестроить документ, FALSE - не перестраивать).

Возвращаемое значение:

TRUE - в случае успешного завершения.

Массив объектов модели(Интерфейсы ksEntityCollection и IEntityCollection)

Интерфейс массива объектов модели.

ksEntityCollection - интерфейс Automation
IEntityCollection - интерфейс COM

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- Интерфейс вырезанного кинематического элемента ksCutEvolutionDefinition или ICutEvolutionDefinition.
- Интерфейс детали или подсборки в составе сборки ksPart или IPart.
- Интерфейс документа-модели ksDocument3D или IDocument3D.
- Интерфейс зеркальной копии ksMirrorCopyDefinitiont или IMirrorCopyDefinitiont.
- Интерфейс объекта Дерева построения ksFeaturet или IFeaturet.
- Интерфейс операции копирования компонентов сборки по кривой ksCurvePartArrayDefinitiont или ICurvePartArrayDefinitiont.
- Интерфейс операции копирования по кривой ksCurveCopyDefinitiont или ICurveCopyDefinitiont.
- Интерфейс операции копирования по окружности ksCircularCopyDefinitiont или ICircularCopyDefinition.
- Интерфейс операции копирования по сетке ksMeshCopyDefinitiont или ICopyMeshDefinitiont.
- Интерфейс параметров вырезанного элемента по сечениям ksCutLoftDefinitiont или ICutLoftDefinitiont.
- Интерфейс параметров запроса к системе ksRequestInfo3D или IRequestInfo.
- Интерфейс параметров кинематической поверхности ksEvolutionSurfaceDefinitiont или IEvolutionSurfaceDefinitiont.
- Интерфейс параметров основания - кинематического элемента ksBaseEvolutionDefinitiont или IBaseEvolutionDefinitiont.
- Интерфейс параметров основания - элемента по сечениям ksBaseLoftDefinition или IBaseLoftDefinition.
- Интерфейс параметров поверхности по сечениям ksLoftSurfaceDefinition или ILoftSurfaceDefinitiont.
- Интерфейс параметров фаски ksChamferDefinition или IChamferDefinition,
- Интерфейс параметров элемента "скругление" ksFilletDefinition или IFilletDefinition.
- Интерфейс приклеенного кинематического элемента ksBossEvolutionDefinition или IBossEvolutionDefinition.
- Интерфейс приклеенного элемента по сечениям ksBossLoftDefinition или IBossLoftDefinition.
- Интерфейс тонкостенной оболочки ksShellDefinition или IShellDefinition.

- Интерфейс уклона ksInclineDefinition или IInclineDefinition.

ksEntityCollection – методы

Add – Добавить объект в массив

Интерфейс...

Синтаксис Automation:

BOOL Add (LPDISPATCH entity);

Синтаксис COM:

BOOL Add (LPENTITY entity);

Входной параметр:

entity	- интерфейс добавляемого элемента ksEntity или IEntity.

Возвращаемое значение:

Функция возвращает TRUE в двух случаях:

- Если при работе с динамическим массивом объектов модели (метод EntityCollection) checkEntity - признак проверки для вновь добавляемых объектов на NULL - имеет значение FALSE (т.е. в массив можно добавить NULL).
- Если при работе с динамическим массивом объектов модели checkEntity имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - не NULL.

  Функция возвращает FALSE, если при работе с динамическим массивом объектов модели checkEntity - имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - NULL.

AddAt – Добавить объект в массив с заданным индексом

Интерфейс...

Синтаксис Automation:

BOOL AddAt (LPDISPATCH entity,

long index);

Синтаксис COM:

BOOL AddAt (LPENTITY entity,

long index);

Входные параметры:

entity	- интерфейс добавляемого элемента ksEntity или IEntity,
index	- индекс в массиве.

Возвращаемое значение:

Функция возвращает TRUE в двух случаях:

- Если при работе с динамическим массивом объектов модели (метод EntityCollection) checkEntity - признак проверки для вновь добавляемых объектов на NULL - имеет значение FALSE (т.е. в массив можно добавить NULL).
- Если при работе с динамическим массивом объектов модели checkEntity имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - не NULL.

  Функция возвращает FALSE, если при работе с динамическим массивом объектов модели checkEntity имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - NULL.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс...

Синтаксис Automation:

BOOL AddBefore (LPDISPATCH entity,

LPDISPATCH base);

Синтаксис COM:

BOOL AddBefore (LPENTITY entity,

LPENTITY base);

Входные параметры:

entity	- интерфейс добавляемого элемента ksEntity или IEntity,
base	- интерфейс базового объекта ksEntity или IEntity (не должен быть 0).

Возвращаемое значение:

Функция возвращает TRUE в двух случаях:

- Если при работе с динамическим массивом объектов модели (метод EntityCollection) checkEntity - признак проверки для вновь добавляемых объектов на NULL - имеет значение FALSE (т.е. в массив можно добавить NULL).
- Если при работе с динамическим массивом объектов модели checkEntity имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - не NULL.

  Функция возвращает FALSE, если при работе с динамическим массивом объектов модели checkEntity имеет значение TRUE (т.е. в массив нельзя добавить NULL) и объект - NULL.

Clear – Очистить динамический массив объектов

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Объекты массива из модели не удаляются.

DetachByBody – Удалить указанный объект из массива

Интерфейс...

Синтаксис Automation:

BOOL DetachByBody(LPDISPATCH entity);

Синтаксис COM:

BOOL DetachByBody(LPENTITY entity);

Входной параметр:

entity - указатель на интерфейс удаляемого элемента ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечания:

1. Указатель на интерфейс удаляемого объекта entity не должен быть 0.
2. Элемент массива из модели не удаляется.

DetachByIndex – Удалить объект из массива по индексу

Интерфейс...

Синтаксис Automation:

BOOL DetachByIndex(long index);

Синтаксис COM:

BOOL DetachByIndex(long index);

Входной параметр:

index - индекс объекта в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Элемент массива из модели не удаляется.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPENTITY GetByIndex (long index);

Входной параметр:

index - номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetByName – Получить указатель на интерфейс объекта в массиве по имени

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByName (BSTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Синтаксис COM:

LPENTITY GetByName (LPOLESTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Входные параметры:

name	- имя объекта,
testFullName	- признак полного имени: TRUE - name - полное имя, FALSE - имя name может быть частью полного имени,
testIgnoreCase	- признак игнорирования регистра символов: TRUE - игнорировать регистр, FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetCount – Получить количество элементов в массиве

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
-1 - если элемент не найден.

Синтаксис COM:

unsigned long FindIt(LPENTITY entity);

Входные параметры:

entity - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
SYS_MAX_UINT - если элемент не найден.

First – Получить первый объект в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPENTITY First();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPENTITY Last();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPENTITY Next();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPENTITY Prev();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Refresh – Обновить массив интерфейсов объектов трехмерной модели (осей, плоскостей и т.п.)

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива в памяти с массивом точек ребра (поверхности) и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс...

Синтаксис Automation:

BOOL SetByIndex (LPDISPATCH entity,

long index);

Синтаксис COM:

BOOL SetByIndex (LPENTITY entity,

long index);

Входные параметры:

entity - указатель на интерфейс присоединяемого элемента ksEntity или IEntity,
index - индекс элемента в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SelectByPoint – Исключить из массива все объекты, не содержащие точку с заданными координатами

Интерфейс...

Синтаксис Automation:

BOOL SelectByPoint (double x,

double y,

double z);

Синтаксис COM:

BOOL SelectByPoint (double x,

double y,

double z);

Входные параметры:

x, y, z - координаты точки,

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

После выполнения этой функции в массиве останутся только те объекты, которые содержат указанную точку (проходят через нее).

Данная функция временно не работает для формообразующих элементов (элементов выдавливания, вращения, по сечениям).

Объект модели (Интерфейсы ksEntity и IEntity)

Интерфейсы ksEntity и iEntity – Интерфейс элемента модели (оси, плоскости, формообразующего элемента)

ksEntity - интерфейс Automation
IEntity - интерфейс COM

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- Интерфейс детали или подсборки в составе сборки ksPart или IPart.
- Методы интерфейса коллекции элементов модели ksEntityCollection, IEntityCollection.
- Метод интерфейса документа-модели ksDocument3D::UserSelectEntity или IDocument3D::UserSelectEntity.
- Интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.
- Интерфейс параметров вершины ломаной ksPolyLineVertexParam или IPolygonalLineVertexParam.
- Интерфейсы элементов модели

свойства

excluded – Признак включения или выключения из расчета

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- исключить из расчёта,
FALSE	- включить в расчёт.

Синтаксис Automation:

excluded = iEntity.excluded	Получить свойство (*)
iEntity.excluded = excluded	Установить свойство (*)
excluded = iEntity.GetExcluded()	Получить свойство (**)
iEntity.SetExcluded(excluded)	Установить свойство (**)

Примечание:

1. После установки признака Excluded в True исключение объекта из расчета происходит сразу, без вызова Update. Поэтому изменение объектов (и вызов Update) нужно делать до переключения Excluded в True.
2. Начиная с КОМПАС-3D V12, исключенные из расчета объекты считаются не валидными.

hidden – Состояние видимости объекта

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- объект скрытый,
FALSE	- объект видимый.

Синтаксис Automation:

hidden = iEntity.hidden	Получить свойство (*)
iEntity.hidden = hidden	Установить свойство (*)
hidden = iEntity.GetHidden()	Получить свойство (**)
iEntity.SetHidden(hidden)	Установить свойство (**)

MultiBodyParts – Признак того, что компонент состоит из нескольких частей

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- компонент состоит из нескольких частей,
FALSE	- компонент не состоит из нескольких частей.

Синтаксис Automation:

MultiBodyParts = iEntity.MultiBodyParts	Получить свойство (*)
MultiBodyParts = iEntity.GetMultiBodyParts()	Получить свойство (**)

Примечания:

Свойство позволяет узнать. состоит ли компонент их нескольких частей.

В результате работы метода можно получить компонент (деталь), состоящую из нескольких частей. С точки зрения конструктора, деталь должна представлять нечто целое. Частей быть не должно. Следовательно, нужно отыскать в компоненте операции, порождающие несколько частей, и выключить из расчета не нужные, чтобы образовалось целое тело.

name – Имя элемента трехмерной модели (оси, плоскости, формообразующего элемента)

Интерфейс...Тип данных: строка

Синтаксис Automation:

name = iEntity.name	Получить свойство (*)
iEntity.name = name	Установить свойство (*)
name = iEntity.GetName()	Получить свойство (**)
iEntity.SetName(name)	Установить свойство (**)

type – Тип объекта трехмерной модели

Интерфейс...Тип данных: short

Значения свойства...

Синтаксис Automation:

type = iEntity.type	Получить свойство (*)

iEntity.type = type	Установить свойство (*)
type = iEntity.GetType()	Получить свойство (**)
iEntity.SetType(type)	Установить свойство (**)

Примечание:

1. Числовой идентификатор типа объекта не должен превышать идентификатор объекта, всегда последнего из Entity.
2. Свойство только для чтения.

useColor – Используемый цвет (цвет источника, цвет хозяина, цвет слоя, собственный цвет)

Интерфейс...Тип данных: UseColor

Синтаксис Automation:

useColor = Object.useColor	Получить свойство(*)
Object.useColor = useColor	Установить свойство (*)
useColor = Object.GetUseColor()	Получить свойство (**)
Object.SetUseColory(useColor)	Установить свойство (**)

методы

BodyCollection – Получить указатель на интерфейс массива трехмерных тел

Интерфейс...**Синтаксис Automation**:

LPDISPATCH BodyCollection();

Синтаксис COM:

LPBODYCOLLECTION BodyCollection();

Возвращаемое значение:

- указатель на интерфейс массива тел ksBodyCollection или IBodyCollection.

ColorParam – Получить указатель на интерфейс параметров цвета и визуальных свойств объекта

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ColorParam();

Синтаксис COM:

LPCOLORPARAM ColorParam();

Возвращаемое значение:

- указатель на интерфейс ksColorParam или IColorParam.

Create – Создать объект в модели

Интерфейс...**Синтаксис Automation**:

BOOL Create();

Синтаксис COM:

BOOL Create();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Данный метод необходимо вызвать для создания объекта в модели после создания интерфейса с помощью метода NewEntity и изменения параметров создаваемого объекта.

Если до вызова данного метода свойства объекта не изменены, то объект создается с параметрами, принятыми по умолчанию.

Если объект уже создан, то функция вызовет метод Update.

GetAdvancedColor – Получить параметры цвета и визуальных свойств объекта

Интерфейс...Справка системы КОМПАС...

`CM_NAME_CAPTION`

Синтаксис Automation:

BOOL GetAdvancedColor (long* color,

double* ambient,

double* diffuse,

double* specularity,

double* shininess,

double* transparency,

double* emission);

Синтаксис COM:

BOOL GetAdvancedColor (COLORREF* color,

double* ambient,

double* diffuse,

double* specularity,

double* shininess,

double* transparency,

double* emission);

Выходные параметры:

color	- цвет,
ambient	- общий свет,
diffuse	- диффузия,
specularity	- зеркальность,
shininess	- блеск,
transparency	- прозрачность,
emission	- излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetBodyParts – Получить указатель на интерфейс частей тела

Интерфейс...**Синтаксис Automation**:

LPBODYPARTS GetBodyParts();

Синтаксис COM:

LPDISPATCH GetBodyParts();

Возвращаемое значение:

указатель на интерфейс массива тел ksBodyParts или IBodyParts.	- если объект состоит из частей,
NULL	- если объект не имеет частей.

Примечание:о

Метод позволяет получить интерфейс частей тела. Если объект не имеет частей, то метод вернет NULL. Части тела могут быть у операций вырезания.

GetDefinition – Получить указатель на интерфейс параметров объектов и элементов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetDefinition();

Возвращаемое значение:

- указатель на интерфейс IDispatch.

Синтаксис COM:

LPUNKNOWN GetDefinition();

Возвращаемое значение:

- указатель на интерфейс IUnknown.

Примечание:

1. Для доступа к параметрам объекта полученный интерфейс необходимо привести к соответствующему типу.

   Интерфейсы элементов модели...
2. Интерфейсы параметров объектов и элементов выбираются из перечня Obj3dType.

GetFeature – Получить указатель на интерфейс объекта дерева, связанного с данным объектом

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetFeature();

Синтаксис COM:

LPFEATURE GetFeature();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature - объект дерева, связанный с данным объектом.

GetParent – Получить указатель на интерфейс компонента, владеющего элементом (например, интерфейс детали, владеющей указанной гранью)

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetParent()

Синтаксис COM:

LPPART GetParent();

Возвращаемое значение:

- указатель на интерфейс ksPart или IPart.

IsCreated – Получить признак существования объекта

Интерфейс...**Синтаксис Automation**:

BOOL IsCreated();

Синтаксис COM:

BOOL IsCreated();

Возвращаемое значение:

TRUE - если объект уже создан,
FALSE - если объект удален или еще не создан.

IsIt – Проверить объект на соответствие указанному типу

Интерфейс...**Синтаксис Automation**:

BOOL IsIt (short objType);

Синтаксис COM:

BOOL IsIt (short objType)

Входной параметр:

objType - тип объекта.

Возвращаемое значение:

TRUE - если тип объекта указан правильно.

Типы объектов...

Примечание:

Числовой идентификатор типа объекта не должен превышать идентификатора объекта, всегда последнего из Entity.

SetAdvancedColor – Изменить параметры цвета и визуальных свойств объекта

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAdvancedColor (long color,
double ambient,
double diffuse,
double specularity,
double shininess,

double transparency,

double emission);

Синтаксис COM:

BOOL SetAdvancedColor (COLORREF color,

double ambient,

double diffuse,

double specularity,

double shininess,

double transparency,

double emission);

Входные параметры:

color	- цвет,
ambient	- общий свет,
diffuse	- диффузия,
specularity	- зеркальность,
shininess	- блеск,
transparency	- прозрачность,
emission	- излучение.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Update – Изменить свойства объекта (используя ранее установленные свойства)

Интерфейс...**Синтаксис Automation**:

BOOL Update();

Синтаксис COM:

BOOL Update();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

1. Изменение параметров объекта через интерфейс его параметров не приводит к перестроению модели.
2. Для перестроения объекта в модели (активизации текущих изменений) необходимо вызвать данный метод.

   Если свойству ksPart::needRebuild компонента, которому принадлежит данный объект, установлено значение TRUE, то вызов данного метода приведет также к перестроению модели. Если свойству ksPart::needRebuild установлено значение FALSE, то данный ме-

тод только передаст изменения свойств модельному объекту, и для реального (визуального) изменения этих свойств нужно вызвать метод ksPart::RebuildModelEx, с параметром redraw равным TRUE.

3. Объект должен быть предварительно создан методом Create.

Интерфейсы пространственных кривых

Коническая спираль (Интерфейсы ksConicSpiralDefinition и IConicSpiralDefinition)

Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Интерфейс параметров конической спирали.

ksConicSpiralDefinition	- интерфейс Automation
IConicSpiralDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IConicSpiralDefinition – свойства

buildDir – Направление построения спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: BOOL

Значения свойства:

TRUE	- нормальное (совпадающее с положительным направлением нормали к базовой плоскости спирали),
FALSE	- обратное.

Синтаксис Automation:

buildDir = iConicSpiral.buildDir		Получить свойство (*)
iConicSpiral.buildDir = buildDir		Установить свойство (*)
buildDir	=	Получить свойство (**)
iConicSpiral.GetBuildDir()		
iConicSpiral.SetBuildDir(buildDir)		Установить свойство (**)

buildMode – Способ построения спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: short

Значения свойства:

0	- по шагу и количеству витков,
1	- по шагу и высоте,
2	- по количеству витков и высоте.

Синтаксис Automation:

buildMode = iConicSpiral.buildMode		Получить свойство (*)
iConicSpiral.buildMode = buildMode		Установить свойство (*)
buildMode	=	Получить свойство (**)
iConicSpiral.GetBuildMode()		
iConicSpiral.SetBuildMode(buildMode)		Установить свойство (**)

firstAngle – Начальный угол (или угол поворота спирали вокруг своей оси)

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных:double.

Синтаксис Automation:

firstAngle = iConicSpiral.firstAngle		Получить свойство (*)
iConicSpiral.firstAngle = firstAngle		Установить свойство (*)
firstAngle	=	Получить свойство (**)
iConicSpiral.GetFirstAngle()		
iConicSpiral.SetFirstAngle(firstAngle)		Установить свойство (**)

height – Высота спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

height = iConicSpiral.height	Получить свойство (*)
iConicSpiral.height = height	Установить свойство (*)

height	=	Получить свойство (**)
iConicSpiral.GetHeight()		
iConicSpiral.SetHeight(height)		Установить свойство (**)

heightAdd – Дополнительная высота

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

heightAdd = iConicSpiral.heightAdd		Получить свойство (*)
iConicSpiral.heightAdd = heightAdd		Установить свойство (*)
heightAdd	=	Получить свойство (**)
iConicSpiral.GetHeightAdd()		
iConicSpiral.SetHeightAdd(heightAdd)		Установить свойство (**)

heightAddHow – Признак дополнительной высоты

Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: BOOL

Значения свойства:

TRUE	- продолжить за объект,
FALSE	- не доходя до объекта.

Синтаксис Automation:

heightAddHow = iConicSpiral.heightAddHow	Получить свойство (*)
iConicSpiral.heightAddHow = heightAddHow	Установить свойство (*)
heightAddHow = iConicSpiral.GetHeightAddHow()	Получить свойство (**)
iConicSpiral.SetHeightAddHow(heightAddHow)	Установить свойство (**)

Примечание:

Данное свойство используется только при способе задания высоты "по объекту" (heightType = 1).

heightType – Способ задания высоты

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: short

Значения свойства:

0 - по размеру,
1 - по объекту.

Синтаксис Automation:

heightType = iConicSpiral.heightType — Получить свойство (*)
iConicSpiral.heightType = heightType — Установить свойство (*)
heightType = iConicSpiral.GetHeightType() — Получить свойство (**)
iConicSpiral.SetHeightType(heightType) — Установить свойство (**)

initialDiam – Начальный диаметр спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

initialDiam = iConicSpiral.initialDiam — Получить свойство (*)
iConicSpiral.initialDiam = initialDiam — Установить свойство (*)
initialDiam = iConicSpiral.GetInitialDiam() — Получить свойство (**)
iConicSpiral.SetInitialDiam(initialDiam) — Установить свойство (**)

initialDiamType – Способ задания начального диаметра

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: short

Значения свойства:

0 - по размеру,
1 - по объекту.

Синтаксис Automation:

initialDiamType = iConicSpiral.initialDiamType — Получить свойство (*)
iConicSpiral.initialDiamType = initialDiamType — Установить свойство (*)
initialDiamType = iConicSpiral.GetInitialDiamType() — Получить свойство (**)
iConicSpiral.SetInitialDiamType(initialDiamType) — Установить свойство (**)

step – Шаг витков спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

step = iConicSpiral.step		Получить свойство (*)
iConicSpiral.step = step		Установить свойство (*)
step	=	Получить свойство (**)
iConicSpiral.GetStep()		
iConicSpiral.SetStep(step)		Установить свойство (**)

terminalDiam – Конечный диаметр спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

terminalDiam = iConicSpiral.terminalDiam	Получить свойство (*)
iConicSpiral.terminalDiam = terminalDiam	Установить свойство (*)
terminalDiam = iConicSpiral.GetTerminalDiam()	Получить свойство (**)
iConicSpiral.SetTerminalDiam(terminalDiam)	Установить свойство (**)

terminalDiamType – Способ задания конечного диаметра

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: short

Значения свойства:

0	- по размеру,
1	- по объекту,
2	- по углу наклона образующей.

Синтаксис Automation:

terminalDiamType = iConicSpiral.terminalDiamType	Получить свойство (*)
iConicSpiral.terminalDiamType = terminalDiamType	Установить свойство (*)
terminalDiamType = iConicSpiral.GetTerminalDiamType()	Получить свойство (**)
iConicSpiral.SetTerminalDiamType(terminalDiamType)	Установить свойство (**)

tiltAngle – Угол наклона образующей спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

tiltAngle = iConicSpiral.tiltAngle		Получить свойство (*)
iConicSpiral.tiltAngle = tiltAngle		Установить свойство (*)
tiltAngle	=	Получить свойство (**)
iConicSpiral.GetTiltAngle()		
iConicSpiral.SetTiltAngle(tiltAngle)		Установить свойство (**)

tiltAngleHow – Направление отсчета угла наклона образующей

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: BOOL

Значения свойства:

TRUE	- наружу,
FALSE	- внутрь (к оси спирали).

Синтаксис Automation:

tiltAngleHow = iConicSpiral.tiltAngleHow		Получить свойство (*)
iConicSpiral.tiltAngleHow = tiltAngleHow		Установить свойство (*)
tiltAngleHow	=	Получить свойство (**)
iConicSpiral.GetTiltAngleHow()		
iConicSpiral.SetTiltAngleHow(tiltAngleHow)		Установить свойство (**)

turn – Количество витков спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: double

Синтаксис Automation:

turn = iConicSpiral.turn		Получить свойство (*)
iConicSpiral.turn = turn		Установить свойство (*)
turn	=	Получить свойство (**)
iConicSpiral.GetTurn()		
iConicSpiral.SetTurn(turn)		Установить свойство (**)

turnDir – Направление навивки спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Тип данных: BOOL

Значения свойства:

TRUE	- по часовой стрелке,
FALSE	- против часовой стрелки.

Синтаксис Automation:

turnDir = iConicSpiral.turnDir()		Получить свойство (*)
iConicSpiral.turnDir = turnDir		Установить свойство (*)
turnDir	=	Получить свойство (**)
iConicSpiral.GetTurnDir()		
iConicSpiral.SetTurnDir(turnDir)		Установить свойство (**)

IConicSpiralDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- интерфейс математической кривой ksCurve3D или ICurve3D.

GetHeightObject – Получить объект, задающий высоту

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

LPDISPATCH GetHeightObject();

Синтаксис COM:

LPENTITY GetHeightObject();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetInitialDiamObject – Получить объект, задающий начальный диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

LPDISPATCH GetInitialDiamObject();

Синтаксис COM:

LPENTITY GetInitialDiamObject();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetLocation – Получить координаты точки привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

BOOL GetLocation (double* x,

double* y);

Синтаксис COM:

BOOL GetLocation (double* x,

double* y);

Выходные параметры:

x, y - координаты точки привязки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetPlane – Получить указатель на базовую плоскость спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс спирали ksEntity или IEntity.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetTerminalDiamObject – Получить объект, задающий конечный диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

LPDISPATCH GetTerminalDiamObject();

Синтаксис COM:

LPENTITY GetTerminalDiamObject();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

SetHeightObject – Установить объект, задающий высоту

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

BOOL SetHeightObject (LPDISPATCH heightObject);

Синтаксис COM:

BOOL SetHeightObject (LPENTITY heightObject);

Входной параметр:

heightObject	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetInitialDiamObject – Установить объект, задающий начальный диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

BOOL SetInitialDiamObject (LPDISPATCH initialDiamObject);

Синтаксис COM:

BOOL SetInitialDiamObject (LPENTITY initialDiamObject);

Входной параметр:

initialDiamObject	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetLocation – Установить точку привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automatixon:

BOOL SetLocation (double x,
double y);

Синтаксис COM:

BOOL SetLocation (double x,
double y);

Входные параметры:

x, y - координаты точки привязки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPlane – Установить указатель на базовую плоскость спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс спирали ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetTerminalDiamObject – Установить объект, задающий конечный диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Conicheskay_spiral

Синтаксис Automation:

BOOL SetTerminalDiamObject (LPDISPATCH TerminalDiamObject);

Синтаксис COM:

BOOL SetTerminalDiamObject (LPENTITY terminalDiamObject);

Входной параметр:

terminalDiamObject - указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Цилиндрическая спираль (Интерфейсы ksCylindricSpiralDefinition и CylindricSpiralDefinition

Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Интерфейс параметров цилиндрической спирали.

ksCylindricSpiralDefinition - интерфейс Automation
ICylindricSpiralDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICylindricSpiralDefinition – свойства

buildDir – Направление построения спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: BOOL

Значения свойства:

TRUE - нормальное (совпадающее с положительным направлением нормали к базовой плоскости спирали),
FALSE - обратное.

Синтаксис Automation:

buildDir = iCylindricSpiral.buildDir Получить свойство (*)
iCylindricSpiral.buildDir = buildDir Установить свойство (*)
buildDir = iCylindricSpiral.GetBuildDir() Получить свойство (**)
iCylindricSpiral.SetBuildDir(buildDir) Установить свойство (**)

buildMode – Способ построения спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: short

Значения свойства:

0	- по шагу и количеству витков,
1	- по шагу и высоте,
2	- по количеству витков и высоте.

Синтаксис Automation:

buildMode = iCylindricSpiral.buildMode	Получить свойство (*)
iCylindricSpiral.buildMode = buildMode	Установить свойство (*)
buildMode = iCylindricSpiral.GetBuildMode()	Получить свойство (**)
iCylindricSpiral.SetBuildMode(buildMode)	Установить свойство (**)

diam – Диаметр спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

diam = iCylindricSpiral.diam	Получить свойство (*)
iCylindricSpiral.diam = diam	Установить свойство (*)
diam = iCylindricSpiral.GetDiam()	Получить свойство (**)
iCylindricSpiral.SetDiam(diam)	Установить свойство (**)

diamType – Способ задания диаметра

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: short

Значения свойства:

0	- по размеру,
1	- по объекту.

Синтаксис Automation:

diamType = iCylindricSpiral.diamType		Получить свойство (*)
iCylindricSpiral.diamType = diamType		Установить свойство (*)
diamType = iCylindricSpiral.GetDiamType()		Получить свойство (**)
iCylindricSpiral.SetDiamType(diamType)		Установить свойство (**)

firstAngle – Начальный угол (или угол поворота спирали вокруг своей оси)

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

firstAngle = iCylindricSpiral.firstAngle		Получить свойство (*)
iCylindricSpiral.firstAngle = firstAngle		Установить свойство (*)
firstAngle = iCylindricSpiral.GetFirstAngle()		Получить свойство (**)
iCylindricSpiral.SetFirstAngle(firstAngle)		Установить свойство (**)

height – Высота спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

height = iCylindricSpiral.height		Получить свойство (*)
iCylindricSpiral.height = height		Установить свойство (*)
height = iCylindricSpiral.GetHeight()		Получить свойство (**)
iCylindricSpiral.SetHeight(height)		Установить свойство (**)

heightAdd – Дополнительная высота

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

heightAdd = iCylindricSpiral.heightAdd		Получить свойство (*)
iCylindricSpiral.heightAdd = heightAdd		Установить свойство (*)
heightAdd	=	Получить свойство (**)
iCylindricSpiral.GetHeightAdd()		
iCylindricSpiral.SetHeightAdd(heightAd d)		Установить свойство (**)

heightAddHow – Признак дополнительной высоты

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: BOOL

Значения свойства:

TRUE	- продолжить за объект,
FALSE	- не доходя до объекта.

Синтаксис Automation:

heightAddHow = iCylindricSpiral.heightAddHow		Получить свойство (*)
iCylindricSpiral.heightAddHow = heightAddHow		Установить свойство (*)
heightAddHow	=	Получить свойство (**)
iCylindricSpiral.GetHeightAddHow()		
iCylindricSpiral.SetHeightAddHow(heightAddHow)		Установить свойство (**)

Примечание

Данное свойство используется только при способе задания высоты "по объекту" (heightType = 1).

heightType – Способ задания высоты

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: short

Значения свойства:

0	- по размеру,
1	- по объекту.

Синтаксис Automation:

heightType = iCylindricSpiral.heightType		Получить свойство (*)
iCylindricSpiral.heightType = heightType		Установить свойство (*)
heightType	=	Получить свойство (**)
iCylindricSpiral.GetHeightType()		
iCylindricSpiral.SetHeightType(heightTyp e)		Установить свойство (**)

step – Шаг витков спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

step = iCylindricSpiral.step		Получить свойство (*)
iCylindricSpiral.step = step		Установить свойство (*)
step	=	Получить свойство (**)
iCylindricSpiral.GetStep()		
iCylindricSpiral.SetStep(ste p)		Установить свойство (**)

turn – Количество витков спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: double

Синтаксис Automation:

turn	=	Получить свойство (*)
iCylindricSpiral.turn		
iCylindricSpiral.turn	=	Установить свойство (*)
turn		
turn	=	Получить свойство (**)
iCylindricSpiral.GetTurn()		
iCylindricSpiral.SetTurn(t urn)		Установить свойство (**)

turnDir– Направление навивки спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Тип данных: BOOL

Значения свойства:

TRUE - по часовой стрелке,
FALSE - против часовой стрелки.

Синтаксис Automation:

turnDir = iCylindricSpiral.turnDir	Получить свойство (*)
iCylindricSpiral.turnDir = turnDir	Установить свойство (*)
turnDir = iCylindricSpiral.GetTurnDir()	Получить свойство (**)
iCylindricSpiral.SetTurnDir(turnDir)	Установить свойство (**)

ICylindricSpiralDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- интерфейс математической кривой ksCurve3D или ICurve3D.

GetDiamObject – Получить объект, задающий диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

LPDISPATCH GetDiamObject();

Синтаксис COM:

LPENTITY GetDiamObject();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetLocation– Получить координаты точки привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

BOOL GetLocation (double* x,

double* y);

Синтаксис COM:

BOOL GetLocation (double* x,

double* y);

Выходные параметры:

x, y - координаты точки привязки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetHeightObject – Получить объект, задающий высоту

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

LPDISPATCH GetHeightObject();

Синтаксис COM:

LPENTITY GetHeightObject();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetPlane – Получить базовую плоскость спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity эскиза.

SetDiamObject – Установить объект, задающий диаметр

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

BOOL SetDiamObject (LPDISPATCH diamObject);

Синтаксис COM:

BOOL SetDiamObject (LPENTITY diamObject);

Входной параметр:

diamObject - указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetHeightObject – Установить объект, задающий высоту

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

BOOL SetHeightObject (LPDISPATCH heightObject);

Синтаксис COM:

BOOL SetHeightObject (LPENTITY heightObject);

Входной параметр:

heightObject	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetLocation – Установить точку привязки спирали на базовой плоскости (точку пересечения оси спирали с базовой плоскостью)

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

BOOL SetLocation (double x,

double y);

Синтаксис COM:

BOOL SetLocation (double x,

double y);

Входные параметры:

x, y	- координаты точки привязки.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetPlane – Установить базовую плоскость спирали

Интерфейс...Справка системы КОМПАС...

501_Postroenie_spirali.htm#Cilindricheskaja_spir

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Сплайн (Интерфейсы ksSplineDefinition и ISplineDefinition)

Справка системы КОМПАС...

Интерфейс параметров сплайна.

ksSplineDefinition - интерфейс Automation
ISplineDefinition - интерфейс COM

Примечание:

Интерфейс может быть получен от интерфейса ksEntity или IEntity с помощью метода ksEntity::GetDefinition или IEntity::GetDefinition и последующим приведением объекта типа IDispatch или IUnknown к интерфейсу типа ksSplineDefinition или ISplineDefinition соответственно.

ISplineDefinition– свойства

cls – Признак замкнутости сплайна

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - замкнутый сплайн,

FALSE - разомкнутый сплайн.

Синтаксис Automation:

cls = iSpline.cls	Получить свойство (*)
iSpline.cls = cls	Установить свойство (*)
cls = iSpline.GetClosed()	Получить свойство (**)
iSpline.SetClosed(cls)	Установить свойство (**)

degree – Порядок сплайна

Интерфейс...Справка системы КОМПАС...

1091_116_6_Splajn.htm#SPLINE_NURBS_3D

Тип данных: long

Синтаксис Automation:

degree = iSpline.degree	Получить свойство (*)
iSpline.degree = degree	Установить свойство (*)
degree = iSpline.GetDegree()	Получить свойство (**)
iSpline.SetDegree(degree)	Установить свойство (**)

splineOnPoles – Признак способа построения сплайна

Интерфейс...Справка системы КОМПАС...

1091_116_6_Splajn.htm

Тип данных: BOOL

Значения свойства:

TRUE - по полюсам,
FALSE - по вершинам.

Синтаксис Automation:

splineOnPoles = iSpline.splineOnPoles	Получить свойство (*)
iSpline.splineOnPoles = splineOnPoles	Установить свойство (*)
splineOnPoles = iSpline.GetSplineOnPoles()	Получить свойство (**)
splineOnPoles = iSpline.GetSplineOnPoles()	Установить свойство (**)

ISplineDefinition – методы

AddVertex – Добавить вершину

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL AddVertex (double x,

double y,

double z,

double weight);

Синтаксис COM:

BOOL AddVertex (double x,

double y,

double z,

double weight);

Входные параметры:

x	- x координата вершины,
y	- y координата вершины,
z	- z координата вершины,
weight	- вес вершины.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

AddVertexAndAssociation – Добавить вершину по опорному объекту

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL AddVertexAndAssociation (long index, LPDISPATCH obj, double weight);

Синтаксис COM:

BOOL AddVertexAndAssociation (long index, LPENTITY obj, double weight);

Входные параметры:

index	- индекс, с которым добавляется вершина,
obj	- указатель на интерфейс точки ksEntity или IEntity,
weight	- вес вершины.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет добавить в пространственную кривую новую вершину. Координаты новой вершины будут равны координатам опорного объекта obj. Вес вершины равен weight.
2. Объект (точка) obj становится опорным.
3. Если index = -1 или значение index превышает количество вершин, то новая вершина добавляется после последней.

AttachAssociation – Задать опорную точку по индексу

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL AttachAssociation (long index, LPDISPATCH obj);

Синтаксис COM:

BOOL AttachAssociation (long index, LPENTITY obj);

Входные параметры:

index	- индекс вершины в массиве,
obj	- указатель на интерфейс точки ksEntity или IEntity,

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет задать опорный объект для существующей вершины с номером index пространственной кривой. Если объект уже был задан, он будет заменен новым.

DeleteVertex – Удалить вершину с указанным индексом

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL DeleteVertex (long index);

Синтаксис COM:

BOOL DeleteVertex (long index);

Входной параметр:

index - индекс вершины в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Flush – Очистить массив вершин

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL Flush();

Синтаксис COM:

BOOL Flush();

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetAssociation – Получить указатель на опорную точку по индексу

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetAssociation(long index);

Синтаксис COM:

LPENTITY GetAssociation(long index);

Входные параметры:

index - индекс вершины в массиве.

Возвращаемое значение:

- указатель на интерфейс точки ksEntity или IEntity,	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет получить опорный объект для существующей вершины с номером index пространственной кривой, если он был задан.

GetCountVertex – Получить количество вершин

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
long GetCountVertex();
Синтаксис COM:
long GetCountVertex();
Возвращаемое значение:

- количество вершин.

GetParamVertex – Получить параметры вершины с указанным индексом

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
BOOL GetParamVertex (long index,
double* x,
double* y,
double* z,
double* weight);
Синтаксис COM:
BOOL GetParamVertex (long index,
double* x,
double* y,
double* z,
double* weight);
Входной параметр:

inde x	- индекс вершины в массиве.

Выходные параметры:

x	- x координата вершины,
y	- y координата вершины,
z	- z координата вершины,

weight	- вес вершины.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

InsertVertex – Вставить вершину перед указанной

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL InsertVertex (long index,
double x,
double y,
double z,
double weight);

Синтаксис COM:

BOOL InsertVertex (long index,
double x,
double y,
double z,
double weight);

Входные параметры:

index	- индекс вершины в массиве,
x	
x	- x координата вершины,
y	- y координата вершины,
z	- z координата вершины,
weight	- вес вершины.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ReadFromFile – Прочитать параметры вершин из текстового файла

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL ReadFromFile (BSTR fileName);

Синтаксис COM:

BOOL ReadFromFile (LPOLESTR fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание. См. также WriteToFile.

SetAssociation – Задать опорную точку по индексу

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAssociation(long index,
LPDISPATCH obj);

Синтаксис COM:

BOOL SetAssociation(long index,
LPENTITY obj);

Входные параметры:

index - индекс вершины,
obj - указатель на интерфейс IEntity или ksEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

WriteToFile – Записать параметры вершин в текстовый файл

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL WriteToFile (LPCTSTR fileName);

Синтаксис COM:

BOOL WriteToFile (LPOLESTR fileName);

Входные параметр:

fileName - имя файла.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание. См. также ReadFromFile.

Ломаная (Интерфейсы ksPolyLineDefinition и IPolygonalLineDefinition)

Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Интерфейс параметров ломаной.

ksPolyLineDefinition	- интерфейс Automation
IPolygonalLineDefinition	- интерфейс COM

Примечание:

Интерфейс может быть получен от интерфейса ksEntity или IEntity с помощью метода ksEntity::GetDefinition или IEntity::GetDefinition и последующим приведением объекта типа IDispatch или IUnknown к интерфейсу типа ksPolyLineDefinition или IPolygonalLineDefinition соответственно.

IPolygonalLineDefinition – свойства

Closed – Признак замкнутости ломаной

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Тип данных: BOOL

Значения свойства:

TRUE - замкнутая ломаная,
FALSE - разомкнутая ломаная.

Синтаксис Automation:

Closed = iPolyLine.Closed	Получить свойство (*)
iPolyLine.Closed = Closed	Установить свойство(*)
Closed = iPolyLine.GetClosed()	Получить освойство (**)
iPolyLine.SetClosed(Closed)	Установить свойство (**)

Синтаксис COM:

iPolyLine->get_Closed(&closed)	Получить свойство
iPolyLine->put_Closed(closed)	Установить свойство

VertexVisible – Признак отображения свободных вершин

Интерфейс...Тип данных: BOOL

Синтаксис:

vertexVisible = Object.vertexVisible	Получить свойство (*)
Object.vertexVisible = vertexVisible	Установить свойство(*)
vertexVisible = Object.GetVertexVisible()	Получить свойство (**)
Object.SetVertexVisible(vertexVisible)	Установить свойство (**)

IPolygonalLineDefinition – методы

AddPointWithParams – Создать новую вершину

Интерфейс...**Синтаксис Automation**:

LPDISPATCH AddPointWithParams(int index);

Синтаксис COM:

LPPOLYGONALLINEVERTEXPARAM AddPointWithParams(int index);

Входные параметры:

index	- индекс новой вершины.

Возвращаемое значение:

- указатель на интерфейс параметров вершины ломаной IPolygonalLineVertexParam или ksPolyLineVertexParam,

Примечание:

Если индекс равен-1, вершина добавляется в конец.

AddVertex – Добавить вершину

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL AddVertex (double x,

double y,

double z,

double radius);

Синтаксис COM:

BOOL AddVertex (double x,

double y,

double z,

double radius);

Входные параметры:

X	- x координата вершины,
Y	- y координата вершины,
Z	- z координата вершины,
Radius	- радиус скругления в вершине.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

DeleteVertex – Удалить вершину с указанным индексом

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL DeleteVertex (long index);

Синтаксис COM:

BOOL DeleteVertex (long index);

Входной параметр:

Index	- индекс вершины в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

EdgeCollection – Получить массив ребер

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

LPDISPATCH EdgeCollection();

Синтаксис COM:

LPEDGECOLLECTION EdgeCollection();

Возвращаемое значение:

Указатель на интерфейс ksEdgeCollection или IEdgeCollection, - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Ребра в коллекции лежат по порядку следования в ломаной.

Flush – Очистить массив вершин ломаной

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL Flush();

Синтаксис COM:

BOOL Flush();

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- интерфейс математической кривой ksCurve3D или ICurve3D.

GetCountVertex – Получить количество вершин ломаной

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

long GetCountVertex();

Синтаксис COM:

long GetCountVertex();

Возвращаемое значение:

- Количество вершин.

GetParamVertex – Получить параметры вершины с указанным индексом

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL GetParamVertex (long index,

double* x,

double* y,

double* z,

double* radius);

Синтаксис COM:

BOOL GetParamVertex (long index,

double* x,

double* y,

double* z,

double* radius);

Входной параметр:

Index - индекс вершины в массиве.

Выходные параметры:

X - x координата вершины,
Y - y координата вершины,
Z - z координата вершины,
Radius - радиус скругления в вершине.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetPointParams – Получить параметры вершины ломаной

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPointParams(int index);

Синтаксис COM:

LPPOLYGONALLINEVERTEXPARAM GetPointParams(int index);

Входные параметры:

index - индекс вершины.

Возвращаемое значение:

- указатель на интерфейс параметров вершины ломаной IPolygonalLineVertexParam или ksPolyLineVertexParam,

InsertVertex – Вставить вершину перед указанной

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL InsertVertex (long index,

double x,

double y,

double z,

double radius);

Синтаксис COM:

BOOL InsertVertex (long index,

double x,

double y,

double z,

double radius);

Входные параметры:

Index - индекс вершины в массиве,
X - x координата вершины,
Y - y координата вершины,
Z - z координата вершины,
Radius - радиус скругления в вершине.

Возвращаемое значение:

TRUE - в случае успешного завершения.

ReadFromFile – Прочитать параметры вершин из текстового файла

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL ReadFromFile (BSTR fileName);

Синтаксис COM:

BOOL ReadFromFile (LPOLESTR fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

См. также WriteToFile

WriteToFile – Записать параметры вершин в текстовый файл

Интерфейс...Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Синтаксис Automation:

BOOL WriteToFile (LPCTSTR fileName);

Синтаксис COM:

BOOL WriteToFile (LPOLESTR fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

См. также ReadFromFile

Интерфейсы поверхностей

Поверхность выдавливания (Интерфейсы ksExtrusionSurfaceDefinition и IExtrusionSurfaceDefinition)

Интерфейс параметров поверхности выдавливания.

ksExtrusionSurfaceDefinition	- интерфейс Automation
IExtrusionSurfaceDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksExtrusionSurfaceDefinition – свойства

closedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: VARIANT_BOOL

Значения свойства:

TRUE	- поверхность замкнута,
FALSE	- поверхность не замкнута.

Синтаксис Automation:

closedShell = iExtrusion.closedShell	Получить свойство (*)
iExtrusion.closedShell = closedShell	Установить свойство(*)
closedShell = iExtrusion.GetClosedShell()	Получить свойство (**)
iExtrusion.SetClosedShell(closedShell)	Установить свойство (**)

Синтаксис COM:

closedShell = iExtrusion.GetClosedShell()	Получить свойство.
iExtrusion.SetClosedShell(closedShell)	Установить свойство.

directionType – Направление выдавливания

Интерфейс...

Тип данных: short

Синтаксис Automation:

directionType iExtrusionSurface.directionType	=	Получить свойство (*)
iExtrusionSurface.directionType= directionType		Установить свойство(*)
directionType iExtrusionSurface.GetDirectionType()	=	Получить свойство (**)
iExtrusionSurface.SetDirectionType(direc tionType)		Установить свойство (**)

Значения свойства:

Типы направлений...

Примечание:

1. Нормаль, проведенная к грани, всегда направлена наружу ("из тела детали").
2. Прямое направление совпадает с нормалью, проведенной к плоскости эскиза.
3. Для вырезаемого элемента выдавливания направление противоположно нормали.

ksExtrusionSurfaceDefinition – методы

ExtrusionParam – Получить указатель на интерфейс параметров элемента выдавливания

Интерфейс...

Синтаксис Automation:

LPDISPATCH ExtrusionParam();

Синтаксис COM:

LPEXTRUSIONPARAM ExtrusionParam();

Возвращаемое значение:

- указатель на интерфейс объекта ksExtrusionParam или IExtrusionParam.

GetDepthObject – Получить объект, задающий глубину выдавливания

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDepthObject (VARIANT_BOOL normal);

Синтаксис COM:

LPENTITY GetDepthObject (BOOL normal);

Входные параметры:

normal - направление выдавливания:
TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза.

Возвращаемое значение:

- указатель на интерфейс объекта глубины выдавливания ksEntity или IEntity.

GetSideParam – Получить параметры выдавливания в одном направлении

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL GetSideParam (VARIANT_BOOL forward,
short *type,
double *depth,
double *draftValue,
VARIANT_BOOL *draftOutward);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,
short *type,
double *depth,
double *draftValue,
BOOL *draftOutward);

Входные параметры:

forward - направление выдавливания:
TRUE - прямое направление,
FALSE - обратное направление.

Выходные параметры:

type - тип выдавливания,
depth - глубина выдавливания,
draftValue - угол уклона,

draftOutward - направление уклона:
FALSE - уклон наружу,
TRUE - уклон внутрь.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetSketch – Получить указатель на интерфейс эскиза поверхности выдавливания

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

ResetDepthObject – Отказаться от установленного методом SetDepthObject объекта, задающего глубину выдавливания

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL ResetDepthObject (VARIANT_BOOL normal);

Синтаксис COM:

BOOL ResetDepthObject (BOOL normal);

Входные параметры:

normal - направление выдавливания:
TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetDepthObject – Установить объект, задающий глубину выдавливания

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetDepthObject (VARIANT_BOOL normal, LPDISPATCH obj);

Синтаксис COM:

BOOL SetDepthObject (BOOL normal, LPENTITY obj);

Входные параметры:

normal	- направление выдавливания: TRUE - совпадает с направлением нормали к плоскости эскиза, FALSE - противоположно направлению нормали к плоскости эскиза.
obj	- указатель на интерфейс объекта глубины выдавливания ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetSideParam – Изменить параметры выдавливания в одном направлении

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetSideParam (VARIANT_BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Входные параметры:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.
type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetSketch – Установить эскиз поверхности выдавливания

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Кинематическая поверхность (Интерфейсы ksEvolutionSurfaceDefinition и IEvolutionSurfaceDefinition)

Интерфейс параметров кинематической поверхности.

ksEvolutionSurfaceDefinition	- интерфейс Automation
IEvolutionSurfaceDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksEvolutionSurfaceDefinition – свойства

closedShel – Признак замкнутости поверхности

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

closedShell = iEvolution.closedShell		Получить свойство (*)
iEvolution.closedShell = closedShell		Установить свойство(*)
closedShell	=	Получить свойство (**)
iEvolution.GetClosedShell()		
iEvolution.SetClosedShell(closedShell)		Установить свойство (**)

Синтаксис COM:

closedShell	=	Получить свойство
iEvolution.GetClosedShell()		
iEvolution.SetClosedShell(closedShell)		Установить свойство

Значения свойства:

TRUE - поверхность замкнута,
FALSE - поверхность не замкнута.

sketchShiftType – Тип движения сечения по траектории

Интерфейс..

Тип данных: short

Синтаксис Automation:

sketchShiftType	=	Получить свойство (*)
iEvolution.sketchShiftType		
iEvolution.sketchShiftType	=	Установить свойство(*)
sketchShiftType		
sketchShiftType	=	Получить свойство (**)
iEvolution.GetSketchShiftType()		
iEvolution.SetSketchShiftType(sketchS hiftType)		Установить свойство (**)

Синтаксис COM:

sketchShiftType	=	Получить свойство
iEvolution.GetSketchShiftType()		

iEvolution.SetSketchShiftType(sketchShiftType)	Установить свойство

Значения свойства:

0	- образующая переносится параллельно самой себе,
1	- образующая при переносе сохраняет исходный угол с направляющей,
2	- плоскость образующей выставляется и сохраняется ортогональной направляющей.

ksEvolutionSurfaceDefinition – методы

GetPathLength – Получить длину результирующей кривой траектории

Интерфейс..**Синтаксис Automation**:

double GetPathLength (unsigned long bitVector);

Синтаксис COM:

double GetPathLength(unsigned long bitVector);

Входные параметры:

bitVector	- единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

длина результирующей кривой траектории	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

Метод работает только на уже построенной операции.

GetSketch – Получить указатель на интерфейс эскиза кинематической поверхности

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

PathPartArray – Получить указатель на интерфейс массива кривых, задающих траекторию движения сечения кинематического элемента

Интерфейс..

Синтаксис Automation:

IDispatch * PathPartArray();

Синтаксис COM:

LPENTITYCOLLECTION PathPartArray();

Возвращаемое значение:

- указатель на интерфейс ksEntityCollection или IEntityCollection.

SetSketch – Установить эскиз кинематической поверхности

Интерфейс..

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входные параметры:

sketch - указатель на интерфейс ksEntity или IEntity

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Поверхность вращения (Интерфейсы ksRotatedSurfaceDefinition и IRotatedSurfaceDefinition)

Интерфейс параметров поверхности вращения.

ksRotatedSurfaceDefinition - интерфейс Automation
IRotatedSurfaceDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksRotatedSurfaceDefinition – свойства

closedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: VARIANT_BOOL

Значения свойства:

TRUE	- поверхность замкнута,
FALSE	- поверхность не замкнута.

Синтаксис Automation:

closedShell = iRotated.closedShell		Получить свойство (*)
iRotated.closedShell = closedShell		Установить свойство(*)
closedShell	=	Получить свойство (**)
iRotated.GetClosedShell()		
iRotated.SetClosedShell(closedShell)		Установить свойство (**)

Синтаксис COM:

closedShell	=	Получить свойство.
iRotated.GetClosedShell()		
iRotated.SetClosedShell(closedShell)		Установить свойство.

directionType – Направление вращения

Интерфейс...

Тип данных: short

Значения свойства:

Типы направлений...

Синтаксис Automation:

directionType	=	Получить свойство (*)
iRotatedSurface.directionType		
iRotatedSurface.directionType	=	Установить свойство(*)
directionType		

directionType	=	Получить свойство (**)
iRotatedSurface.GetDirectionType()		
iRotatedSurface.SetDirectionType (directionType)		Установить свойство (**)

Примечание:

1. Прямое направление - вдоль нормали к плоскости эскиза и всегда против часовой стрелки.
2. Нормаль, проведенная к поверхности тела, всегда выходит из тела.

toroidShapeType – Признак тороида

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- тороид,
FALSE	- сфероид.

Синтаксис Automation:

toroidShapeType	=	Получить
iRotatedSurface.toroidShapeType		свойство (*)
iRotatedSurface.toroidShapeType	=	Установить
toroidShapeType		свойство(*)
toroidShapeType	=	Получить
iRotatedSurface.GetToroidShapeType()		свойство (**)
iRotatedSurface.SetToroidShapeType		Установить
(toroidShapeType)		свойство (**)

ksRotatedSurfaceDefinition – методы

GetSideParam – Получить параметры вращения в одном направлении

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL GetSideParam (VARIANT_BOOL forward,

double *angle);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,

double * angle);

Входные параметры:

forward - направление вращения:
TRUE - прямое направление,
FALSE - обратное направление.

Выходные параметры:

angle - угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetSketch – Получить указатель на интерфейс эскиза поверхности вращения

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

RotatedParam – Получить указатель на интерфейс параметров элемента вращения

Интерфейс...

Синтаксис Automation:

LPDISPATCH RotatedParam();

Синтаксис COM:

LPROTATEDPARAM RotatedParam();

Возвращаемое значение:

- указатель на интерфейс объекта ksRotatedParam или IrotatedParam.

SetSideParam – Изменить параметры вращения в одном направлении

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetSideParam (short forward,

double angle);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

double angle);

Входные параметры:

forward	- направление вращения: TRUE - прямое направление, FALSE - обратное направление,
angle	- угол вращения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetSketch – Установить эскиз поверхности вращения

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Поверхность по сечениям(Интерфейсы ksLoftSurfaceDefinition и ILoftSurfaceDefinition)

Интерфейс параметров поверхности по сечениям.

ksLoftSurfaceDefinition - интерфейс Automation
ILoftSurfaceDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksLoftSurfaceDefinition – свойства

closedShell – Признак замкнутости поверхности

Интерфейс...

Тип данных: VARIANT_BOOL

Значения свойства:

TRUE - поверхность замкнута,
FALSE - поверхность не замкнута.

Синтаксис Automation:

```
closedShell = iLoft.closedShell          Получить свойство (* )
iLoft.closedShell = closedShell          Установить свойство(* )
closedShell = iLoft.GetClosedShell()     Получить свойство (**)
iLoft.SetClosedShell(closedShell)        Установить свойство (**)
```

Синтаксис COM:

```
closedShell = iLoft.GetClosedShell()     Получить свойство.
iLoft.SetClosedShell(closedShell)        Установить свойство.
```

ksLoftSurfaceDefinition – методы

GetLoftParam – Получить параметры элемента по сечениям

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL GetLoftParam (VARIANT_BOOL *closed,

VARIANT_BOOL *flipVertex,

VARIANT_BOOL *autoPath);

Синтаксис COM:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Выходные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint,

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetDirectionalLine – Получить направляющую линию. Эскиз в котором лежит кривая

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDirectionalLine();

Синтаксис COM:

LPENTITY GetDirectionalLine();

Возвращаемое значение:

TRUE	- указатель на интерфейс ksEntity или Ientity.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

SetDirectionalLine – Задать направляющую линию. Эскиз в котором лежит кривая

Интерфейс...

Синтаксис Automation:

BOOL SetDirectionalLine (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetDirectionalLine (LPENTITY sketch);

Входные параметры:

sketch - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

SetLoftParam – Изменить параметры операции по сечениям

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetLoftParam (VARIANT_BOOL closed,

VARIANT_BOOL flipVertex,

VARIANT_BOOL autoPath);

Синтаксис COM:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Выходные параметры:

Closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint,

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Sketches – Получить указатель на интерфейс массива эскизов элемента по сечениям

Интерфейс...

Синтаксис Automation:

LPDISPATCH Sketches();

Синтаксис COM:

LPENTITYCOLLECTION Sketches();

Возвращаемое значение:

TRUE	- указатель на интерфейс массива эскизов ksEntityCollection или IEntityCollection.
FALSE	- в случае неудачи.

Примечание:

1. В массиве включен контроль, не позволяющий добавить в него нулевой указатель на эскиз.
2. Эскизы из данного массива используются для построения элемента по сечениям.

Интерфейс операции вычитания компонентов (ksMoldCavityDefinition, IMoldCavityDefinition)

Интерфейс операции вычитания компонентов.

ksMoldCavityDefinition	- интерфейс Automation
IMoldCavityDefinition	- интерфейс COM

Примечания.

1. Позволяет создать полость, имеющую форму другой детали.
2. Операция доступна в режиме редактирования детали в контексте сборки.
3. Для входа в режим редактирования детали в контексте сборки используется функция ksPart::BeginEdit.
4. В операции вычитания могут участвовать только детали. Вычесть из детали подсборку невозможно.
5. Если необходимо, чтобы размеры создаваемой полости отличались от размеров вычитаемой детали, нужно задать коэффициент масштабирования в процентах scale.
6. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksMoldCavityDefinition – свойства

scale – Коэффициент масштабирования, %.

Интерфейс...Тип данных: double

Синтаксис Automation:

scale = iObject.scale	Получить свойство (*)
iObject.scale = scale	Установить свойство(*)
scale = iObject.GetScale()	Получить свойство (**)
iObject.SetScale (scale)	Установить свойство (**)

ksMoldCavityDefinition – методы

GetScaleCentre – Получить вершину, относительно которой выполняется масштабирование

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetScaleCentre();

Синтаксис COM:

LPENTITY GetScaleCentre();

Возвращаемое значение:

	- указатель на интерфейс коллекции компонентов ksEntity или IEntity,
NULL	- если вершина не задана.

Примечания.

В качестве вершины могут использоваться характерные точки графических объектов в эскизах, (например, конец отрезка, центр окружности и т.п.) или начала координат. Если вершина не указана, то используется точка, являющаяся центром масс деталей.

PartArray – Получить массив вычитаемых компонентов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

- Указатель на интерфейс коллекции компонентов ksPartCollection или IPartCollection.

SetScaleCentre – Установить вершину, относительно которой выполняется масштабирование

Интерфейс...**Синтаксис Automation**:

VARIANT_BOOL SetScaleCentre(LPDISPATCH vert);

Синтаксис COM:

BOOL SetScaleCentre(LPENTITY vert);

Входные параметры:

vert	- Указатель на интерфейс коллекции компонентов ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи

Примечания.

В качестве вершины могут использоваться характерные точки графических объектов в эскизах (например, конец отрезка, центр окружности и т.п.) или начала координат. Если вершина не указана, то используется точка, являющаяся центром масс деталей.

Интерфейс макроэлемента (ksMacro3DDefinition, IMacro3DDefinition)

Справка системы КОМПАС...

Интерфейс макроэлемента документа-модели.

ksMacro3DDefinition	- интерфейс Automation
IMacro3DDefinition	- интерфейс COM

Описание:

Является 3D объектом, объединяющим в себе другие 3D объекты, в том числе и другие макроэлементы 3D, с возможностью скрывать свой состав, сохранять дополнительные пользовательские параметры и редактировать данный объект через библиотеку (если задано имя файла, имя библиотеки и команда редактирования).

Последовательность создания макроэлементов через API.

Создание макроэлемента с последующим добавлением объектов в его состав.

1. Создание пустого макроэлемента:

▼ Создание у компонента макроэлемента (ksPart::NewEntity соответствующего типа).

- Установка свойств макроэлемента (Видимость состава StaffVisible).
- Создание макро в модели ksEntity::Create.
- Установка пользовательских параметров SetUserParam.

2. Добавление существующих объектов в создаваемый макроэлемент:

- Добавление объектов в коллекцию объектов, входящих в макро ksFeatureCollection; при этом в модели ничего не происходит.
- Обновление макроэлемента ksEntity::Update; все новые объекты будут перенесены в макро в модели, у самих объектов тоже будут вызваны методы Update,

3. Создание новых объектов и добавление в макро:

- Создание нового объекта (NewEntity).
- Задание свойств нового объекта.
- Добавление объекта в коллекцию объектов, входящих в макро, при этом в модели ничего не происходит.
- Обновление макро ksEntity::Update - все новые, не созданные объекты, создадутся; у них вызовется ksEntity::Create.

  Создание макроэлемента с одновременным наполнением его объектами.

  - Создание у компонента макроэлемента (NewEntity соответствующего типа).
  - Задание свойств макроэлемента.
  - Создание при необходимости новых объектов (NewEntity) и задание их свойства (Create не вызывать), иначе переход к следующему пункту,
  - Добавление объектов в коллекцию объектов, входящих в макро, в модели ничего не происходит,

- Создание макро в модели (ksEntity::Create) - все новые объекты создадутся, у них вызовется метод ksEntity::Create. Существующие объекты будут обновлены, у них вызовется метод ksEntity::Update.

4. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IMacro3DDefinition – свойства

DoubleClickEditOff – Получить признак внешнего редактирования детали

Интерфейс...Тип данных: BOOL

Синтаксис Automation:

```
off = iObj.DoubleClickEditOff                Получить свойство (* )
iObj.DoubleClickEditOff = off                Установить свойство(* )
off = iObj.GetDoubleClickEditOff()           Получить свойство (**)
iObj.SetDoubleClickEditOff (off)             Установить свойство (**)
```

Примечание:

Свойство позволяет для библиотечных элементов, имеющих макропараметры, оставить стандартное поведение компонента и объектов, входящих в состав компонента. Для этого нужно установить значение свойства равным TRUE.

PropertyObjectEditable – Поддерживается интерфейс внешних свойств объекта

Интерфейс...Тип данных: BOOL

Синтаксис Automation:

PropertyObjectEditable = Object.PropertyObjectEditable	=	Получить свойство (*)
Object.PropertyObjectEditable = PropertyObjectEditable	=	Установить свойство(*)
PropertyObjectEditable = Object.GetPropertyObjectEditable()	=	Получить свойство (**)
Object.SetPropertyObjectEditable(PropertyObjectEditable)		Установить свойство (**)

StaffVisible – Управление видимостью состава

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Синтаксис Automation:

staffVisible = iObject.StaffVisible		Получить свойство (*)
iObject.StaffVisible = staffVisible		Установить свойство(*)
staffVisible = iObject.GetStaffVisible()	=	Получить свойство (**)
iObject.SetStaffVisible (staffVisible)		Установить свойство (**)

Синтаксис COM:

staffVisible = iObject->GetStaffVisible()		Получить свойство
iObject->SetStaffVisible (&StaffVisible)		Установить свойство

Значение свойства:

TRUE	- состав объекта можно посмотреть в Дереве построения,
FALSE	- состав недоступен для просмотра.

Примечание:

Свойство позволяет управлять видимостью состава макроэлемента.

IMacro3DDefinition – методы

Add – Добавить объект в состав макроэлемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL Add (LPDISPATCH obj);

Синтаксис COM:

BOOL Add (LPUNKNOWN obj);

Входные параметры:

obj	- указатель на интерфейс объекта, добавляемого в макроэлемент.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. В качестве входного параметра могут быть переданы следующие интерфейсы:

- ksFeature,
- ksEntity,
- ksPart,
- ksMateConstraint.

2. Если объект еще не создан в модели, то он создастся после вызова методов ksEntity::Create или ksEntity::Update макроэлемента. Для существовавших объектов в этом случае будет вызван метод ksEntity::Update .
3. Добавляемый объект должен принадлежать тому же компоненту, что и сам макроэлемент. Он не должен принадлежать другому макроэлементу.
4. При добавлении тела в макроэлемент в макроэлемент добавляются все операции данного тела на момент создания макроэлемента. Само тело в макроэлемент не добавляется.

ClearAllObj – Удалить все вспомогательные объекты, сохраненные в макро

Интерфейс...**Синтаксис Automation**:

BOOL ClearAllObj();

Синтаксис COM:

BOOL ClearAllObj();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

Destroy – Разрушить макроэлемент

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL Destroy();

Синтаксис COM:

BOOL Destroy();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

После вызова этого метода макроэлемент будет разрушен, а все объекты, входившие в его состав, станут самостоятельными.

FeatureCollection – Получить массив объектов, входящих в макроэлемент

Интерфейс...**Синтаксис Automation**:

LPDISPATCH FeatureCollection();

Синтаксис COM:

LPFEATURECOLLECTION FeatureCollection();

Возвращаемое значение:

- указатель на интерфейс ksFeatureCollection или IFeatureCollection

Примечание:

1. Через данную коллекцию можно добавлять только уже созданные объекты. Добавление новых (несозданных) объектов осуществляется при помощи метода ksMacro3DDefinition::Add.
2. Метод ksFeatureCollection::Refresh наполняет содержимое коллекции объектами, входящими в макроэлемент. Все изменения, сделанные в коллекции, после вызова этого метода будут потеряны.
3. При добавлении объектов в коллекцию добавляемый объект должен принадлежать тому же компоненту, что и сам макроэлемент; он не должен принадлежать другому макроэлементу.

GetCountObj – Получить количество вспомогательных объектов, сохраненных в макро

Интерфейс...**Синтаксис Automation**:

long GetCountObj();

Синтаксис COM:

long GetCountObj();

Возвращаемое значение:

- Количество вспомогательных объектов, сохранённых в макро.

Примечания:

Объект - это вспомогательная информация и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

GetObject – Получить указатель на вспомогательный объект, сохраненный в макро по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetObject (long index);

Синтаксис COM:

LPUNKNOWN GetObject (long index);

Входные параметры:

index - номер объекта.

Возвращаемое значение:

- Указатель на интерфейс IDispatch или IUnknown сохранённого объекта.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

См. также: IMacro3DDefinition::SetObject

GetUserLibraryCommand – Получить номер команды пользовательской библиотеки, при помощи которой можно редактировать макроэлемент

Интерфейс.

Аналог данного метода при использовании Automation - ksUserParam::number.

Пример...

Синтаксис COM:

long GetUserLibraryCommand();

Возвращаемое значение:

Номер библиотечной команды	- в случае успешного завершения,
-1	- если библиотеки нет.

GetUserLibraryFileName – Получить имя файла пользовательской библиотеки, при помощи которой можно редактировать макроэлемент

Интерфейс...Аналог данного метода при использовании Automation - ksUserParam::fileName.

Пример...

Синтаксис COM:

LPOLESTR GetUserLibraryFileName();

Возвращаемое значение:

Имя файла пользовательской библиотеки	- в случае успешного завершения,
NULL	- если имя файла библиотеки не задано.

GetUserLibraryName – Получить имя пользовательской библиотеки, при помощи которой можно редактировать макроэлемент

Интерфейс...

Аналог данного метода при использовании Automation - ksUserParam::libName.

Пример....

Синтаксис COM:

LPOLESTR GetUserLibraryName();

Возвращаемое значение:

Имя пользовательской библиотеки	- в случае успешного завершения,
NULL	- если имя библиотеки не задано.

GetUserParam – Получить параметры пользователя

Интерфейс...**Синтаксис Automation**:

BOOL GetUserParam (LPDISPATCH userPars);

Выходные параметры:

user Pars	- указатель на интерфейс пользовательских параметров ksUserParam.

Синтаксис COM:

BOOL GetUserParam (void *value,
unsigned int size);

Входные параметры:

value	- указатель на пользовательскую структуру параметров,
size	- размер структуры параметров.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Способ получения пользовательских данных (userPars) должен совпадать со способом их сохранения (void*, SAFEARRAY, или DynamicArray) через ksMacro3DDefinition::SetUserParam.

- Если пользовательские данные были сохранены через SAFEARRAY (userPars), то перед их получением нужно создать SAFEARRAY соответствующего размера (размер данных можно определить при помощи ksMacro3DDefinition::GetUserParamSize).
- Если пользовательские данные были сохранены через ksUserParam::SetUserArray, то перед их получением нужно создать UserArray, аналогичный по структуре используемому при сохранении, и передать его в ksUserParam::SetUserArray.

GetUserParamSize – Получить размер структуры параметров пользователя, хранимых в макроэлементе

Интерфейс...**Синтаксис Automation**:

long GetUserParamSize();

Синтаксис COM:

long GetUserParamSize();

Возвращаемое значение:

- Размер структуры параметров пользователя, хранимых в макроэлементе в байтах.

SetObject – Сохранить указатель на объект в макро по индексу

Интерфейс...**Синтаксис Automation**:

BOOL SetObject(long index, LPDISPATCH obj);

Синтаксис COM:

BOOL SetObject(long index, LPUNKNOWN obj);

Входные параметры:

index - номер объекта,
obj - указатель на интерфейс объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Объект - это вспомогательная информация, и визуально никак не отображается. Например, если создать болт как макро относительно какого-нибудь объекта (например отверстие - цилиндрическая поверхность), и запомнить это отверстие в макро болта, то при следующем редактирование можно получить сохранённый указатель на отверстие.

См. также: IMacro3DDefinition::GetObject

SetUserParam – Установить параметры пользователя

Интерфейс...**Синтаксис Automation**:

BOOL SetUserParam (LPDISPATCH userPars);

Входные параметры:

userPars	- указатель на интерфейс пользовательских параметров ksUserParam.

Синтаксис COM:

BOOL SetUserParam (void *value,
unsigned int size,
LPOLESTR nameFile,
LPOLESTR nameLib,
int number);

Входные параметры:

value	- указатель на пользовательскую структуру параметров,
size	- размер структуры параметров,
nameFile	- имя файла прикладной библиотеки для последующего редактирования через библиотеку (0 - запоминается текущая библиотека),
nameLib	- имя прикладной библиотеки для последующего редактирования через библиотеку (0 - запоминается текущая библиотека),
number	- номер команды в прикладной библиотеке для последующего редактирования через библиотеку (-1 - запоминается текущая команда).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Данный метод позволяет сохранить параметры пользователя для последующего редактирования с помощью библиотеки.

Интерфейс операции объединения компонентов (ksUnionComponentsDefinition и IUnionComponentsDefinition)

Справка системы КОМПАС...

Интерфейс операции объединения компонентов.

ksUnionComponentsDefinition	- интерфейс Automation
IUnionComponentsDefinition	- интерфейс COM

Примечания.

1. Позволяет создать объединение двух или более деталей, входящих в сборку в новой или уже существующей детали.
2. Операция доступна в режиме редактирования детали в контексте сборки.
3. Для входа в режим редактирования детали в контексте сборки используется функция ksPart::BeginEdit.
4. В операции объединения могут участвовать только детали. Объединить деталь с подсборкой невозможно.
5. При необходимости вы можете скрыть или исключить из расчета детали, объединением которых является новая деталь.
6. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksUnionComponentsDefinition – методы

PartArray – Получить массив объединяемых компонентов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

- Указатель на интерфейс коллекции компонентов ksPartCollection или IPartCollection.

Интерфейсы копирования

Интерфейс массива удаленных индексов для операций копирования и массивов компонентов (ksDeletedCopyCollection и IDeletedCopyCollection)

ksDeletedCopyCollection	- интерфейс Automation
IDeletedCopyCollection	- интерфейс COM

Примечание:

1. Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.
2. Данный интерфейс можно получить от следующих интерфейсов:

▼ Интерфейсы копирования компонентов сборки. Интерфейсы копирования.

ksDeletedCopyCollection – методы

Add – Добавить объект в конец массива

Интерфейс...**Синтаксис Automation**:

BOOL Add (long index1, long index2);

Синтаксис COM:

BOOL GetByIndex (long index, long * index1, long * index2);

Входные параметры:

index1, index2	- индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

AddAt – Добавить элемент перед элементом с заданным индексом

Интерфейс...Справка системы КОМПАС...

`Glava_48_Obzhie_svedeniy.`

Синтаксис Automation:

BOOL AddAt (long index1, long index2, long index);

Синтаксис COM:

BOOL AddAt (long index1, long index2, long index);

Входные параметры:

index1, index2	- индексы удаленной копии в узоре,
index	- удаленной копии в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Clear – Очистить массив

Интерфейс...**Синтаксис Automation**:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

DetachByBody – Отсоединить элемент

Интерфейс...**Синтаксис Automation**:

BOOL DetachByBody (long index1, long index2);

Синтаксис COM:

BOOL DetachByBody (long index1, long index2);

Входные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

DetachByIndex – Отсоединить элемент по индексу

Интерфейс...**Синтаксис Automation**:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входной параметр:

index - индекс удаленной копии в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Элемент массива из модели не удаляется.

FindIt – Получить индекс элемента в массиве

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

long FindIt (long index1, long index2);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
-1 - если элемент не найден.

Синтаксис COM:

unsigned long FindIt (long index1, long index2);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
0xffffffff - если элемент не найден.

Входные параметры:

index1, index2 - индексы удаленной копии в узоре.

First – Получить индексы первой удаленной копии

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL First(long * index1, long * index2);

Синтаксис COM:

BOOL First(long * index1, long * index2);

Выходные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetByIndex – Получить индексы удаленной копии по индексу

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL GetByIndex (long index, long * index1, long * index2);

Синтаксис COM:

BOOL GetByIndex (long index, long * index1, long * index2);

Входной параметр:

index - индекс удаленной копии в массиве.

Выходные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetCount – Получить количество элементов в массиве

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

Last – Получить индексы последней удаленной копии

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL Last (long * index1, long * index2);

Синтаксис COM:

BOOL Last (long * index1, long * index2);

Выходные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Next – Получить индексы следующей удаленной копии

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL Next (long * index1, long * index2);

Синтаксис COM:

BOOL Next (long * index1, long * index2);

Выходные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Prev – Получить индексы предыдущей удаленной копии

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL Prev (long * index1, long * index2);

Синтаксис COM:

BOOL Prev (long * index1, long * index2);

Выходные параметры:

index1, index2 - индексы удаленной копии в узоре.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Refresh – Обновить массив

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetByIndex – Заменить элемент

Интерфейс...Справка системы КОМПАС...

Glava_48_Obzhie_svedeniy.htm

Синтаксис Automation:

BOOL SetByIndex (long index1, long index2, long index);

Синтаксис COM:

BOOL SetByIndex (long index1, long index2, long index);

Входные параметры:

index1, index2 - индексы удаленной копии в узоре,
index - удаленной копии в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Копия элементов детали по параллелограммной сетке(Интерфейсы ksMeshCopyDefinition и ICopyMeshDefinition)

Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main
Интерфейс операции копирования по сетке.

ksMeshCopyDefinition	- интерфейс Automation
ICopyMeshDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICopyMeshDefinition – свойства

angle1 – Угол наклона первой оси в текущей системе координат

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main
Тип данных: double

Синтаксис Automation:

angle1 = iCopyMesh.angle1		Получить свойство(*)
iCopyMesh.angle1 = angle1		Установить свойство (*)
angle1	=	Получить свойство (**)
iCopyMesh.GetAngle1()		
iCopyMesh.SetAngle1(angle1)		Установить свойство (**)

angle2 – Угол наклона второй оси относительно первой

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main
Тип данных: double

Синтаксис Automation:

angle2 = iCopyMesh.angle2		Получить свойство(*)
iCopyMesh.angle2 = angle2		Установить свойство (*)
angle2	=	Получить свойство (**)
iCopyMesh.GetAngle2()		
iCopyMesh.SetAngle2(angle2)		Установить свойство (**)

count1 – Количество копий вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: long

Синтаксис Automation:

count1 = iCopyMesh.count1		Получить свойство(*)
iCopyMesh.count1 = count1		Установить свойство (*)
count1	=	Получить свойство (**)
iCopyMesh.GetCount1()		
iCopyMesh.SetCount1(count1)		Установить свойство (**)

count2 – Количество копий вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: long

Синтаксис Automation:

count2 = iCopyMesh.count2		Получить свойство(*)
iCopyMesh.count2 = count2		Установить свойство (*)
count2	=	Получить свойство (**)
iCopyMesh.GetCount2()		
iCopyMesh.SetCount2(count2)		Установить свойство (**)

factor1 – Признак полного шага вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step1 устанавливает расстояние между крайними элементами вдоль первой оси,
FALSE	- свойство step1 устанавливает расстояние между соседними элементами вдоль первой оси.

Синтаксис Automation:

factor1 = iCopyMesh.factor1		Получить свойство(*)
iCopyMesh.factor1 = factor1		Установить свойство (*)
factor1	=	Получить свойство (**)
iCopyMesh.GetFactor1()		
iCopyMesh.SetFactor1(factor1)		Установить свойство (**)

factor2 – Признак полного шага вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step2 устанавливает расстояние между крайними элементами вдоль второй оси,
FALSE	- свойство step2 устанавливает расстояние между соседними элементами вдоль второй оси.

Синтаксис Automation:

factor2 = iCopyMesh.factor2		Получить свойство(*)
iCopyMesh.factor2 = factor2		Установить свойство (*)
factor2	=	Получить свойство (**)
iCopyMesh.GetFactor2()		
iCopyMesh.SetFactor2(factor2)		Установить свойство (**)

geomArray – Признак использования геометрического копирования

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: BOOL

Значения свойства:

TRUE	- использовать геометрическое копирование. Содержимое операций-источников (грани, ребра) копируется без пересчета. В простых случаях операция ускоряется, но возможны ошибки при копировании скруглений и фасок на криволинейных поверхностях,
FALSE	- при копировании все операции-источники строятся заново, т.е. производится полноценный расчет.

Синтаксис Automation:

geomArray	=	Получить свойство(*)
ICopyMeshDefinition.geomArray		
ICopyMeshDefinition.geomArray	=	Установить свойство (*)
geomArray		
geomArray	=	Получить свойство (**)
ICopyMeshDefinition.GetGeomArray()		

ICopyMeshDefinition.SetGeomArray(geomArray)	Установить свойство (**)

insideFlag – Признак наличия копий внутри сетки

Интерфейс...Справка системы КОМПАС...

Glava_49_Massiv_po_setke.htm#COPY_MESH_PATTERN_main

Тип данных: BOOL

Значения свойства:

TRUE	- оставлять копии во всех узлах сетки,
FALSE	- оставлять копии по периметру сетки.

Синтаксис Automation:

insideFlag = iCopyMesh.insideFlag		Получить свойство(*)
iCopyMesh.insideFlag = insideFlag		Установить свойство (*)
insideFlag	=	Получить свойство (**)
iCopyMesh.GetInsideFlag()		
iCopyMesh.SetInsideFlag(insideFlag)		Установить свойство (**)

step1 – Шаг вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step1 = iCopyMesh.step1		Получить свойство(*)
iCopyMesh.step1 = stept1		Установить свойство (*)
step1	=	Получить свойство (**)
iCopyMesh.GetStep1()		
iCopyMesh.SetStep1(step1)		Установить свойство (**)

step2 – Шаг вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step2 = iCopyMesh.step2	Получить свойство(*)

iCopyMesh.step2 = stept2		Установить свойство (*)
step2	=	Получить свойство (**)
iCopyMesh.GetStep2()		
iCopyMesh.SetStep2(step2)		Установить свойство (**)

ICopyMeshDefinition –методы

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection NULL	- в случае успешного завершения, - в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

GetAxis1– Получить первую ось операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetAxis1();

Синтаксис COM:

LPENTITY GetAxis1();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity. NULL	- в случае успешного завершения, - в случае неудачи.

Примечание:

Направление сетки можно определить, указывая ребра или оси, вдоль которых строится сетка. Если ось 1 определена, то игнорируется свойство angle1 (Угол наклона 1-ой оси).

GetAxis2 – Получить вторую ось операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetAxis2();

Синтаксис COM:

LPENTITY GetAxis2();

Возвращаемое значение:

указатель на интерфейс ksEntity или IEntity.	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Направление сетки можно определить, указывая ребра или оси, вдоль которых строится сетка. Если ось 2 определена, то игнорируется свойство angle2 (Угол наклона 2-ой оси).

GetCopyParamAlongAxis – Получить параметры копирования вдоль оси

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetCopyParamAlongAxis (BOOL firstAxis,
double * angle,
long * count,
double * step,
BOOL * factor);

Синтаксис COM:

BOOL GetCopyParamAlongAxis (BOOL firstAxis,
double * angle,
long * count,
double * step,
BOOL * factor);

Выходные параметры:

firstAxis	- ось: TRUE - первая, FALSE - вторая,
angle	- угол наклона оси,
count	- количество копий,
step	- шаг вдоль оси,
factor	- признак полного шага.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

OperationArray – Получить указатель на массив элементов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

IDispatch * OperationArray();

Синтаксис COM:

LPENTITYCOLLECTION OperationArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива элементов, подлежащих копированию ksEntityCollection или IEntityCollection.

SetAxis1 – Задать первую ось операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis1 (LPDISPATCH entity);

Синтаксис COM:

BOOL SetAxis1 (LPENTITY);

Входные параметры:

entity	- указатель на интерфейс ksEntity или IEntity (ось или прямолинейное ребро).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Направление сетки можно определить, указывая ребра или оси, вдоль которых строится сетка. Если ось 1 определена, то игнорируется свойство angle1 (Угол наклона 1-ой оси).

SetAxis2– Задать вторую ось операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis2 (LPDISPATCH entity);

Синтаксис COM:

BOOL SetAxis2 (LPENTITY);

Входные параметры:

entity - указатель на интерфейс ksEntity или IEntity (ось или прямолинейное ребро).

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Направление сетки можно определить, указывая ребра или оси, вдоль которых строится сетка. Если ось 1 определена, то игнорируется свойство angle2 (Угол наклона 2-ой оси).

SetCopyParamAlongAxis – Установить параметры копирования вдоль оси

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetCopyParamAlongAxis (BOOL firstAxis,

double angle,

long count,

double step,

BOOL factor);

Синтаксис COM:

BOOL SetCopyParamAlongAxis (BOOL firstAxis,

double angle,

long count,

double step,

BOOL factor);

Входные параметры:

firstAxis	- ось: TRUE - первая, FALSE - вторая,
angle	- угол наклона оси,
count	- количество копий,
step	- шаг вдоль оси,
factor	- признак полного шага.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Копия элементов детали по кривой (Интерфейсы ksCurveCopyDefinition и ICopyCurveDefinition)

Справка системы КОМПАС...

Интерфейс операции копирования по кривой.

ksCurveCopyDefinition	- интерфейс Automation
ICopyCurveDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksCurveCopyDefinition – свойства

count – Количество копий

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count = iCopyCurve.count		Получить свойство(*)
iCopyCurve.count = count		Установить свойство (*)
count	=	Получить свойство (**)
iCopyCurve.GetCount()		
iCopyCurve.SetCount(count)		Установить свойство (**)

factor – Признак полного шага

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- полный шаг,
FALSE	- шаг между соседними копиями.

Синтаксис Automation:

factor = iCopyCurve.factor		Получить свойство(*)
iCopyCurve.factor = factor		Установить свойство (*)
factor	=	Получить свойство (**)
iCopyCurve.GetFactor()		
iCopyCurve.SetFactor(factor)		Установить свойство (**)

fullCurve – Признак распределения копий

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- по всей длине траектории,
FALSE	- по шагу и количеству.

Синтаксис Automation:

fullCurve = iCopyCurve.fullCurve		Получить свойство(*)
iCopyCurve.fullCurve = fullCurve		Установить свойство (*)
fullCurve	=	Получить свойство (**)
iCopyCurve.GetFullCurve()		
iCopyCurve.SetFullCurve(fullCurve)		Установить свойство (**)

geomArray – Признак использования геометрического копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- использовать геометрическое копирование. Содержимое операций источников (грани, ребра) копируется без пересчета. В простых случаях операция ускоряется, но возможны ошибки при копировании скруглений и фасок на криволинейных поверхностях;
FALSE	- при копировании все операции-источники строятся заново, т.е. производится полноценный расчет.

Синтаксис Automation:

geomArray = iCopyCurveDefinition.geomArray	=	Получить свойство(*)
iCopyCurveDefinition.geomArray = geomArray	=	Установить свойство (*)
geomArray = iCopyCurveDefinition.GetGeomArray()	=	Получить свойство (**)
iCopyCurveDefinition.SetGeomArray(geomArray)		Установить свойство (**)

keepAngle – Признак сохранения угла элементов относительно траектории в начальной точке

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- сохранять угол,
FALSE	- не сохранять угол.

Синтаксис Automation:

keepAngle = iCopyCurve.keepAngle	Получить свойство(*)

iCopyCurve.keepAngle = keepAngle	Установить свойство (*)
keepAngle = iCopyCurve.GetKeepAngle()	Получить свойство (**)
iCopyCurve.SetKeepAngle(keepAngle)	Установить свойство (**)

sence – Направление копирования

Интерфейс...Справка системы КОМПАС...

`make_copy.htm#copy_of_curve`

Тип данных: BOOL

Значения свойства:

TRUE	- по часовой стрелке,
FALSE	- против часовой стрелки.

Синтаксис Automation:

sence = iCopyCurve.sence		Получить свойство(*)
iCopyCurve.sence = sence		Установить свойство (*)
sence	=	Получить свойство (**)
iCopyCurve.GetSence()		
iCopyCurve.SetSence(sence)		Установить свойство (**)

step – Шаг копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step = iCopyCurve.step		Получить свойство(*)
iCopyCurve.step = step		Установить свойство (*)
step	=	Получить свойство (**)
iCopyCurve.GetStep()		
iCopyCurve.SetStep(step)		Установить свойство (**)

ksCurveCopyDefinition – методы

OperationArray – Получить указатель на массив элементов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

IDispatch * OperationArray();

Синтаксис COM:

LPENTITYCOLLECTION OperationArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива операций, подлежащих копированию ksEntityCollection или IEntityCollection.

CurveArray – Получить указатель на массив кривых, образующих траекторию копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

IDispatch * CurveArray();

Синтаксис COM:

LPENTITYCOLLECTION CurveArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива кривых ksEntityCollection или IEntityCollection.

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:::/Glava_50_Massiv_po_koncentr_setke.htm#CIRCULAR_PATTERN_main

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

Копия элементов детали по концентрической сетке (Интерфейсы ksCircularCopyDefinition, ICopyCircularDefinition)

Справка системы КОМПАС...

Интерфейс операции копирования по окружности.

ksCircularCopyDefinition	- интерфейс Automation
ICopyCircularDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICopyCircularDefinition – свойства

count1 – Количество копий в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count1 = iCopyCirc.count1	Получить свойство(*)
iCopyCirc.count1 = count1	Установить свойство (*)
count1 = iCopyCirc.GetCount1()	Получить свойство (**)
iCopyCirc.SetCount1(count1)	Установить свойство (**)

count2 – Количество копий в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count2 = iCopyCirc.count2	Получить свойство(*)
iCopyCirc.count2 = count2	Установить свойство (*)
count2 = iCopyCirc.GetCount2()	Получить свойство (**)
iCopyCirc.SetCount2(count2)	Установить свойство (**)

factor1 – Признак полного шага в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step1 устанавливает расстояние между крайними элементами массива в радиальном направлении,
FALSE	- свойство step1 устанавливает расстояние между соседними элементами массива в радиальном направлении.

Синтаксис Automation:

factor1 = iCopyCirc.factor1		Получить свойство(*)
iCopyCirc.factor1 = factor1		Установить свойство (*)
factor1	=	Получить свойство (**)
iCopyCirc.GetFactor1()		
iCopyCirc.SetFactor1(factor1)		Установить свойство (**)

factor2 – Признак полного шага в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step2 устанавливает расстояние между крайними элементами массива в кольцевом направлении,

FALSE - свойство step2 устанавливает расстояние между соседними элементами массива в кольцевом направлении.

Синтаксис Automation:

```
factor2 = iCopyCirc.factor2                Получить свойство(* )
iCopyCirc.factor2 = factor2                Установить свойство (* )
factor2 = iCopyCirc.GetFactor2()           Получить свойство (**)
iCopyCirc.SetFactor2(factor2)              Установить свойство (**)
```

geomArray – Признак использования геометрического копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - использовать геометрическое копирование. Содержимое операций источников (грани, ребра) копируется без пересчета. В простых случаях операция ускоряется, но возможны ошибки при копировании скруглений и фасок на криволинейных поверхностях,

FALSE - при копировании все операции-источники строятся заново, т.е. производится полноценный расчет.

Синтаксис Automation:

```
geomArray = iCircularCopyDefinition.geomArray      Получить свойство(* )
iCircularCopyDefinition.geomArray = geomArray      Установить свойство (* )
iCircularCopyDefinition.GetGeomArray()             Получить свойство (**)
iCircularCopyDefinition.SetGeomArray(geomArray)    Установить свойство (**)
```

inverce – Признак направления копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - обратное направление,
FALSE - прямое направление.

Синтаксис Automation:

inverce = iCopyCirc.inverce	Получить свойство(*)
iCopyCirc.inverce = inverce	Установить свойство (*)
inverce = iCopyCirc.GetInverce()	Получить свойство (**)
iCopyCirc.SetInverce(inverce)	Установить свойство (**)

step1 – Шаг копирования в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step1 = iCopyCirc.step1	Получить свойство(*)
iCopyCirc.step1 = stept1	Установить свойство (*)
step1 = iCopyCirc.GetStep1()	Получить свойство (**)
iCopyCirc.SetStep1(step1)	Установить свойство (**)

step2 – Шаг в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

step2 = iCopyCirc.step2	Получить свойство(*)
iCopyCirc.step2 = stept2	Установить свойство (*)
step2 = iCopyCirc.GetStep2()	Получить свойство (**)
iCopyCirc.SetStep2(step2)	Установить свойство (**)

ICopyCircularDefinition – методы

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

GetAxis – Получить ось копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetAxis();

Синтаксис COM:

LPENTITY GetAxis();

Возвращаемое значение:

- указатель на интерфейс оси ksEntity или IEntity.

GetCopyParamAlongDir – Получить параметры копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetCopyParamAlongDir (long* count,

double* step,

BOOL* factor,

BOOL dir);

Синтаксис COM:

BOOL GetCopyParamAlongDir (long* count,

double* step,

BOOL* factor,

BOOL dir)

Выходные параметры:

count - количество копий,
step - шаг,
factor - признак полного шага,
dir - направление копирования.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetOperationArray – Получить указатель на массив элементов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetOperationArray();

Синтаксис COM:

LPENTITYCOLLECTION GetOperationArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива элементов, подлежащих копированию ksEntityCollection или IEntityCollection.

SetAxis – Установить ось копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis(LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis(LPENTITY axis);

Входной параметр:

axis - указатель на интерфейс оси ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetCopyParamAlongDir – Установить параметры копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetCopyParamAlongDir (long count,

double step,

BOOL factor,

BOOL dir);

Синтаксис COM:

BOOL SetCopyParamAlongDir (long count,

double step,

BOOL factor,

BOOL dir)

Входные параметры:

count - количество копий,
step - шаг,
factor - признак полного шага,
dir - направление копирования.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Зеркальная копия элементов детали Интерфейсы ksMirrorCopyDefinition, IMirrorDefinition)

Справка системы КОМПАС...

Интерфейс зеркальной копии.

ksMirrorCopyDefinition - интерфейс Automation
IMirrorDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IMirrorDefinition – методы

GetPlane – Получить плоскость симметрии (зеркального копирования)

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

OperationArray – Получить указатель на интерфейс массива копируемых элементов

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetOperationArray();

Синтаксис COM:

LPENTITYCOLLECTION OperationArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива копируемых элементов ksEntityCollection или IEntityCollection.

SetPlane – Установить плоскость симметрии (зеркального копирования)

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Зеркальная копия элементов детали (Интерфейсы ksMirrorCopyAllDefinition, IMirrorAllDefinition)

Справка системы КОМПАС...

Интерфейс зеркального отражения всех элементов.

ksMirrorCopyAllDefinition	- интерфейс Automation
IMirrorAllDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IMirrorAllDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

GetPlane – Получить плоскость отражения

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоскости отражения ksEntity или IEntity.

SetPlane – Установить плоскость отражения

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоскости отражения ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Интерфейсы копирования компонентов сборки

Копия компонентов сборки по параллельной сетке (Интерфейсы ksMeshPartArrayDefinition, IMeshPartArrayDefinition)

Справка системы КОМПАС...

Интерфейс операции копирования компонентов сборки по параллелограммной сетке.

ksMeshPartArrayDefinition	- интерфейс Automation
IMeshPartArrayDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksMeshPartArrayDefinition – свойства

angle1– Угол наклона первой оси в текущей системе координат

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

angle1 = iMeshArray.angle1		Получить свойство(*)
iMeshArray.angle1 = angle1		Установить свойство (*)
angle1	=	Получить свойство (**)
iMeshArray.GetAngle1()		
iMeshArray.SetAngle1(angle1)		Установить свойство (**)

angle2 – Угол наклона второй оси сетки относительно первой

Интерфейс...Справка системы КОМПАС...

`49_3_1_Napravlenie_pervoy_osi.htm`

Тип данных: double

Синтаксис Automation:

angle2 = iMeshArray.angle2		Получить свойство(*)
iMeshArray.angle2 = angle2		Установить свойство (*)
angle2	=	Получить свойство (**)
iMeshArray.GetAngle2()		
iMeshArray.SetAngle2(angle2)		Установить свойство (**)

count1 – Количество копий вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count1 = iMeshArray.count1		Получить свойство(*)
iMeshArray.count1 = count1		Установить свойство (*)
count1	=	Получить свойство (**)
iMeshArray.GetCount1()		
iMeshArray.SetCount1(count1)		Установить свойство (**)

count2 – Количество копий вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count2 = iMeshArray.count2		Получить свойство(*)
iMeshArray.count2 = count2		Установить свойство (*)
count2	=	Получить свойство (**)
iMeshArray.GetCount2()		
iMeshArray.SetCount2(count2)		Установить свойство (**)

factor1 – Признак полного шага копирования вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step1 устанавливает расстояние между крайними элементами вдоль 1-ой оси,
FALSE	- свойство step1 устанавливает расстояние между соседними элементами вдоль 1-ой оси.

Синтаксис Automation:

factor1 = iMeshArray.factor1		Получить свойство(*)
iMeshArray.factor1 = factor1		Установить свойство (*)
factor1	=	Получить свойство (**)
iMeshArray.GetFactor1()		
iMeshArray.SetFactor1(factor1)		Установить свойство (**)

factor2 – Признак полного шага копирования вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step2 устанавливает расстояние между крайними элементами вдоль второй оси,
FALSE	- свойство step2 устанавливает расстояние между соседними элементами вдоль второй оси.

Синтаксис Automation:

```
factor2 = iMeshArray.factor2          Получить свойство(* )
iMeshArray.factor2 = factor2          Установить свойство (* )
factor2                          =    Получить свойство (**)
iMeshArray.GetFactor2()
iMeshArray.SetFactor2(factor2)        Установить свойство (**)
```

insideFlag – Признак наличия копий внутри сетки

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- оставлять экземпляры копий во всех узлах сетки,
FALSE	- оставлять экземпляры копий только по периметру сетки.

Синтаксис Automation:

```
insideFlag = iMeshArray.insideFlag          Получить свойство(* )
iMeshArray.insideFlag = insideFlag          Установить свойство (* )
insideFlag                             =    Получить свойство (**)
iMeshArray.GetInsideFlag()
iMeshArray.SetInsideFlag(insideFlag)        Установить свойство (**)
```

step1 – Шаг копирования вдоль первой оси

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step1 = iMeshArray.step1		Получить свойство(*)
iMeshArray.step1 = stept1		Установить свойство (*)
step1	=	Получить свойство (**)
iMeshArray.GetStep1()		
iMeshArray.SetStep1(step1)		Установить свойство (**)

step2 – Шаг копирования вдоль второй оси

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step2 = iMeshArray.step2		Получить свойство(*)
iMeshArray.step2 = stept2		Установить свойство (*)
step2	=	Получить свойство (**)
iMeshArray.GetStep2()		
iMeshArray.SetStep2(step2)		Установить свойство (**)

ksMeshPartArrayDefinition – методы

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection. NULL	- в случае успешного завершения, - в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования.

В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

GetAxis1 – Получить первую ось сетки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
LPDISPATCH GetAxis1();
Синтаксис COM:
LPENTITY GetAxis1();
Возвращаемое значение:

- указатель на интерфейс оси ksEntity или IEntity.

GetAxis2 – Получить вторую ось сетки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
LPDISPATCH GetAxis2();
Синтаксис COM:
LPENTITY GetAxis2();
Возвращаемое значение:

- указатель на интерфейс оси ksEntity или IEntity.

GetCopyParamAlongAxis – Получить параметры копирования вдоль оси

Интерфейс...Справка системы КОМПАС...

`CM_COPY_MESH_PATTERN.htm`
Синтаксис Automation:
BOOL GetCopyParamAlongAxis (BOOL firstAxis,
double * angle,
long * count,
double * step,

BOOL * factor);

Синтаксис COM:

BOOL GetCopyParamAlongAxis (BOOL firstAxis,

double * angle,

long * count,

double * step,

BOOL * factor);

Выходные параметры:

firstAxis	- ось: TRUE - первая, FALSE - вторая,
angle	- угол наклона оси,
count	- количество копий,
step	- шаг вдоль оси,
factor	- признак полного шага.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

PartArray – Получить указатель на массив компонентов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива компонентов, подлежащих копированию ksPartCollection или IPartCollection.

SetAxis1 – Установить первую ось сетки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis1 (LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis1 (LPENTITY axis);

Входной параметр:

axis - указатель на интерфейс оси ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetAxis2 – Установить вторую ось сетки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis2 (LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis2 (LPENTITY axis);

Входной параметр:

axis - указатель на интерфейс оси операции ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetCopyParamAlongAxis – Установить параметры копирования вдоль оси

Интерфейс...Справка системы КОМПАС...

CM_COPY_MESH_PATTERN.htm

Синтаксис Automation:

BOOL SetCopyParamAlongAxis (BOOL firstAxis,
double angle,
long count,
double step,
BOOL factor);

Синтаксис COM:

BOOL SetCopyParamAlongAxis (BOOL firstAxis,

double angle,

long count,

double step,

BOOL factor);

Входные параметры:

firstAxis	- ось: TRUE - первая, FALSE - вторая,
angle	- угол наклона оси,
count	- количество копий,
step	- шаг вдоль оси,
factor	- признак полного шага.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Копия компонентов сборки по концентрической сетке (Интерфейсы ksCircularPartArrayDefinition, ICircularPartArrayDefinition)

Справка системы КОМПАС...

Интерфейс копирования компонентов сборки по концентрической сетке.

ksCircularPartArrayDefinition	- интерфейс Automation
ICircularPartArrayDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksCircularPartArrayDefinition – свойства

count1 – Количество копий в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count1 = iArrayCirc.count1		Получить свойство(*)
iArrayCirc.count1 = count1		Установить свойство (*)
count1	=	Получить свойство (**)
iArrayCirc.GetCount1()		
iArrayCirc.SetCount1(count1)		Установить свойство (**)

count2 – Количество копий в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count2 = iArrayCirc.count2		Получить свойство(*)
iArrayCirc.count2 = count2		Установить свойство (*)
count2	=	Получить свойство (**)
iArrayCirc.GetCount2()		
iArrayCirc.SetCount2(count2)		Установить свойство (**)

factor1 – Признак полного шага в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step1 задает расстояние между крайними в радиальном направлении копиями,
FALSE	- свойство step1 задает расстояние между соседними в радиальном направлении копиями.

Синтаксис Automation:

factor1 = iArrayCirc.factor1		Получить свойство(*)
iArrayCirc.factor1 = factor1		Установить свойство (*)
factor1	=	Получить свойство (**)
iArrayCirc.GetFactor1()		
iArrayCirc.SetFactor1(factor1)		Установить свойство (**)

factor2 – Признак полного шага в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step2 задает угол между крайними лучами сетки,
FALSE	- свойство step2 задает угол между соседними лучами сетки.

Синтаксис Automation:

factor2 = iArrayCirc.factor2		Получить свойство(*)
iArrayCirc.factor2 = factor2		Установить свойство (*)
factor2	=	Получить свойство (**)
iArrayCirc.GetFactor2()		
iArrayCirc.SetFactor2(factor2)		Установить свойство (**)

inverce – Признак направления копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- обратное направление,
FALSE	- прямое направление.

Синтаксис Automation:

inverce = iArrayCirc.inverce		Получить свойство(*)
iArrayCirc.inverce = inverce		Установить свойство (*)
inverce	=	Получить свойство (**)
iArrayCirc.GetInverce()		
iArrayCirc.SetInverce(inverce)		Установить свойство (**)

keepAngle – Признак сохранения угла копий

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- обратное направление,

FALSE - прямое направление.

Синтаксис Automation:

```
keepAngle = iCircularPartArray.keepAngle          Получить свойство(* )
iCircularPartArray.keepAngle = keepAngle          Установить свойство (* )
keepAngle = iCircularPartArray.GetKeepAngle()     Получить свойство (**)
iCircularPartArray.SetKeepAngle( keepAngle )      Установить свойство (**)
```

Примечание:

Если значение свойства равно TRUE, то необходимо сохранять угол копий.

step1 – Шаг копирования в радиальном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

```
step1 = iArrayCirc.step1          Получить свойство(* )
iArrayCirc.step1 = stept1         Установить свойство (* )
step1                       =     Получить свойство (**)
iArrayCirc.GetStep1()
iArrayCirc.SetStep1(step1)        Установить свойство (**)
```

step2 – Шаг копирования в кольцевом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

```
step2 = iArrayCirc.step2          Получить свойство(* )
iArrayCirc.step2 = stept2         Установить свойство (* )
step2                       =     Получить свойство (**)
iArrayCirc.GetStep2()
iArrayCirc.SetStep2(step2)        Установить свойство (**)
```

ksCircularPartArrayDefinition – методы

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации об удаленных копиях, а также управления составом удаленных копий. Информация об удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

GetAxis – Получить указатель на ось копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetAxis();

Синтаксис COM:

LPENTITY GetAxis();

Возвращаемое значение:

- указатель на интерфейс оси ksEntity или IEntity.

GetCopyParamAlongDir – Получить параметры копирования

Интерфейс...Справка системы КОМПАС...

CM_COPY_CIRCULAR_PATTERN.htm

Синтаксис Automation:

BOOL GetCopyParamAlongDir (long* count,

double* step,

BOOL* factor,

BOOL dir);

Синтаксис COM:

BOOL GetCopyParamAlongDir (long* count,

double* step,

BOOL* factor,

BOOL dir)

Выходные параметры:

count - количество копий,
step - шаг,
factor - признак полного шага,
dir - направление.

Возвращаемое значение:

TRUE - в случае успешного завершения.

PartArray – Указатель на массив компонентов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива операций, подлежащих копированию ksPartCollection или IPartCollection.

SetAxis – Установить указатель на ось копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetAxis (LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis (LPENTITY axis);

Входной параметр:

axis - указатель на интерфейс оси ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetCopyParamAlongDir – Установить параметры копирования

Интерфейс...Справка системы КОМПАС...

CM_COPY_CIRCULAR_PATTERN.htm

Синтаксис Automation:

BOOL SetCopyParamAlongDir (long count,

double step,

BOOL factor,

BOOL dir);

Синтаксис COM:

BOOL SetCopyParamAlongDir (long count,

double step,

BOOL factor,

BOOL dir)

Входные параметры:

count	- количество копий,
step	- шаг,
factor	- признак полного шага,
dir	- направление.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Копия компонентов сборки по кривой (Интерфейсы ksCurvePartArrayDefinition, ICurvePartArrayDefinition)

Справка системы КОМПАС...

Интерфейс операции копирования компонентов сборки по кривой.

ksCurvePartArrayDefinition	- интерфейс Automation
ICurvePartArrayDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition .

ksCurvePartArrayDefinition – свойства

count – Количество копий

Интерфейс... Справка системы КОМПАС...

Тип данных: long

Синтаксис Automation:

count = iArrayCurve.count		Получить свойство(*)
iArrayCurve.count = count		Установить свойство (*)
count	=	Получить свойство (**)
iArrayCurve.GetCount()		
iArrayCurve.SetCount(coun t)		Установить свойство (**)

factor – Признак полного шага

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- свойство step устанавливает расстояние между крайними элементами массива вдоль траектории,
FALSE	- свойство step устанавливает расстояние между соседними элементами массива вдоль траектории.

Синтаксис Automation:

factor = iArrayCurve.factor		Получить свойство(*)
iArrayCurve.factor = factor		Установить свойство (*)
factor	=	Получить свойство (**)
iArrayCurve.GetFactor()		
iArrayCurve.SetFactor(factor)		Установить свойство (**)

fullCurve – Признак расположения копий вдоль всей траектории

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - первый и последний элементы массива располагаются соответственно в начальной и конечной точках траектории. Свойства step и factor определяются автоматически с учетом условия count.

FALSE - элементы массива располагаются вдоль траектории в соответствии со свойствами step, factor и count.

Синтаксис Automation:

fullCurve = iArrayCurve.fullCurve		Получить свойство(*)
iArrayCurve.fullCurve = fullCurve		Установить свойство (*)
fullCurve	=	Получить свойство (**)
iArrayCurve.GetFullCurve()		
iArrayCurve.SetFullCurve(fullCurve)		Установить свойство (**)

keepAngle – Признак сохранения угла между исходным элементом и траекторией в начальной точке

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - угол сохраняется,

FALSE - угол не сохраняется.

Синтаксис Automation:

keepAngle = iArrayCurve.keepAngle	Получить свойство(*)
iArrayCurve.keepAngle = keepAngle	Установить свойство (*)
keepAngle = iArrayCurve.GetKeepAngle()	Получить свойство (**)
iArrayCurve.SetKeepAngle(keepAngle)	Установить свойство (**)

sence – Направление копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - прямое направление,

FALSE - обратное направление.

Синтаксис Automation:

sence = iArrayCurve.sence		Получить свойство(*)
iArrayCurve.sence = sence		Установить свойство (*)
sence	=	Получить свойство (**)
iArrayCurve.GetSence()		
iArrayCurve.SetSence(sence)		Установить свойство (**)

step – Шаг копирования

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

step = iArrayCurve.step		Получить свойство(*)
iArrayCurve.step = step		Установить свойство (*)
step	=	Получить свойство (**)
iArrayCurve.GetStep()		
iArrayCurve.SetStep(step)		Установить свойство (**)

ksCurvePartArrayDefinition – методы

CurveArray – Получить указатель на массив кривых, определяющих траекторию копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH CurveArray();

Синтаксис COM:

LPDISPATCH CurveArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива кривых, по которым происходит копирование ksEntityCollection или IEntityCollection.

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection NULL	- в случае успешного завершения, - в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации об удаленных копиях, а также управления составом удаленных копий. Информация об удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

PartArray – Получить указатель на массив компонентов для копирования

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

	- указатель на интерфейс динамического массива операций, подлежащих копированию ksPartCollection или IPartCollection.

Копия компонентов сборки по образцу (Интерфейсы ksDerivativePartArrayDefinition, IDerivativePartArrayDefinition)

Справка системы КОМПАС...

Glava_56_Massiv_po_obrazcu.htm

Интерфейс копирования компонентов сборки по образцу.

ksDerivativePartArrayDefinition	- интерфейс Automation
IDerivativePartArrayDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksDerivativePartArrayDefinition – методы

DeletedCollection – Получить массив индексов удаленных копий

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH DeletedCollection();

Синтаксис COM:

LPDELETEDCOPYCOLLECTION DeletedCollection();

Возвращаемое значение:

указатель на интерфейс ksDeletedCopyCollection или IDeletedCopyCollection	- в случае успешного завершения,
NULL	- в случае неудачи.

Примечание:

Интерфейс применяется для массивов компонентов и операций копирования, для получения информации о удаленных копиях, а также управления составом удаленных копий. Информация о удаленной копии выдается в виде индексов копии в узоре копирования. В общем случае это два индекса по ортогональным осям в плоскости узора. Для копий по кривой используется один индекс (первый). Значения индексов начинаются с 1.

GetDeriv – Получить указатель на интерфейс массива-образца

Интерфейс...Справка системы КОМПАС...

Glava_56_Massiv_po_obrazcu.htm

Синтаксис Automation:

LPDISPATCH GetDeriv();

Синтаксис COM:

LPENTITY GetDeriv();

Возвращаемое значение:

- указатель на интерфейс массива-образца ksEntity или IEntity.

PartArray – Получить указатель на массив компонентов для копирования

Интерфейс...Справка системы КОМПАС...

Glava_56_Massiv_po_obrazcu.htm

Синтаксис Automation:

LPDISPATCH PartArray();

Синтаксис COM:

LPPARTCOLLECTION PartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива компонентов, подлежащих копированию ksPartCollection или IPartCollection.

SetDeriv – Установить указатель на интерфейс массива-образца

Интерфейс...Справка системы КОМПАС...

Glava_56_Massiv_po_obrazcu.htm

Синтаксис Automation:

BOOL SetDeriv (LPDISPATCH deriv);

Синтаксис COM:

BOOL SetDeriv (LPENTITY deriv);

Входной параметр:

deriv - указатель на интерфейс массива-образца ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Интерфейсы формообразующих операций

Основание — элемент выдавливания (Интерфейс ksBaseExtrusionDefinition, IBaseExtrusionDefinition)

Интерфейс параметров основания – элемента выдавливания.

ksBaseExtrusionDefinition	- интерфейс Automation
IBaseExtrusionDefinition	- интерфейс COM

Примечание:

1. Данный интерфейс устарел. Рекомендуется использовать вместо него интерфейс ksBossExtrusionDefinition
2. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksBaseExtrusionDefinition – свойства

directionType – Направление выдавливания

Интерфейс...Тип данных:short.

Синтаксис Automation:

directionType = iBaseExtrusion.directionType	=	Получить свойство(*)
iBaseExtrusion.directionType= directionType		Установить свойство (*)
directionType = iBaseExtrusion.GetDirectionType()	=	Получить свойство (**)
iBaseExtrusion.SetDirectionType(directionType)		Установить свойство (**)

Значения свойства:

Типы направлений...

Примечание:

1. Нормаль, проведенная к грани, всегда направлена наружу ("из тела детали").
2. Прямое направление совпадает с нормалью, проведенной к плоскости эскиза.
3. Для вырезаемого элемента выдавливания направление противоположно нормали.

ksBaseExtrusionDefinition – методы

ExtrusionParam – Получить указатель на интерфейс параметров элемента выдавливания

Интерфейс...

Синтаксис Automation:

LPDISPATCH ExtrusionParam();

Синтаксис COM:

LPEXTRUSIONPARAM ExtrusionParam();

Возвращаемое значение:

- указатель на интерфейс ksExtrusionParam или IExtrusionParam.

GetDepthObject – Получить объект, задающий глубину выдавливания

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetDepthObject (BOOL normal);

Синтаксис COM:

LPENTITY GetDepthObject (BOOL normal);

Входные параметры:

normal	- направление выдавливания: (TRUE - совпадает с направлением нормали к плоскости эскиза, FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

- указатель на интерфейс объекта глубины выдавливания ksEntity или IEntity.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetSideParam – Получить параметры выдавливания в одном направлении

Интерфейс...**Синтаксис Automation**:

BOOL GetSideParam (BOOL forward,
short *type,
double *depth,
double *draftValue,
BOOL *draftOutward);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,
short *type,
double *depth,
double *draftValue,
BOOL *draftOutward);

Входной параметр:

forward - направление выдавливания:
TRUE - прямое направление,
FALSE - обратное направление.

Выходные параметры:

type - тип выдавливания,
depth - глубина выдавливания,
draftValue - угол уклона,
draftOutward - направление уклона:
FALSE - уклон наружу,
TRUE - уклон внутрь.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ResetDepthObject – Отказаться от установленного методом SetDepthObject объекта, задающего глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL ResetDepthObject (BOOL normal);

Синтаксис COM:

BOOL ResetDepthObject (BOOL normal);

Входные параметры:

normal	- направление выдавливания: (TRUE - совпадает с направлением нормали к плоскости эскиза, FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetDepthObject – Установить объект, задающий глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetDepthObject (BOOL normal, LPDISPATCH obj);

Синтаксис COM:

BOOL SetDepthObject (BOOL normal, LPENTITY obj);

Входные параметры:

238_gl_25_oper_vydavl.htm

normal	- направление выдавливания: (TRUE - совпадает с направлением нормали к плоскости эскиза, FALSE - противоположно направлению нормали к плоскости эскиза),
obj	- указатель на интерфейс объекта глубины выдавливания ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetSideParam – Установить параметры выдавливания в одном направлении

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Входные параметры:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.
type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetSketch – Задать указатель на интерфейс эскиза элемента

Интерфейс...**Синтаксис Automation**:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch - указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Приклеенный элемент выдавливания (Интерфейсы ksBossExtrusionDefinition, IBossExtrusionDefinition)

Интерфейс параметров приклеенного элемента выдавливания.

ksBossExtrusionDefinition	- интерфейс Automation
IBossExtrusionDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksBossExtrusionDefinition – свойства

chooseType – Тип области применения

Интерфейс...Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType iBossExtrusion.chooseType	=	Получить свойство(*)
iBossExtrusion.chooseType chooseType	=	Установить свойство (*)
chooseType iBossExtrusion.GetchooseType()	=	Получить свойство (**)
iBossExtrusion.SetchooseType(chooseType)		Установить свойство (**)

directionType – Направление выдавливания

Интерфейс...Тип данных: short

Синтаксис Automation:

directionType iBossExtrusion.directionType	=	Получить свойство(*)

iBossExtrusion.directionType= directionType		Установить свойство (*)
directionType	=	Получить свойство (**)
iBossExtrusion.GetDirectionType()		
iBossExtrusion.SetDirectionType(directionType)		Установить свойство (**)

Значения свойства:

Типы направлений...

Примечание:

1. Нормаль, проведенная к грани, всегда направлена наружу ("из тела детали").
2. Прямое направление совпадает с нормалью, проведенной к плоскости эскиза.
3. Для вырезаемого элемента выдавливания направление противоположно нормали.

ksBossExtrusionDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс области применения для компонентов сборки в сборочной операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEPARTS ChooseParts();

Возвращаемое значение:

- указатель на интерфейс области применения для компонентов сборки ksChooseParts или IChooseParts.

ExtrusionParam – Получить указатель на интерфейс параметров элемента выдавливания

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ExtrusionParam();

Синтаксис COM:

LPEXTRUSIONPARAM ExtrusionParam();

Возвращаемое значение:

- указатель на интерфейс ksExtrusionParam или IExtrusionParam.

GetDepthObject – Получить объект, задающий глубину выдавливания

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDepthObject (BOOL normal);

Синтаксис COM:

LPENTITY GetDepthObject (BOOL normal);

Входные параметры:

normal	- направление выдавливания: (TRUE - совпадает с направлением нормали к плоскости эскиза, FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetSideParam – Получить параметры выдавливания в одном направлении

Интерфейс...**Синтаксис Automation**:

BOOL GetSideParam (BOOL forward,

short *type,

double *depth,
double *draftValue,
BOOL *draftOutward);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,
short *type,
double *depth,
double *draftValue,
BOOL *draftOutward);

Входной параметр:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.

Выходные параметры:

type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Типы выдавливания...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ResetDepthObject – Отказаться от установленного методом SetDepthObject объекта, задающего глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL ResetDepthObject (BOOL normal);

Синтаксис COM:

BOOL ResetDepthObject (BOOL normal);

Входные параметры:

normal - направление выдавливания:
(TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

SetDepthObject – Установить объект, задающий глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetDepthObject (BOOL normal, LPDISPATCH obj);

Синтаксис COM:

BOOL SetDepthObject (BOOL normal, LPENTITY obj);

Входные параметры:

normal - направление выдавливания:
(TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза).

obj - указатель на интерфейс объекта ksEntity

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

SetSideParam – Изменить параметры выдавливания в одном направлении

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Входные параметры:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.
type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetSketch – Изменить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Основание — элемент вращения (Интерфейсы ksBaseRotatedDefinition, IBaseRotatedDefinition)

Справка системы КОМПАС...

252_gl_26_oper_vr.htm
Интерфейс параметров основания – элемента вращения.

ksBaseRotatedDefinition	- интерфейс Automation
IBaseRotatedDefinition	- интерфейс COM

Примечание:

1. Данный интерфейс устарел. Рекомендуется использовать вместо него интерфейс ksBossRotatedDefinition,
2. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IBaseRotatedDefinition – свойства

directionType – Направление вращения

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm
Тип данных: short

Синтаксис Automation:

directionType = iBaseRotated.directionType	=	Получить свойство(*)
iBaseRotated.directionType= directionType		Установить свойство (*)
directionType = iBaseRotated.GetDirectionType()	=	Получить свойство (**)
iBaseRotated.SetDirectionType (directionType)		Установить свойство (**)

Значения свойства:

Типы направлений...

Примечание:

1. Прямое направление - вдоль нормали к плоскости эскиза и всегда против часовой стрелки.
2. Нормаль, проведенная к поверхности тела, всегда выходит из тела.

toroidShapeType – Признак тороида

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- тороид,
FALSE	- сфероид.

Синтаксис Automation:

toroidShapeType iBaseRotated.toroidShapeType	=	Получить свойство(*)
iBaseRotated.toroidShapeType toroidShapeType	=	Установить свойство (*)
toroidShapeType iBaseRotated.GetToroidShapeType()	=	Получить свойство (**)
iBaseRotated.SetToroidShapeType (toroidShapeType)		Установить свойство (**)

IBaseRotatedDefinition – методы

GetSideParam – Получить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

BOOL GetSideParam (BOOL forward,

double *angle);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,

double * angle);

Входной параметр:

forward	- направление вращения: TRUE - прямое, FALSE - обратное.

Выходной параметр:

angle	- угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Выходные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

RotatedParam – Получить указатель на интерфейс параметров элемента вращения

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

LPDISPATCH RotatedParam();

Синтаксис COM:

LPROTATEDPARAM RotatedParam();

Возвращаемое значение:

- указатель на интерфейс ksRotatedParam или IRotatedParam.

SetSideParam – Изменить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

BOOL SetSideParam (short forward, double angle);

Синтаксис COM:

BOOL SetSideParam (BOOL forward, double angle);

Входные параметры:

forward - направление вращения:
TRUE - прямое,
FALSE - обратное.

angle - угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetSketch – Изменить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch - указатель на интерфейс эскиза ksEntity или IEntity

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,
double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,
double reverseThickness);

Входные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParamили IThinParam.

Приклееный элемент вращения (Интерфейс ksBossRotatedDefinition, IBossRotatedDefinition)

Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Интерфейс приклеенного элемента вращения.

ksBossRotatedDefinition	- интерфейс Automation
IBossRotatedDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IBossRotatedDefinition – свойства

chooseType – Тип области применения

Интерфейс...Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType = iBossRotated.chooseType	Получить свойство(*)
iBossRotated.chooseType= chooseType	Установить свойство (*)

chooseType iBossRotated.GetchooseType()	=	Получить свойство (**)
iBossRotated.SetchooseType (chooseType)		Установить свойство (**)

directionType – Направление вращения

Интерфейс...Справка системы КОМПАС...

Тип данных: short

Значения свойства:

Типы направлений...

Синтаксис Automation:

directionType iBossRotated.directionType	=	Получить свойство(*)
iBossRotated.directionType= directionType		Установить свойство (*)
directionType iBossRotated.GetDirectionType()	=	Получить свойство (**)
iBossRotated.SetDirectionType (directionType)		Установить свойство (**)

Примечание:

Прямое направление - вдоль нормали к плоскости эскиза и всегда против часовой стрелки.

Нормаль, проведенная к поверхности тела, всегда выходит из тела.

toroidShapeType – Признак тороида

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- тороид,
FALSE	- сфероид.

Синтаксис Automation:

toroidShapeType iBossRotated.toroidShapeType	=	Получить свойство(*)

iBossRotated.toroidShapeType = toroidShapeType	=	Установить свойство (*)
toroidShapeType = iBossRotated.GetToroidShapeType()	=	Получить свойство (**)
iBossRotated.SetToroidShapeType(toroidShapeType)		Установить свойство (**)

IBossRotatedDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс области применения для компонентов сборки в сборочной операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEPARTS ChooseParts();

Возвращаемое значение:

- указатель на интерфейс области применения для компонентов сборки ksChooseParts или IChooseParts.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetSideParam – Получить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

BOOL GetSideParam (BOOL forward,

double *angle);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,

double * angle);

Входной параметр:

forward - направление вращения:
TRUE - прямое,
FALSE - обратное.

Выходной параметр:

angle - угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetThinParam – Получить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

RotatedParam – Получить указатель на интерфейс параметров элемента вращения

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

LPDISPATCH RotatedParam();

Синтаксис COM:

LPROTATEDPARAM RotatedParam();

Возвращаемое значение:

- указатель на интерфейс ksRotatedParam или IRotatedParam.

SetSideParam – Изменить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

252_gl_26_oper_vr.htm

Синтаксис Automation:

BOOL SetSideParam (short forward,

double angle);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

double angle);

Входные параметры:

forward - направление вращения:
TRUE - прямое,
FALSE - обратное,
angle - угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetSketch – Изменить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
BOOL SetSketch (LPDISPATCH sketch);
Синтаксис COM:
BOOL SetSketch (LPENTITY sketch);
Входной параметр:

sketch - указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,
double reverseThickness);
Синтаксис COM:
BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParamn.

Вырезанный кинематический элемент (Интерфейс ksCutEvolutionDefinition, ICutEvolutionDefinition)

Справка системы КОМПАС...

261_gl_27_oper_kin.htm

Интерфейс вырезанного кинематического элемента.

ksCutEvolutionDefinition	- интерфейс Automation
ICutEvolutionDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutEvolutionDefinition – свойства

chooseType – Тип области применения

Интерфейс...Справка системы КОМПАС...

Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType =iEvolution.chooseType		Получить свойство(*)
iEvolution.chooseType = chooseType		Установить свойство (*)
chooseType	=	Получить свойство (**)
iEvolution.GetchooseType()		
iEvolution.SetchooseType(chooseType)		Установить свойство (**)

cut – Признак результата операции

Интерфейс...Тип данных: BOOL

Значения свойства:

TRUE	- результат операции - вычитание элемента,
FALSE	- результат операции - пересечение элементов.

Синтаксис Automation:

cut = iEvolution.cut	Получить свойство(*)
iEvolution.cut = cut	Установить свойство (*)
cut = iEvolution.GetCut()	Получить свойство (**)
iEvolution.SetCut(cut)	Установить свойство (**)

sketchShiftType – Тип движения сечения по траектории

Интерфейс...Тип данных: short

Значения свойства:

0	- образующая переносится параллельно самой себе,
1	- образующая при переносе сохраняет исходный угол с направляющей,
2	- плоскость образующей выставляется и сохраняется ортогональной направляющей.

Синтаксис Automation:

sketchShiftType = iEvolution.sketchShiftType	Получить свойство(*)
iEvolution.sketchShiftType = sketchShiftType	Установить свойство (*)

sketchShiftType = iEvolution.GetSketchShiftType()	Получить свойство (**)
iEvolution.SetSketchShiftType(sketchShiftType)	Установить свойство (**)

ICutEvolutionDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...Справка системы КОМПАС...

906_107_2_Oblastq_primenenija_o.htm

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

GetPathLength – Получить длину результирующей кривой траектории

Интерфейс...**Синтаксис Automation**:

double GetPathLength (unsigned long bitVector);

Синтаксис COM:

double GetPathLength(unsigned long bitVector);

Входные параметры:

bitVector	- единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

длина результирующей кривой траектории	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

Метод работает только на уже построенной операции.

GetSketch – Получить указатель на интерфейс эскиза сечения кинематического элемента

Интерфейс...Справка системы КОМПАС...

`261_gl_27_oper_kin.htm`

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

	- указатель на интерфейс эскиза ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,
BYTE *thinType,
double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThick ness	- толщина стенки в прямом направлении,
reverseThic kness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

PathPartArray – Получить указатель на интерфейс массива кривых, задающих траекторию движения сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

IDispatch * PathPartArray();

Синтаксис COM:

LPENTITYCOLLECTION PathPartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива кривых ksEntityCollection или IEntityCollection.

SetSketch – Установить эскиз сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Установить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

BYTE thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

ThinParam – Получить указатель на интерфейс тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPROPERTY ThinParam();

Возвращаемое значение:

- указатель на интерфейс тонкой стенки ksThinParam или IThinParam.

Приклеенный кинематический элемент (Интерфейсы ksBossEvolutionDefinition, IBossEvolutionDefinition)

Интерфейс приклеенного кинематического элемента.

ksBossEvolutionDefinition	- интерфейс Automation
IBossEvolutionDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IBossEvolutionDefinition – свойства

chooseType – Тип области применения

Интерфейс...Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType	=	Получить свойство(*)
iBossEvolution.chooseType		
iBossEvolution.chooseType= chooseType		Установить свойство (*)
chooseType	=	Получить свойство (**)
iBossEvolution.GetchooseType()		
iBossEvolution.SetchooseType (chooseType)		Установить свойство (**)

sketchShiftType – Тип движения сечения по траектории

Интерфейс...Тип данных: short

Значения свойства:

0	- образующая переносится параллельно самой себе,
1	- образующая при переносе сохраняет исходный угол с направляющей,
2	- плоскость образующей выставляется и сохраняется ортогональной направляющей.

Синтаксис Automation:

sketchShiftType = iEvolution.sketchShiftType	Получить свойство(*)
iEvolution.sketchShiftType = sketchShiftType	Установить свойство (*)
sketchShiftType = iEvolution.GetSketchShiftType()	Получить свойство (**)
iEvolution.SetSketchShiftType(sketchShiftType)	Установить свойство (**)

IBossEvolutionDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс области применения для компонентов сборки в сборочной операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEPARTS ChooseParts();

Возвращаемое значение:

- указатель на интерфейс области применения для компонентов сборки ksChooseParts или IChooseParts.

GetPathLength – Получить длину результирующей кривой траектории

Интерфейс...**Синтаксис Automation**:

double GetPathLength (unsigned long bitVector);

Синтаксис COM:

double GetPathLength(unsigned long bitVector);

Входные параметры:

bitVector - единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

длина результирующей кривой траектории	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

Метод работает только на уже построенной операции.

GetSketch – Получить указатель на интерфейс эскиза сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

	- указатель на интерфейс эскиза элемента ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

BYTE *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

PathPartArray – Получить указатель на интерфейс массива кривых, задающих траекторию движения сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

IDispatch * PathPartArray();

Синтаксис COM:

LPENTITYCOLLECTION PathPartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива кривых ksEntityCollection или IEntityCollection.

SetSketch – Установить эскиз сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch - указатель на интерфейс эскиза элемента ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetThinParam – Установить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,
double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

BYTE thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений..

Возвращаемое значение:

TRUE	- в случае успешного завершения.

ThinParam – Получить указатель на интерфейс тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPROPERTY ThinParam();

Возвращаемое значение:

- указатель на интерфейс тонкой стенки ksThinParam или IThinParam.

Вырезанный элемент по сечениям (Интерфейсы ksCutLoftDefinition, ICutLoftDefinition)

Справка системы КОМПАС...

CM_BASE_LOFT_SOLID.htm

Интерфейс параметров вырезанного элемента по сечениям.

ksCutLoftDefinition	- интерфейс Automation
ICutLoftDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutLoftDefinition – свойства

chooseType – Тип области применения

Интерфейс...Справка системы КОМПАС...

Тип данных: из перечисления...

Синтаксис Automation:

chooseType = iCutLoft.chooseType	Получить свойство(*)
iCutLoft.chooseType = chooseType	Установить свойство (*)
chooseType = iCutLoft.GetchooseType()	Получить свойство (**)
iCutLoft.SetchooseType(chooseType)	Установить свойство (**)

cut – Признак результата операции

Интерфейс...Справка системы КОМПАС...

249_25_8_Rez_oper.htm

Тип данных: BOOL

Значения свойства:

TRUE	- результат операции - вычитание элемента,
FALSE	- результат операции - пересечение элементов.

Синтаксис Automation:

cut = iCutLoft.cut	Получить свойство(*)
iCutLoft.cut = cut	Установить свойство (*)
cut = iCutLoft.GetCut()	Получить свойство (**)
iCutLoft.SetCut (cut)	Установить свойство (**)

ICutLoftDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...906_107_2_Oblastq_primenenija_o.htm

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

GetDirectionalLine – Получить направляющую линию. Эскиз в котором лежит кривая

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetDirectionalLine();

Синтаксис COM:

LPENTITY GetDirectionalLine();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

GetLoftParam – Получить параметры вырезанного элемента по сечениям

Интерфейс...**Синтаксис Automation**:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Синтаксис COM:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Выходные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE	- в случае успешного завершения.

SetDirectionalLine – Установить направляющую линию. Эскиз в котором лежит кривая

Интерфейс...**Синтаксис Automation**:

BOOL SetDirectionalLine (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetDirectionalLine (LPENTITY sketch);

Входные параметры:

sketch	- указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

SetLoftParam – Изменить параметры вырезанного элемента по сечениям

Интерфейс...**Синтаксис Automation**:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Синтаксис COM:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Входные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,

normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE - в случае успешного завершения.

Sketches – Получить указатель на интерфейс массива эскизов элемента по сечениям

Интерфейс...Справка системы КОМПАС...

CM_BASE_LOFT_SOLID.htm

Синтаксис Automation:

LPDISPATCH Sketches();

Синтаксис COM:

LPENTITYCOLLECTION Sketches();

Возвращаемое значение:

- указатель на интерфейс массива эскизов ksEntityCollection или IEntityCollection.

Примечания:

1. В массиве включен контроль, не позволяющий добавить в него нулевой указатель на эскиз.
2. Эскизы из данного массива используются для построения элемента по сечениям.

Приклеенный элемент по сечениям (Интерфейсы ksBossLoftDefinition, IBossLoftDefinition)

Справка системы КОМПАС...

CM_BASE_LOFT_SOLID.htm

Интерфейс приклеенного элемента по сечениям.

ksBossLoftDefinition - интерфейс Automation
IBossLoftDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

BossLoftDefinition – свойства

chooseType – Тип области применения

Интерфейс...Тип данных: из перечисления ksChooseType.

Синтаксис Automation:

chooseType = BossLoft.chooseType	Получить свойство(*)
BossLoft.chooseType = chooseType	Установить свойство (*)
chooseType = BossLoft.GetchooseType()	Получить свойство (**)
BossLoft.SetchooseType(chooseType)	Установить свойство (**)

IBossLoftDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...Справка системы КОМПАС...

`906_107_2_Oblastq_primenenija_o.htm`

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс области применения для компонентов сборки в сборочной операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEPARTS ChooseParts();

Возвращаемое значение:

- указатель на интерфейс области применения для компонентов сборки ksChooseParts или IChooseParts.

GetDirectionalLine – Получить направляющую линию. Эскиз, в котором лежит кривая

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetDirectionalLine();

Синтаксис COM:

LPENTITY GetDirectionalLine();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

GetLoftParam – Получить параметры приклеенного элемента по сечениям

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Синтаксис COM:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Выходные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,

autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetDirectionalLine – Установить направляющую линию. Эскиз в котором лежит кривая

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetDirectionalLine (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetDirectionalLine (LPENTITY sketch);

Входные параметры:

sketch - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Если задана направляющая, то признак замкнутости траектории устанавливается автоматически, исходя из замкнутости направляющей.

SetLoftParam – Изменить параметры приклеенного элемента по сечениям

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Синтаксис COM:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Входные параметры:

closed - признак замкнутости траектории:
TRUE - траектория замкнута,
FALSE - траектория разомкнута,

flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Sketches – Получить указатель на интерфейс массива эскизов элемента по сечениям

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH Sketches();

Синтаксис COM:

LPENTITYCOLLECTION Sketches();

Возвращаемое значение:

- указатель на интерфейс массива эскизов ksEntityCollection или IEntityCollection.

Примечание:

В массиве включен контроль, не позволяющий добавить в него нулевой указатель на эскиз.

Эскизы из данного массива используются для построения элемента по сечениям.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Основание — элемент по сечениям (Интерфейсы ksBaseLoftDefinition, IBaseLoftDefinition)

Справка системы КОМПАС...

Интерфейс параметров основания – элемента по сечениям.

ksBaseLoftDefinition	- интерфейс Automation
IBaseLoftDefinition	- интерфейс COM

Примечания:

1. Данный интерфейс устарел. Рекомендуется использовать вместо него интерфейс ksBossLoftDefinition.
2. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IBaseLoftDefinition – методы

GetLoftParam – Получить параметры элемента по сечениям

Интерфейс...**Синтаксис Automation**:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Синтаксис COM:

BOOL GetLoftParam (BOOL *closed,

BOOL *flipVertex,

BOOL *autoPath);

Выходные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,
flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetLoftParam – Изменить параметры операции по сечениям

Интерфейс...**Синтаксис Automation**:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Синтаксис COM:

BOOL SetLoftParam (BOOL closed,

BOOL flipVertex,

BOOL autoPath);

Входные параметры:

closed	- признак замкнутости траектории: TRUE - траектория замкнута, FALSE - траектория разомкнута,

flipVertex	- параметр зарезервирован для дальнейшего использования,
autoPath	- признак автоматического формирования траектории: TRUE - формировать траекторию автоматически, FALSE - для формирования траектории использовать точки, заданные методом SetLoftPoint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Sketches – Получить указатель на интерфейс массива эскизов элемента по сечениям

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Sketches();

Синтаксис COM:

LPENTITYCOLLECTION Sketches();

Возвращаемое значение:

- указатель на интерфейс массива эскизов ksEntityCollection или IEntityCollection.

Примечания:

1. В массиве включен контроль, не позволяющий добавить в него нулевой указатель на эскиз.
2. Эскизы из данного массива используются для построения элемента по сечениям.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Элемент выдавливания (Интерфейсы ksExtrusionParam, IExtrusionParam)

Справка системы КОМПАС...

Интерфейс параметров элемента выдавливания.

ksExtrusionParam - интерфейс Automation

IExtrusionParam - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IExtrusionParam – свойства

depthNormal – Глубина выдавливания в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

depthNormal iExtrusionParam.depthNormal	=	Получить свойство(*)
iExtrusionParam.depthNormal depthNormal	=	Установить свойство (*)
depthNormal iExtrusionParam.GetDepthNormal()	=	Получить свойство (**)
iExtrusionParam.SetDepthNormal (depthNormal)		Установить свойство (**)

depthReverse – Глубина выдавливания в обратном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

depthReverse iExtrusionParam.depthReverse	=	Получить свойство(*)
iExtrusionParam.depthReverse depthReverse	=	Установить свойство (*)
depthReverse iExtrusionParam.GetDepthReverse()	=	Получить свойство (**)
iExtrusionParam.SetDepthReverse (depthReverse)		Установить свойство (**)

direction – Направление выдавливания

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Прямое направление совпадает с нормалью, проведенной к плоскости эскиза.

Для вырезаемого элемента выдавливания направление противоположно нормали.

Значения свойства:

Типы направлений...

Синтаксис Automation:

direction = iExtrusionParam.direction		Получить свойство(*)
iExtrusionParam.direction = direction		Установить свойство (*)
direction	=	Получить свойство (**)
iExtrusionParam.GetDirection()		
iExtrusionParam.SetDirection(direction)		Установить свойство (**)

Примечание:

Нормаль, проведенная к грани, всегда направлена наружу ("из тела детали").

draftOutwardNormal – Признак уклона внутрь в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

FALSE	- уклон наружу,
TRUE	- уклон внутрь.

Синтаксис Automation:

draftOutwardNormal	=	Получить свойство(*)
iExtrusionParam.draftOutwardNormal		
iExtrusionParam.draftOutwardNormal	=	Установить свойство (*)
draftOutwardNormal		
draftOutwardNormal	=	Получить свойство (**)
iExtrusionParam.GetDraftOutwardNormal		
()		
iExtrusionParam.SetDraftOutwardNormal		Установить свойство (**)
(draftOutwardNormal)		

draftOutwardReverse – Признак уклона внутрь в обратном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

FALSE - уклон наружу,
TRUE - уклон внутрь.

Синтаксис Automation:

draftOutwardReverse = iExtrusionParam.draftOutwardReverse	=	Получить свойство(*)
iExtrusionParam.draftOutwardReverse = draftOutwardReverse	=	Установить свойство (*)
draftOutwardReverse = iExtrusionParam.GetDraftOutwardReverse( )	=	Получить свойство (**)
iExtrusionParam.SetDraftOutwardReverse (draftOutwardReverse)		Установить свойство (**)

draftValueNormal – Угол уклона в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

draftValueNormal = iExtrusionParam.draftValueNormal	=	Получить свойство(*)
iExtrusionParam.draftValueNormal = draftValueNormal	=	Установить свойство (*)
draftValueNormal = iExtrusionParam.GetDraftValueNormal()	=	Получить свойство (**)
iExtrusionParam.SetDraftValueNormal (draftValueNormal)		Установить свойство (**)

draftValueReverse – Угол уклона в обратном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

draftValueReverse iExtrusionParam.draftValueReverse	=	Получить свойство(*)
iExtrusionParam.draftValueReverse draftValueReverse	=	Установить свойство (*)
draftValueReverse iExtrusionParam.GetDraftValueReverse()	=	Получить свойство (**)
iExtrusionParam.SetDraftValueReverse (draftValueReverse)		Установить свойство (**)

typeNormal – Тип выдавливания в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: short

Значения свойства:

Типы выдавливания...

Синтаксис Automation:

typeNormal iExtrusionParam.typeNormal	=	Получить свойство(*)
iExtrusionParam.typeNormal typeNormal	=	Установить свойство (*)
typeNormal iExtrusionParam.GetTypeNormal()	=	Получить свойство (**)
iExtrusionParam.SetTypeNormal (typeNormal)		Установить свойство (**)

typeReverse – Тип выдавливания в обратном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: short

Значения свойства:

Типы выдавливания...

Синтаксис Automation:

typeReverse	=	Получить свойство(*)
iExtrusionParam.typeReverse		
iExtrusionParam.typeReverse	=	Установить свойство (*)
typeReverse		
typeReverse	=	Получить свойство (**)
iExtrusionParam.GetTypeReverse()		
iExtrusionParam.SetTypeReverse (typeReverse)		Установить свойство (**)

Вырезанный элемент выдавливания (Интерфейсы ksCutExtrusionDefinition, ICutExtrusionDefinition)

Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Интерфейс параметров вырезанного элемента выдавливания.

ksCutExtrusionDefinition	- интерфейс Automation
ICutExtrusionDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutExtrusionDefinition – свойства

chooseType – Тип области применения

Интерфейс...Справка системы КОМПАС...

Тип данных: из перечисления...

Синтаксис Automation:

chooseType = iCutExtrusion.chooseType		Получить свойство(*)
iCutExtrusion.chooseType = chooseType		Установить свойство (*)
chooseType	=	Получить свойство (**)
iCutExtrusion.GetchooseType()		
iCutExtrusion.SetchooseType(chooseType)		Установить свойство (**)

cut – Признак результата операции

Интерфейс...Справка системы КОМПАС...

249_25_8_Rez_oper.htm

Тип данных: BOOL

Значения свойства:

TRUE	- результат операции - вычитание элемента,
FALSE	- результат операции - пересечение элементов.

Синтаксис Automation:

cut = iCutExtrusion.cut	Получить свойство(*)
iCutExtrusion.cut = cut	Установить свойство (*)
cut = iCutExtrusion.GetCut()	Получить свойство (**)
iCutExtrusion.SetCut(cut)	Установить свойство (**)

directionType – Направление выдавливания

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Тип данных: short

Синтаксис Automation:

directionType = iCutExtrusion.directionType	=	Получить свойство(*)
iCutExtrusion.directionType= directionType		Установить свойство (*)
directionType = iCutExtrusion.GetDirectionType()	=	Получить свойство (**)
iCutExtrusion.SetDirectionType(directionType)		Установить свойство (**)

Значения свойства:

Типы направлений..

Примечание:

1. Нормаль, проведенная к грани, всегда направлена наружу ("из тела детали").
2. Прямое направление совпадает с нормалью, проведенной к плоскости эскиза.
3. Для вырезаемого элемента выдавливания направление противоположно нормали.

ICutExtrusionDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

ExtrusionParam – Получить указатель на интерфейс параметров элемента выдавливания

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

LPDISPATCH ExtrusionParam();

Синтаксис COM:

LPEXTRUSIONPARAM ExtrusionParam();

Возвращаемое значение:

- указатель на интерфейс ksExtrusionParam или IExtrusionParam.

GetDepthObject – Получить объект, задающий глубину выдавливания

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

LPDISPATCH GetDepthObject (BOOL normal);

Синтаксис COM:

LPENTITY GetDepthObject (BOOL normal);

Входной параметр:

normal - направление выдавливания:
(TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetSideParam – Получить параметры выдавливания в одном направлении

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

BOOL GetSideParam (BOOL forward,

short *type,

double *depth,

double *draftValue,

BOOL *draftOutward);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,
short *type,
double *depth,
double *draftValue,
BOOL *draftOutward);

Входной параметр:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.

Выходные параметры:

type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Типы выдавливания...

Возвращаемое значение:

TRUE	- в случае успешного завершения.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,
short *thinType,
double *normalThickness,
double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThic kness	- толщина стенки в прямом направлении,
reverseThic kness	- толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE - в случае успешного завершения.

ResetDepthObject – Отказаться от установленного методом SetDepthObject объекта, задающего глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL ResetDepthObject (BOOL normal);

Синтаксис COM:

BOOL ResetDepthObject (BOOL normal);

Входной параметр:

normal - направление выдавливания:
(TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза).

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetDepthObject – Установить объект, задающий глубину выдавливания

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetDepthObject (BOOL normal, LPDISPATCH obj);

Синтаксис COM:

BOOL SetDepthObject (BOOL normal, LPENTITY obj);

Входной параметр:

normal - направление выдавливания:
(TRUE - совпадает с направлением нормали к плоскости эскиза,
FALSE - противоположно направлению нормали к плоскости эскиза),
obj - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetSideParam – Изменить параметры выдавливания в одном направлении

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

short type,

double depth,

double draftValue,

BOOL draftOutward);

Входные параметры:

forward	- направление выдавливания: TRUE - прямое направление, FALSE - обратное направление.
type	- тип выдавливания,
depth	- глубина выдавливания,
draftValue	- угол уклона,
draftOutward	- направление уклона: FALSE - уклон наружу, TRUE - уклон внутрь.

Типы выдавливания...**Возвращаемое значение**:

TRUE	- в случае успешного завершения.

SetSketch – Изменить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

CM_BASE_EXTRUSION_SOLID.htm

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE	- в случае успешного завершения.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Сечение плоскостью (Интерфейсы ksCutByPlaneDefinition, ICutByPlaneDefinition)

Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Интерфейс операции сечения плоскостью.

ksCutByPlaneDefinition	- интерфейс Automation
ICutByPlaneDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutByPlaneDefinition – свойства

chooseType – Тип области применения

Интерфейс... Справка системы КОМПАС...

Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType = iCutByPlane.chooseType		Получить свойство(*)
iCutByPlane.chooseType = chooseType		Установить свойство (*)
chooseType = iCutByPlane.GetchooseType()	=	Получить свойство (**)
iCutByPlane.SetchooseType(chooseType)		Установить свойство (**)

direction – Направление отсечения

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Тип данных: BOOL

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Синтаксис Automation:

direction = iCutByPlane.direction		Получить свойство(*)
iCutByPlane.direction = direction		Установить свойство (*)
direction	=	Получить свойство (**)
iCutByPlane.GetDirection()		
iCutByPlane.SetDirection(direction)		Установить свойство (**)

ICutByPlaneDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Синтаксис Automation:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

GetPlane – Получить секущую плоскость

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс секущей плоскости ksEntity или IEntity.

SetPlane – Установить секущую плоскость

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс секущей плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Сечение по эскизу (Интерфейсы ksCutBySketchDefinition, ICutBySketchDefinition)

Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Интерфейс операции сечения эскизом.

ksCutBySketchDefinition	- интерфейс Automation
ICutBySketchDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutBySketchDefinition – свойства

direction – Направление отсечения

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Тип данных: BOOL

Значения свойства:

TRUE - прямое направление,
FALSE - обратное направление.

Синтаксис Automation:

direction = iCutBySketch.direction	Получить свойство(*)
iCutBySketch.direction = direction	Установить свойство (*)
direction = iCutBySketch.GetDirection()	Получить свойство (**)
iCutBySketch.SetDirection(direction)	Установить свойство (**)

chooseType – Тип области применения

Интерфейс... Справка системы КОМПАС...

Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType = iCutBySketch.chooseType	Получить свойство(*)
iCutBySketch.chooseType = chooseType	Установить свойство (*)
chooseType = iCutBySketch.GetchooseType()	Получить свойство (**)
iCutBySketch.SetchooseType(chooseType)	Установить свойство (**)

ICutBySketchDefinition – методы

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс... Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс... Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

GetSketch – Получить эскиз секущей поверхности

Интерфейс... Справка системы КОМПАС...

`CM_MAKE_CUT_OPER.htm`

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза секущей поверхности ksEntity или IEntity.

SetSketch – Установить эскиз секущей поверхности

Интерфейс... Справка системы КОМПАС...

CM_MAKE_CUT_OPER.htm

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

plane	- указатель на интерфейс эскиза секущей поверхности ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Тонкостенная оболочка (Интерфейсы ksShellDefinition, IShellDefinition)

Справка системы КОМПАС...

Интерфейс тонкостенной оболочки.

ksShellDefinition	- интерфейс Automation
IShellDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition .

ksShellDefinition – свойства

thickness – Толщина оболочки

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

thickness = iShell.thickness	Получить свойство(*)
iShell.thickness = thickness	Установить свойство (*)

thickness	=	Получить свойство (**)
iShell.GetThickness()		
iShell.SetThickness(thickness)		Установить свойство (**)

thinType – Направление построения оболочки

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- внутрь,
FALSE	- наружу.

Синтаксис Automation:

thinType = iShell.thinType	Получить свойство(*)
iShell.thinType = thinType	Установить свойство (*)
thinType = iEvolution.GetThinType()	Получить свойство (**)
iShell.SetThinType(thinType)	Установить свойство (**)

ksShellDefinition – методы

FaceArray – Получить указатель на интерфейс массива удаляемых граней

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH FaceArray();

Синтаксис COM:

LPENTITYCOLLECTION FaceArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива удаляемых граней ksEntityCollection или IEntityCollection.

Ребро жесткости (ksRibDefinition, IRibDefinition)

Справка системы КОМПАС...

Интерфейс ребра жесткости.

ksRibDefinition - интерфейс Automation
IRibDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksRibDefinition – свойства

angle – Угол уклона ребра жесткости

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

angle = iRib.angle	Получить свойство(*)
iRib.angle = angle	Установить свойство (*)
angle = iRib.GetAngle()	Получить свойство (**)
iRib.SetAngle(angle)	Установить свойство (**)

index – Сегмент, задающий направление уклона

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Значения свойства:

0	- первое ребро эскиза,
1, 2,… и т.д.	- последующие ребра эскиза.

Синтаксис Automation:

index = iRib.index	Получить свойство(*)
iRib.index = index	Установить свойство (*)
index = iRib.GetIndex()	Получить свойство (**)
iRib.SetIndex(index)	Установить свойство (**)

side – Направление формирования ребра

Интерфейс...Справка системы КОМПАС...

Тип данных: long

Значения свойства:

0	- прямое направление в плоскости эскиза,
1	- обратное направление в плоскости эскиза,
2	- прямое направление ортогонально эскизу,
3	- обратное направление ортогонально эскизу.

Синтаксис Automation:

side = iRib.side	Получить свойство(*)
iRib.side = side	Установить свойство (*)
side = iRib.GetSide()	Получить свойство (**)
iRib.SetAngle(angle)	Установить свойство (**)

ksRibDefinition – методы

GetSketch – Получить указатель на интерфейс эскиза ребра

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH * GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ребра ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (short *thinType,
double *normalThickness,
double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BYTE *thinType,
double *normalThickness,
double *reverseThickness);

Выходные параметры:

thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Типы направлений выдавливания..

SetSketch – Установить эскиз ребра

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс эскиза ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Установить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (short thinType,
double normalThickness,
double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BYTE thinType,
double normalThickness,
double reverseThickness);

Входные параметры:

thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Типы направлений выдавливания..

ThinParam – Получить указатель на интерфейс тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPROPERTY ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Уклон (Интерфейсы ksInclineDefinition, IInclineDefinition)

Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Интерфейс уклона.

ksInclineDefinition	- интерфейс Automation
IInclineDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition

ksInclineDefinition – свойства

angle – Угол уклона

Интерфейс...Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Тип данных: double

Синтаксис Automation:

angle = iIncline.angle	Получить свойство(*)
iIncline.angle = angle	Установить свойство (*)
angle = iIncline.GetAngle()	Получить свойство (**)
iIncline.SetAngle(angle)	Установить свойство (**)

direction – Направление уклона

Интерфейс...Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Тип данных: BOOL

Значения свойства:

TRUE	- наружу,
FALSE	- внутрь.

Синтаксис Automation:

direction = iIncline.direction		Получить свойство(*)
iIncline.direction = direction		Установить свойство (*)
direction = iIncline.GetDirection()	=	Получить свойство (**)
iIncline.SetDirection(direction)		Установить свойство (**)

ksInclineDefinition – методы

FaceArray – Получить указатель на интерфейс массива уклоняемых граней

Интерфейс...Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Синтаксис Automation:

LPDISPATCH FaceArray();

Синтаксис COM:

LPENTITYCOLLECTION FaceArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива граней ksEntityCollection или IEntityCollection.

GetPlane – Получить указатель на интерфейс плоскости основания уклона

Интерфейс...Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоского объекта ksEntity или IEntity.

SetPlane – Установить указатель на интерфейс плоскости основания уклона

Интерфейс...Справка системы КОМПАС...

CM_MAKE_DRAFT.htm

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоскости операции ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Скругление (Интерфейсы ksFilletDefinition, IFilletDefinition)

Справка системы КОМПАС...

Интерфейс параметров элемента "скругление".

ksFilletDefinition - интерфейс Automation
IFilletDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IFilletDefinition – свойства

radius – Радиус скругления

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

radius = iFilletDefinition.radius	Получить свойство(*)
iFilletDefinition.radius = radius	Установить свойство (*)
radius = iFilletDefinition.GetRadius()	Получить свойство (**)
iFilletDefinition.SetRadius (radius)	Установить свойство (**)

tangent – Признак продолжения скругления по касательным ребрам

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Синтаксис Automation:

tangent = iFilletDefinition.tangent	Получить свойство(*)
iFilletDefinition.tangent = tangent	Установить свойство (*)
tangent = iFilletDefinition.GetTangent()	Получить свойство (**)
iFilletDefinition.SetTangent (tangent)	Установить свойство (**)

Значения свойства:

TRUE - продолжать скругление по касательным ребрам,
FALSE - не продолжать скругление по касательным ребрам.

IFilletDefinition – методы

Array – Получить указатель на интерфейс массива скругляемых объектов (граней и ребер)

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH Array();

Синтаксис COM:

LPENTITYCOLLECTION Array();

Возвращаемое значение:

- указатель на интерфейс массива объектов ksEntityCollection или IEntityCollection.

Примечание:

В возвращаемом массиве включен контроль, не позволяющий добавить в массив нулевые указатели на элементы.

Фаска (Интерфейсы ksChamferDefinition, IChamferDefinition)

Справка системы КОМПАС...

Интерфейс параметров фаски.

ksChamferDefinition	- интерфейс Automation
IChamferDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksChamferDefinition – свойства

tangent – Признак продолжения фаски по касательным ребрам

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Синтаксис Automation:

tangent = iChamferDefinition.tangent		Получить свойство(*)
iChamferDefinition.tangent = tangent		Установить свойство (*)
tangent	=	Получить свойство (**)
iChamferDefinition.GetTangent()		
iChamferDefinition.SetTangent (tangent)		Установить свойство (**)

Значения свойства:

TRUE	- продолжать фаску по касательным ребрам,
FALSE	- не продолжать фаску по касательным ребрам.

ksChamferDefinition – методы

Array – Получить указатель на интерфейс массива ребер и граней, на которых строится фаска

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH Array();

Синтаксис COM:

LPENTITYCOLLECTION Array();

Возвращаемое значение:

- указатель на интерфейс ksEntityCollection или IEntityCollection.

Примечание:

В полученный с помощью данного метода массив нельзя добавлять нулевые указатели.

GetChamferParam – Получить параметры фаски

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetChamferParam (BOOL *transfer,

double *distance1,

double *distance2);

Синтаксис COM:

BOOL GetChamferParam (BOOL *transfer,

double *distance1,

double *distance2);

Выходные параметры:

transfer - признак направления фаски,
distance1 - размер первого катета фаски,
distance2 - размер второго катета фаски.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetChamferParam – Изменить параметры фаски

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetChamferParam (BOOL transferTRUE,

double distance1,

double distance2);

Синтаксис COM:

BOOL SetChamferParam (BOOL transferTRUE,

double distance1,

double distance2);

Входные параметры:

transfer - признак направления фаски,
distance1 - размер первого катета фаски,
distance2 - размер второго катета фаски.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Основание – кинематический элемент (Интерфейсы ksBaseEvolutionDefinition, IBaseEvolutionDefinition)

Справка системы КОМПАС...

Интерфейс параметров основания – кинематического элемента.

ksBaseEvolutionDefinition - интерфейс Automation
IBaseEvolutionDefinition - интерфейс COM

Примечания:

1. Данный интерфейс устарел. Рекомендуется использовать вместо него интерфейс ksBossLoftDefinition.
2. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksBaseEvolutionDefinition – свойства

sketchShiftType – Тип движения сечения по траектории

Интерфейс...Тип данных: short

Значения свойства:

0	- образующая переносится параллельно самой себе,
1	- образующая при переносе сохраняет исходный угол с направляющей,
2	- плоскость образующей выставляется и сохраняется ортогональной направляющей.

Синтаксис Automation:

sketchShiftType = iEvolution.sketchShiftType		Получить свойство(*)
iEvolution.sketchShiftType = sketchShiftType		Установить свойство (*)
sketchShiftType	=	Получить свойство (**)
iEvolution.GetSketchShiftType()		
iEvolution.SetSketchShiftType(sketchShiftType)		Установить свойство (**)

ksBaseEvolutionDefinition – методы

GetPathLength – Получить длину результирующей кривой траектории

Интерфейс...**Синтаксис Automation**:

double GetPathLength (unsigned long bitVector);

Синтаксис COM:

double GetPathLength(unsigned long bitVector);

Входные параметры:

bitVector	- единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

длина результирующей кривой траектории 0	- в случае успешного завершения, - в случае неудачи.

Примечания:

Метод работает только на уже построенной операции.

GetSketch – Получить указатель на интерфейс эскиза сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetThinParam – Получить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

BYTE *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

PathPartArray – Получить указатель на интерфейс массива кривых, задающих траекторию движения сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

IDispatch * PathPartArray();

Синтаксис COM:

LPENTITYCOLLECTION PathPartArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива кривых ksEntityCollection или IEntityCollection.

SetSketch – Установить эскиз сечения кинематического элемента

Интерфейс...**Синтаксис Automation**:

BOOL SetSketch (LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch (LPENTITY sketch);

Входной параметр:

sketch - указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetThinParam – Установить параметры тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,

double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,

BYTE thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...

Возвращаемое значение:

TRUE - в случае успешного завершения.

ThinParam – Получить указатель на интерфейс тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPROPERTY ThinParam();

Возвращаемое значение:

- указатель на интерфейс тонкой стенки ksThinParam или IThinParam.

Эскиз (Интерфейсы ksSketchDefinition, ISketchDefinition)

Справка системы КОМПАС...

Интерфейс параметров эскиза.

ksSketchDefinition	- интерфейс Automation
ISketchDefinition	- интерфейс COM

Примечание:

1. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.
2. На момент редактирования (до вызова метода EndEdit), текущим документом является документ эскиза.

ISketchDefinition – свойства

angle – Угол поворота эскиза относительно проекции системы координат модели на плоскость эскиза (в градусах)

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

angle = iSketchDefinition.angle	Получить свойство(*)
iSketchDefinition.angle = angle	Установить свойство (*)
angle = iSketchDefinition.GetAngle()	Получить свойство (**)
iSketchDefinition.SetAngle(angle)	Установить свойство (**)

ISketchDefinition – методы

AddProjectionOf – Добавить в эскиз проекцию объекта

Интерфейс...Справка системы КОМПАС...

CM_PROJECTION_OBJECT.htm

Синтаксис Automation:

long AddProjectionOf (LPDISPATCH entity);

Синтаксис COM:

long AddProjectionOf (LPENTITY entity);

Входной параметр:

entity	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

- указатель на группу объектов, получившихся в результате проецирования.

Примечание:

1. Данный метод работает только в режиме редактирования эскиза.
2. Параметр entity может быть указателем на ребро, вершину или грань.

BeginEdit – Войти в режим редактирования эскиза (ksDocument2D)

Интерфейс...Справка системы КОМПАС...

236_22_1_Obshch_sved_ob_esk.htm#Raz29527

Синтаксис Automation:

LPDISPATCH BeginEdit();

Возвращаемое значение:

- указатель на интерфейс эскиза ksDocument2D.

Синтаксис COM:

BOOL BeginEdit();

Возвращаемое значение:

TRUE - в случае успешного завершения (режим редактирования включен).

Примечание:

После редактирования необходимо вызвать функцию EndEdit.

EndEdit – Выйти из режима редактирования эскиза

Интерфейс...Справка системы КОМПАС...

236_22_1_Obshch_sved_ob_esk.htm#Raz29527

Синтаксис Automation:

BOOL EndEdit();

Синтаксис COM:

BOOL EndEdit();

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetLocation – Получить смещение системы координат эскиза относительно проекции системы координат модели на плоскость эскиза

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetLocation (double * x, double * y);

Синтаксис COM:

BOOL GetLocation (double * x, double * y);

Выходные параметры:

x, y - смещение вдоль осей.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetLoftPoint – Получить координаты точки в плоскости эскиза

Интерфейс...**Синтаксис Automation**:

BOOL GetLoftPoint (double * x, double * y);

Синтаксис COM:

BOOL GetLoftPoint (double * x, double * y);

Выходные параметры:

x, y - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

При формировании элемента по сечениям (операции ksBaseLoftDefinition, ksBossLoftDefinition, ksCutLoftDefinition) полученная точка определяет ближайшую к ней вершину как точку, через которую пройдет линия, соединяющая эскизы.

GetPlane – Получить базовую плоскость эскиза

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetLoftPoint – Изменить координаты точки в плоскости эскиза

Интерфейс...**Синтаксис Automation**:

BOOL SetLoftPoint (double x, double y);

Синтаксис COM:

BOOL SetLoftPoint (double x, double y);

Входные параметры:

x, y - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

При формировании элемента по сечениям (операции ksBaseLoftDefinition, ksBossLoftDefinition, ksCutLoftDefinition, ksLoftSurfaceDefinition) указанная точка определяет ближайшую к ней вершину как точку, через которую пройдет линия, соединяющая эскизы.

SetLocation – Изменить смещение системы координат эскиза относительно проекции системы координат модели на плоскость эскиза

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetLocation (double x, double y);

Синтаксис COM:

BOOL SetLocation (double x, double y);

Входные параметры:

x, y - смещение вдоль осей.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPlane – Изменить базовую плоскость эскиза

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс базовой плоскости эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

UserSetPlacement – Перейти в режим указания пользователем нового положения эскиза на той же плоскости (или плоской грани), где он был расположен ранее

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL UserSetPlacement (BSTR prompt);

Синтаксис COM:

BOOL UserSetPlacement (LPOLESTR prompt);

Входной параметр:

prompt - текст строки-подсказки.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

После вызова этой функции запускается такой же режим, как после вызова в КОМПАС-3D команды **Разместить эскиз**.

Вырезанный элемент вращения(ИнтерфейсыksCutRotatedDefinition, ICutRotatedDefinition)

Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Интерфейс вырезанного элемента вращения.

ksCutRotatedDefinition - интерфейс Automation
ICutRotatedDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ICutRotatedDefinition – свойства

chooseType – Тип области применения

Интерфейс...Справка системы КОМПАС...

Тип данных: из перечисления ksChooseType

Синтаксис Automation:

chooseType = iCutRotated.chooseType	Получить свойство(*)
iCutRotated.chooseType = chooseType	Установить свойство (*)
chooseType = iCutRotated.GetchooseType()	Получить свойство (**)
iCutRotated.SetchooseType(chooseType)	Установить свойство (**)

cut – Признак результата операции

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - результат операции - вычитание объекта,
FALSE - результат операции - пересечение объектов.

Синтаксис Automation:

cut = iCutRotated.cut	Получить свойство(*)
iCutRotated.cut = cut	Установить свойство (*)
cut = iCutRotated.GetCut()	Получить свойство (**)
iCutRotated.SetCut (cut)	Установить свойство (**)

directionType – Направление вращения

Интерфейс...Справка системы КОМПАС...

Тип данных: short

Значения свойства:

Типы направлений...

Синтаксис Automation:

directionType = iCutRotated.directionType	Получить свойство(*)
iCutRotated.directionType= directionType	Установить свойство (*)
directionType = iCutRotated.GetDirectionType()	Получить свойство (**)
iCutRotated.SetDirectionType (directionType)	Установить свойство (**)

Примечание:

1. Прямое направление - вдоль нормали к плоскости эскиза и всегда против часовой стрелки.
2. Нормаль, проведенная к поверхности тела, всегда выходит из тела.

toroidShapeType – Признак тороида

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE - тороид,
FALSE - сфероид.

Синтаксис Automation:

toroidShapeType = iCutRotated.toroidShapeType	Получить свойство(*)
iCutRotated.toroidShapeType = toroidShapeType	Установить свойство (*)
toroidShapeType = iCutRotated.GetToroidShapeType()	Получить свойство (**)
iCutRotated.SetToroidShapeType (toroidShapeType)	Установить свойство (**)

ICutRotatedDefinition – методы

ChooseBodies – Получить указатель на интерфейс области применения для тел в операции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseBodies();

Синтаксис COM:

LPCHOOSEBODIES ChooseBodies();

Возвращаемое значение:

- указатель на интерфейс области применения для тел ksChooseBodies или IChooseBodies.

ChooseParts – Получить указатель на интерфейс для работы с областью применения для компонентов

Интерфейс...**Синтаксис Automation**:

LPDISPATCH ChooseParts();

Синтаксис COM:

LPCHOOSEBODIES ChooseParts();

Возвращаемое значение:

- указатель на интерфейс для работы с областью применения для компонентов ksChooseParts или IChooseParts.

Примечание.

Интерфейс можно получить только для операции в сборке.

GetSketch – Получить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Синтаксис Automation:

LPDISPATCH GetSketch();

Синтаксис COM:

LPENTITY GetSketch();

Возвращаемое значение:

- указатель на интерфейс эскиза ksEntity или IEntity.

GetSideParam – Получить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Синтаксис Automation:

BOOL GetSideParam (BOOL forward,

double *angle);

Синтаксис COM:

BOOL GetSideParam (BOOL forward,

double * angle);

Входной параметр:

forward - направление вращения:
TRUE - прямое,
FALSE - обратное.

Выходной параметр:

angle - угол вращения.

Возвращаемое значение:

TRUE - в случае успешного завершения.

GetThinParam – Получить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Синтаксис COM:

BOOL GetThinParam (BOOL *thin,

short *thinType,

double *normalThickness,

double *reverseThickness);

Выходные параметры:

thin - признак тонкостенной операции,
thinType - направление построения тонкой стенки,
normalThickness - толщина стенки в прямом направлении,
reverseThickness - толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE - в случае успешного завершения.

RotatedParam – Получить указатель на интерфейс параметров элемента вращения

Интерфейс...Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Синтаксис Automation:

LPDISPATCH RotatedParam();

Синтаксис COM:

LPROTATEDPARAM RotatedParam();

Возвращаемое значение:

- Указатель на интерфейс ksRotatedParam или IRotatedParam.

SetSideParam – Изменить параметры вращения в одном направлении

Интерфейс...Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Синтаксис Automation:

BOOL SetSideParam (short forward,

double angle);

Синтаксис COM:

BOOL SetSideParam (BOOL forward,

double angle);

Входные параметры:

forward	- направление вращения: TRUE - прямое, FALSE - обратное,
angle	- угол вращения.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetSketch – Изменить указатель на интерфейс эскиза элемента

Интерфейс...Справка системы КОМПАС...

CM_BASE_ROTATED_SOLID.htm

Синтаксис Automation:

BOOL SetSketch(LPDISPATCH sketch);

Синтаксис COM:

BOOL SetSketch(LPENTITY sketch);

Входной параметр:

sketch	- указатель на интерфейс эскиза ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetThinParam – Изменить параметры тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetThinParam (BOOL thin,
short thinType,
double normalThickness,
double reverseThickness);

Синтаксис COM:

BOOL SetThinParam (BOOL thin,
short thinType,

double normalThickness,

double reverseThickness);

Входные параметры:

thin	- признак тонкостенной операции,
thinType	- направление построения тонкой стенки,
normalThickness	- толщина стенки в прямом направлении,
reverseThickness	- толщина стенки в обратном направлении.

Типы направлений...**Возвращаемое значение**:

TRUE	- в случае успешного завершения.

ThinParam – Получить указатель на интерфейс параметров тонкой стенки

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH ThinParam();

Синтаксис COM:

LPTHINPARAM ThinParam();

Возвращаемое значение:

- указатель на интерфейс ksThinParam или IThinParam.

Тонкая стенка (Интерфейсы ksThinParam, IThinParam)

Справка системы КОМПАС...

Интерфейс структуры параметров тонкой стенки.

ksThinParam	- интерфейс Automation
IThinParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода ThinParam интерфейсов формообразующих операций.

IThinParam – свойства

normalThickness – Толщина стенки в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

thin = iThinParam.normalThickness	Получить свойство(*)
iThinParam.normalThickness = normalThickness	Установить свойство (*)
normalThickness = iThinParam.GetNormalThickness()	Получить свойство (**)
iThinParam.SetNormalThickness(normalThickness)	Установить свойство (**)

thin – Признак формирования тонкой стенки элемента

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- тонкая стенка формируется,
FALSE	- тонкая стенка не формируется.

Синтаксис Automation:

thin = iThinParam.thin	Получить свойство(*)
iThinParam.thin = thin	Установить свойство (*)
thin = iThinParam.GetThin()	Получить свойство (**)
iThinParam.SetThin (thin)	Установить свойство (**)

thinType – Направление формирования тонкой стенки

Интерфейс...Справка системы КОМПАС...

Тип данных: short

Значения свойства:

Типы направлений...

Синтаксис Automation:

thinType = iThinParam.thinType	Получить свойство(*)
iThinParam.thinType = thinType	Установить свойство (*)
thinType = iThinParam.GetThinType()	Получить свойство (**)
iThinParam.SetThinType(thinType)	Установить свойство (**)

reverseThickness – Толщина стенки в обратном направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

reverseThickness = iThinParam.reverseThickness	Получить свойство(*)
iThinParam.reverseThickness = reverseThickness	Установить свойство (*)
reverseThickness = iThinParam.GetReverseThickness()	Получить свойство (**)
iThinParam.SetReverseThickness(reverse Thickness)	Установить свойство (**)

Интерфейсы дополнительных элементов

Описание вершины (Интерфейсы ksVertexDefinition,IVertexDefinition)

Интерфейс описания вершины.

ksVertexDefinition	- интерфейс Automation
IVertexDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IVertexDefinition – свойства

sketchVertex – Является ли вершина точкой эскиза

Интерфейс...

Тип данных: BOOL

Синтаксис:

sketchVertex = iEllipse3dParam.sketchVertex	Получить свойство (*)
sketchVertex = iEllipse3dParam.GetsketchVertex()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

topologyVertex – Является ли точка топологической вершиной

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

topologyVertex = iVertexDefinition.topologyVertex	Получить свойство (*)
topologyVertex = iVertexDefinition.GetTopologyVertex()	Получить свойство (**)

Примечание:

1. Свойство доступно только для чтения.
2. Точка является топологической вершиной, если принадлежит ребру.

freeVertex – Является ли вершина свободной

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
freeVertex = iVertexDefinition.freeVertex          Получить свойство (* )
freeVertex = iVertexDefinition.GetFreeVertex()     Получить свойство (**)
```

Примечание:

1. Свойство доступно только для чтения.
2. Вершина является свободной, если не принадлежит ребру и не принадлежит элементам эскиза (дуги, отрезки, кривые).

IVertexDefinition – методы

GetOwnerEntity – Получить указатель на объект, породивший вершину

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetOwnerEntity();

Синтаксис COM:

LPENTITY GetOwnerEntity();

Выходные параметры:

x, y, z - координаты вершины.

Возвращаемое значение:

указатель на интерфейс ksEntity или iEntity

GetPoint – Получить координаты вершины

Интерфейс...**Синтаксис Automation**:

BOOL GetPoint (double *x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint (double *x, double *y, double *z);

Выходные параметры:

x, y, z - координаты вершины.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Ребро (Интерфейсы ksEdgeDefinition, IEdgeDefinition)

Интерфейс свойств ребра.

ksEdgeDefinition	- интерфейс Automation
IEdgeDefinition	- интерфейс COM

Описание:

Ребро содержит информацию о математической кривой в трехмерном пространстве, о вершинах и о присоединенных гранях. Ориентация ребра относительно грани является положительной, если грань расположена слева от него или отрицательной, если справа.

Примечание:

Данный интерфейс можно получить, используя следующие методы:

- метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition
- методы интерфейса ksEdgeCollection или IEdgeCollection.

IEdgeDefinition – свойства

sketchEdge – Является ли ребро кривой эскиза

Интерфейс...

Тип данных: BOOL

Синтаксис:

sketchEdge = iEllipse3dParam.sketchEdge		Получить свойство (*)
sketchEdge	=	Получить свойство (**)
iEllipse3dParam.GetSketchEdge()		

Примечание:

Свойство доступно только для чтения.

IEdgeDefinition – методы

EdgeCollection – Получить указатель на интерфейс массива ребер, стыкующихся с данным ребром

Интерфейс...

Синтаксис Automation:

LPDISPATCH EdgeCollection (BOOL begin);

Синтаксис COM:

LPEDGECOLLECTION EdgeCollection (BOOL begin);

Входные параметры:

begin	- TRUE - в начале ребра, - FALSE - в конце ребра.

Возвращаемое значение:

- указатель на интерфейс ksEdgeCollection или IEdgeCollection.

GetAdjacentFace – Получить указатель на интерфейс грани, в цикл которой входит ребро

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetAdjacentFace (BOOL facePlus);

Синтаксис COM:

LPFACEDEFINITION GetAdjacentFace (BOOL facePlus);

Входные параметры:

facePlus	- TRUE - ориентация грани относительно отрезка положительная, - FALSE - ориентация грани относительно отрезка отрицательная.

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinitionили IFaceDefinition.

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- указатель на интерфейс объекта ksCurve3D или ICurve3D.

GetEntity – Получить указатель на интерфейс ребра

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetEntity();

Синтаксис COM:

LPENTITY GetEntity();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity

GetLength – Получить длину ребра

Интерфейс...

Синтаксис Automation:

double GetLength (long bitVector);

Синтаксис COM:

double GetLength (unsigned int bitVector);

Входные параметры:

bitVector - единицы измерения.

Возвращаемое значение:

длина ребра - в случае успеха,
0 - в случае неудачи.

GetOwnerEntity – Получить указатель на объект, породивший ребро

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetOwnerEntity();

Синтаксис COM:

LPENTITY GetOwnerEntity();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetVertex – Получить указатель на интерфейс вершин: начальной и конечной

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetVertex (BOOL start);

Синтаксис COM:

LPVERTEXDEFINITION GetVertex (BOOL start);

Входные параметры:

start	- TRUE - начальная вершина, - FALSE - конечная вершина.

Возвращаемое значение:

- указатель на интерфейс ksVertexDefinition или IVertexDefinition.

IsStraight – Определить, прямолинейно ли ребро

Интерфейс...

Синтаксис Automation:

BOOL IsStraight();

Синтаксис COM:

BOOL IsStraight();

Возвращаемое значение:

TRUE	- ребро прямолинейное,
FALSE	- ребро криволинейное.

OrientedEdgeCollection – Получить указатель на интерфейс массива ориентированных ребер, стыкующихся с данным ребром

Интерфейс...

Синтаксис Automation:

LPDISPATCH OrientedEdgeCollection();

Синтаксис COM:

LPORIENTEDEDGECOLLECTION OrientedEdgeCollection();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdgeCollection или IOrientedEdgeCollection.

IsArc – Определить, является ли кривая дугой окружности

Интерфейс...

Синтаксис Automation:

BOOL IsArc();

Синтаксис COM:

BOOL IsArc();

Возвращаемое значение:

TRUE	- кривая является дугой окружности,
FALSE	- кривая не является окружности.

IsCircle – Определить, является ли кривая окружностью

Интерфейс...

Синтаксис Automation:

BOOL IsCircle();

Синтаксис COM:

BOOL IsCircle();

Возвращаемое значение:

TRUE	- кривая является окружностью,
FALSE	- кривая не является окружностью.

IsEllipse – Определить, является ли кривая эллипсом

Интерфейс...

Синтаксис Automation:

BOOL IsEllipse();

Синтаксис COM:

BOOL IsEllipse();

Возвращаемое значение:

TRUE	- кривая является эллипсом,
FALSE	- кривая не является эллипсом.

IsNurbs – Определить, является ли кривая NURBS

Интерфейс...

Синтаксис Automation:

BOOL IsNurbs();

Синтаксис COM:

BOOL IsNurbs();

Возвращаемое значение:

TRUE	- кривая является NURBS,
FALSE	- кривая не является NURBS.

IsPeriodic – Определить, является ли кривая периодичной

Интерфейс...

Синтаксис Automation:

BOOL IsPeriodic();

Синтаксис COM:

BOOL IsPeriodic();

Возвращаемое значение:

TRUE	- кривая периодична,
FALSE	- кривая непериодична.

Массив граней (Интерфейсы ksFaceCollection, IFaceCollection)

Интерфейс массива граней.

ksFaceCollection	- интерфейс Automation
IFaceCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен следующими способами:

- от интерфейса свойств грани ksFaceDefinition, IFaceDefinition с помощью метода ksFaceDefinition::ConnectedFaceCollection,
- от интерфейса тела ksBody, IBody с помощью методов ksBody::FaceCollection, IBody::FaceCollection.

IFaceCollection – методы

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPFACEDEFINITION First();

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

FindIt – Получить индекс элемента в массиве

Интерфейс...**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt(LPFACEDEFINITION entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity	- указатель на интерфейс компонента ksFaceDefinition или IFaceDefinition.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPFACEDEFINITION GetByIndex (long index);

Входной параметр:

index	- номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

GetByName – Получить указатель на интерфейс объекта в массиве по имени

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByName (BSTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Синтаксис COM:

LPFACEDEFINITION GetByName (LPOLESTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Входные параметры:

name	- название грани,
testFullName	- признак полного имени: TRUE - name - проверять по полному пути, FALSE - имя name может быть частью полного имени,
testIgnoreCase	- признак игнорирования регистра символов: TRUE - игнорировать регистр, FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPFACEDEFINITION Last();

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPFACEDEFINITION Next();

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPFACEDEFINITION Next();

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом граней трехмерного тела и удаляются все предыдущие изменения в массиве.

Грань (Интерфейсы ksFaceDefinition, IFaceDefinition)

Интерфейс свойств грани.

ksFaceDefinition	- интерфейс Automation
IFaceDefinition	- интерфейс COM

Описание:

Грань - это объект топологии. Содержит информацию о геометрии (поверхность), информацию о связях с другими гранями (массив циклов) и признак ориентации нормали. Этот признак имеет значение TRUE, если он совпадает с направлением нормали математической поверхности.

Примечание:

Данный интерфейс может быть получен следующими способами:

- от интерфейса коллекции свойств грани ksFaceCollection, IFaceCollection,
- от интерфейса свойств ребра ksEdgeDefinition, IEdgeDefinition.
- используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IFaceDefinition – свойства

normalOrientation – Признак ориентации нормали

Интерфейс..Тип данных: BOOL

Синтаксис Automation:

normalOrient = iFaceDefinition.normalOrientation	Получить свойство (*)
iFaceDefinition.GetNormalOrientation()	Получить свойство (**)

Возвращаемое значение:

TRUE	- направление нормали совпадает с направлением нормали математической поверхности,
FALSE	- направление нормали не совпадает с направлением нормали математической поверхности ksSurface.

Примечание:

Свойство доступно только для чтения.

IFaceDefinition – методы

ConnectedFaceCollection – Получить указатель на интерфейс массива граней, стыкующихся с данной гранью

Интерфейс..**Синтаксис Automation**:

LPDISPATCH ConnectedFaceCollection();

Синтаксис COM:

LPFACECOLLECTION ConnectedFaceCollection();

Возвращаемое значение:

- указатель на интерфейс ksFaceCollection или IFaceCollection.

EdgeCollection – Получить указатель на интерфейс массива ребер, ограничивающих грань

Интерфейс..**Синтаксис Automation**:

LPDISPATCH EdgeCollection();

Синтаксис COM:

LPEDGECOLLECTION EdgeCollection();

Возвращаемое значение:

- указатель на интерфейс ksEdgeCollection или IEdgeCollection.

GetArea – Получить площадь грани

Интерфейс..**Синтаксис Automation**:

double GetArea (long bitVector);

Синтаксис COM:

double GetArea (unsigned long bitVector);

Входные параметры:

bitVector	- единицы измерения.

Возвращаемое значение:

площадь грани	- в случае успешного завершения.
0	- в случае неудачи.

GetCylinderParam – Получить параметры цилиндрической грани

Интерфейс..**Синтаксис Automation**:

BOOL GetCylinderParam (double* h,

double* r);

Синтаксис COM:

BOOL GetCylinderParam (double * h,

double * r);

Выходные параметры:

h - высота цилиндра,
r - радиус цилиндра.

Возвращаемое значение:

TRUE - в случае успеха.

GetEntity – Получить указатель на интерфейс грани

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetEntity();

Синтаксис COM:

LPENTITY GetEntity();

Возвращаемое значение:

- указатель на интерфейс грани ksEntity или IEntity

GetNextFace – Получить указатель на интерфейс следующей грани в этом теле

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetNextFace();

Синтаксис COM:

LPFACEDEFINITION GetNextFace();

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

GetOwnerEntity – Получить указатель на объект, породивший цилиндрическую грань

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetOwnerEntity();

Синтаксис COM:

LPENTITY GetOwnerEntity();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

GetTessellation – Получить указатель на интерфейс триангуляции

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetTessellation();

Синтаксис COM:

LPTESSELLATION GetTessellation();

Возвращаемое значение:

- указатель на интерфейс ksTessellation или ITessellation.

IsCone – Определить, коническая ли грань

Интерфейс..**Синтаксис Automation**:

BOOL IsCone();

Синтаксис COM:

BOOL IsCone();

Возвращаемое значение:

TRUE	- грань коническая,
FALSE	- грань не коническая.

IsConnectedWith – Проверить, связаны ли грани

Интерфейс..**Синтаксис Automation**:

BOOL IsConnectedWith (LPDISPATCH other);

Синтаксис COM:

BOOL IsConnectedWith (LPFACEDEFINITION other);

Входные параметры:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition другой грани.

Возвращаемое значение:

TRUE	- если грани связаны,
FALSE	- если грани не связаны.

IsCylinder – Определить, цилиндрическая ли грань

Интерфейс..**Синтаксис Automation**:

BOOL IsCylinder();

Синтаксис COM:

BOOL IsCylinder();

Возвращаемое значение:

TRUE	- грань цилиндрическая,
FALSE	- грань не цилиндрическая.

IsNurbsSurface – Определить, является ли грань NURBS поверхностью

Интерфейс..**Синтаксис Automation**:

BOOL IsNurbsSurface();

Синтаксис COM:

BOOL IsNurbsSurface();

Возвращаемое значение:

TRUE	- грань является NURBS поверхностью,
FALSE	- грань не является NURBS поверхностью.

IsPlanar – Определить, плоская ли грань

Интерфейс...**Синтаксис Automation**:

BOOL IsPlanar();

Синтаксис COM:

BOOL IsPlanar();

Возвращаемое значение:

TRUE	- грань плоская,
FALSE	- грань не плоская.

IsRevolved – Определить, является ли грань поверхностью вращения

Интерфейс..**Синтаксис Automation**:

BOOL IsRevolved();

Синтаксис COM:

BOOL IsRevolved();

Возвращаемое значение:

TRUE	- грань является поверхностью вращения,
FALSE	- грань не является поверхностью вращения.

IsSphere – Определить, является ли грань сферической

Интерфейс..**Синтаксис Automation**:

BOOL IsSphere();

Синтаксис COM:

BOOL IsSphere();

Возвращаемое значение:

TRUE	- грань является сферической,
FALSE	- грань не является сферической.

IsSwept – Определить, задается ли грань поверхностью по траектории

Интерфейс..**Синтаксис Automation**:

BOOL IsSwept();

Синтаксис COM:

BOOL IsSwept();

Возвращаемое значение:

TRUE	- грань задается поверхностью по траектории,
FALSE	- грань не задается поверхностью по траектории.

IsTorus – Определить, является ли грань тором

Интерфейс..**Синтаксис Automation**:

BOOL IsTorus();

Синтаксис COM:

BOOL IsTorus();

Возвращаемое значение:

TRUE - грань является тором,
FALSE - грань не является тором.

IsValid – Получить индикатор доступности объекта.

Интерфейс..**Синтаксис Automation**:

BOOL IsValid();

Синтаксис COM:

BOOL IsValid();

Возвращаемое значение:

TRUE - объект доступен,
FALSE - объект недоступен.

LoopCollection – Получить указатель на интерфейс массива циклов

Интерфейс..**Синтаксис Automation**:

LPDISPATCH LoopCollection();

Синтаксис COM:

LPLOOPCOLLECTION LoopCollection();

Возвращаемое значение:

- указатель на интерфейс ksLoopCollection или ILoopCollection.

Описание импортированной поверхности (Интерфейсы ksImportedSurfaceDefinition, IImportedSurfaceDefinition)

Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Интерфейс параметров импортированной поверхности.

ksImportedSurfaceDefinition - интерфейс Automation
IImportedSurfaceDefinition - интерфейс COM

Примечание:

1. Поверхность можно импортировать, задав массив упорядоченных массивов точек. Один упорядоченный массив точек IVertexCollection определяет одну кривую. Если первая и

последняя точки кривой совпадают, это является признаком замкнутости кривой. Для нормального построения поверхности кривых должно быть, как минимум, две.

2. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IImportedSurfaceDefinition – методы

Добавить кривую по массиву точек

Пример...

Интерфейс...Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

BOOL AddCurve (VARIANT arr);

Синтаксис COM:

BOOL AddCurve (VARIANT arr);

Входные параметры:

arr	- массив координат точек кривой.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Тип переменной arr - VT_ARRAY | VT_R8. Одномерный SAFEARRAY массив double значений.
2. Каждая точка представлена тремя координатами x, y, z. Следовательно число точек в массиве должно быть кратно.

AddPoint – Добавить точку в определение новой кривой

Интерфейс...Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

BOOL AddPoint(double x, double y, double z);

Синтаксис COM:

BOOL AddPoint(double x, double y, double z);

Входные параметры:

x, y, z - координаты точки кривой.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Перед добавлением точек кривой нужно открыть описание кривой методом ksImportedSurfaceDefinition::BeginCurve.

После добавления точек необходимо закрыть определение кривой методом ksImportedSurfaceDefinition::EndCurve.

BeginCurve – Начать создание новой кривой

Интерфейс...Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

BOOL BeginCurve();

Синтаксис COM:

BOOL BeginCurve();

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

Для добавления точек кривой используется метод ksImportedSurfaceDefinition::AddPoint. После добавления точек необходимо завершить задание кривой методом ksImportedSurfaceDefinition::EndCurve.

Clear – Очистить содержимое

Интерфейс...Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Очищается имя файла и массив кривых.

EndCurve – Завершить создание новой кривой

Интерфейс...Справка системы КОМПАС...

CM_IMPORTEDSURFACE.htm

Синтаксис Automation:

BOOL EndCurve();

Синтаксис COM:

BOOL EndCurve();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Перед добавлением точек кривой нужно открыть описание кривой методом ksImportedSurfaceDefinition::BeginCurve.

Для добавления точек используется метод ksImportedSurfaceDefinition::AddPoint.

Условное изображение резьбы (Интерфейсы ksThreadDefinition, IThreadDefinition)

Справка системы КОМПАС...

Интерфейс условного изображения резьбы.

ksThreadDefinition	- интерфейс Automation
IThreadDefinition	- интерфейс COM

Примечание:

1. Определяется параметрами резьбы: номинальный диаметр резьбы, длина резьбы, шаг резьбы, тип резьбы (внутренняя, внешняя), направление построения резьбы, а также объектами необходимыми для построения: базовый объект (ребро или грань), начальная и конечная грань.
2. Если базовым объектом является ребро, то базовая грань определяется автоматически. При этом задание начальной и конечной грани не обязательно, хотя возможно. Если базовым объектом является грань, то указывать начальную и конечную грани необходимо. Базовая грань должна быть цилиндрической либо конической, начальная и конечная

грань - любые грани, ограничивающие резьбу. Базовое ребро должно быть окружностью, принадлежащей цилиндру.

3. Если базовое ребро изменено, то при вызове методов Update или Create автоматически будет произведена проверка направления резьбы. В случае ошибки направление изменится.
4. Начальная и конечная грань применяются для определения длины резьбы при включенной опции автоматически определения длины. Кроме того, начальная грань указывает, откуда откладывается резьба.
5. Если включено автоопределение диаметра, то номинальный диаметр резьбы не зависит от значения указанного пользователем. Если включено автоопределение длины резьбы, длина не будет зависеть от значения указанного пользователем.
6. Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IThreadDefinition – свойства

allLength – Полная длина

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- длина резьбы length не зависит от значения указанного пользователем,
FALSE	- длина резьбы задается пользователем.

Синтаксис Automation:

allLength = iThreadDefinition.allLength	Получить свойство (*)
iThreadDefinition.allLength = allLength	Установить свойство (*)
allLength = iThreadDefinition.GetAllLength()	Получить свойство (**)
iThreadDefinition.SetAllLength(allLength)	Установить свойство (**)

autoDefinDr – Автоопределение диаметра

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- номинальный диаметр резьбы dr не зависит от значения указанного пользователем,

FALSE - номинальный диаметр резьбы задается пользователем.

Синтаксис Automation:

autoDefinDr = iThreadDefinition.autoDefinDr	Получить свойство (*)
iThreadDefinition.autoDefinDr = autoDefinDr	Установить свойство (*)
iThreadDefinition.GetAutoDefinDr()	Получить свойство (**)
iThreadDefinition.SetAutoDefinDr(autoDefinDr)	Установить свойство (**)

dr – Номинальный диаметр резьбы

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

dr = iThreadDefinition.dr	Получить свойство (*)
iThreadDefinition.dr = dr	Установить свойство (*)
dr = iThreadDefinition.GetDr()	Получить свойство (**)
iThreadDefinition.SetDr(dr)	Установить свойство (**)

Примечание:

Если свойство автоопределения диаметра autoDefinDrимеет значение TRUE, номинальный диаметр резьбы не будет зависеть от значения указанного пользователем.

faceValue– Направление построения резьбы

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Синтаксис Automation:

faceValue = iThreadDefinition.faceValue	Получить свойство (*)
iThreadDefinition.faceValue = faceValue	Установить свойство (*)
iThreadDefinition.GetFaceValue()	Получить свойство (**)
iThreadDefinition.SetFaceValue(faceValue)	Установить свойство (**)

Примечание:

Если базовое ребро изменено, то при вызове методов Update или Create автоматически будет произведена проверка направления резьбы. В случае ошибки направление изменится.

length – Длина резьбы

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

length = iThreadDefinition.length	Получить свойство (*)
iThreadDefinition.length = length	Установить свойство (*)
length = iThreadDefinition.GetLength()	Получить свойство (**)
iThreadDefinition.SetLength(length)	Установить свойство (**)

Примечание:

Если свойство автоопределения длины резьбы allLength имеет значение TRUE, длина резьбы не будет зависеть от значения указанного пользователем.

outside – Тип резьбы

Интерфейс...Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- резьба внешняя,
FALSE	- резьба внутренняя.

Синтаксис Automation:

outside = iThreadDefinition.outside	Получить свойство (*)
outside = iThreadDefinition.GetOutside()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

p – Шаг резьбы

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

p = iThreadDefinition.p	Получить свойство (*)
iThreadDefinition.p = p	Установить свойство (*)
p = iThreadDefinition.GetP()	Получить свойство (**)
iThreadDefinition.SetP(p)	Установить свойство (**)

IThreadDefinition – методы

GetBaseObject – Получить указатель на интерфейс базового объекта (ребра или грани), по которому строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetBaseObject();

Синтаксис COM:

LPENTITY GetBaseObject();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечание:

1. Базовый объект может быть как ребром, так и гранью. Тип базового объекта определяется методами интерфейсов ksEntity или IEntity.
2. Если базовым объектом является ребро, то базовая грань определяется автоматически. При этом задание начальной и конечной грани не обязательно, хотя возможно. Если базовым объектом является грань, то указывать начальную и конечную грани необходимо. Базовая грань должна быть цилиндрической либо конической, начальная и конечная грань - любые грани, ограничивающие резьбу. Базовое ребро должно быть окружностью, принадлежащей цилиндру.

GetFaceBegin – Получить указатель на интерфейс базового объекта (ребра или грани), от которого строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetFaceBegin();

Синтаксис COM:

LPENTITY GetFaceBegin();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

Примечание:

Начальная и конечная грань - любые грани, ограничивающие резьбу. Начальная и конечная грань применяются для определения длины резьбы при включенной опции автома-

тически определения длины. Кроме того, начальная грань указывает, откуда откладывается резьба.

GetFaceEnd – Получить указатель на интерфейс базового объекта (ребра или грани), до которого строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetFaceEnd();

Синтаксис COM:

LPENTITY GetFaceEnd();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

Примечание:

Начальная и конечная грань - любые грани, ограничивающие резьбу. Начальная и конечная грань применяются для определения длины резьбы при включенной опции автоматического определения длины. Кроме того, начальная грань указывает, откуда откладывается резьба.

SetBaseObject – Установить ребро или грань, по которой строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetBaseObject(LPDISPATCH obj);

Синтаксис COM:

BOOL SetBaseObject(LPENTITY obj);

Входной параметр:

obj - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Базовый объект может быть как ребром, так и гранью. Тип базового объекта определяется методами интерфейсов ksEntity или IEntity.
2. Если базовым объектом является ребро, то базовая грань определяется автоматически. При этом задание начальной и конечной грани не обязательно, хотя возможно. Если базовым объектом является грань, то указывать начальную и конечную грани необходимо. Базовая грань должна быть цилиндрической либо конической, начальная и конечная грань - любые грани, ограничивающие резьбу. Базовое ребро должно быть окружностью, принадлежащей цилиндру.
3. Если базовое ребро изменено, то при вызове методов Update или Create автоматически будет произведена проверка направления резьбы. В случае ошибки направление изменится.

SetFaceBegin – Установить грань, от которой строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetFaceBegin(LPDISPATCH face);

Синтаксис COM:

BOOL SetFaceBegin(LPENTITY face);

Входной параметр:

face - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Начальная и конечная грань - любые грани, ограничивающие резьбу. Начальная и конечная грань применяются для определения длины резьбы при включенной опции автоматического определения длины. Кроме того, начальная грань указывает, откуда откладывается резьба.

SetFaceEnd – Установить грань, до которой строится резьба

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetFaceEnd (LPDISPATCH face);

Синтаксис COM:

BOOL SetFaceEnd (LPENTITY face);

Входной параметр:

face	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Начальная и конечная грань - любые грани, ограничивающие резьбу. Начальная и конечная грань применяются для определения длины резьбы при включенной опции автоматического определения длины. Кроме того, начальная грань указывает, откуда откладывается резьба.

Массив циклов грани (Интерфейсы ksLoopCollection, ILoopCollection)

Интерфейс массива циклов.

ksLoopCollection	- интерфейс Automation
ILoopCollection	- интерфейс COM

Примечание:

Данный интерфейс можно получить от интерфейса свойств грани ksFaceDefinition или IFaceDefinition.

ILoopCollection – методы

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPLOOP GetByIndex (long index);

Входной параметр:

index	- номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksLoop или ILoop.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

FindIt – Получить индекс элемента в массиве

Интерфейс...**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPLOOP entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве.
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity	- указатель на интерфейс компонента ksLoop или ILoop.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPLOOP First();

Возвращаемое значение:

- указатель на интерфейс ksLoop или ILoop.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPLOOP Last();

Возвращаемое значение:

- указатель на интерфейс ksLoop или ILoop.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPLOOP Next();

Возвращаемое значение:

- указатель на интерфейс ksLoop или ILoop.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPLOOP Prev();

Возвращаемое значение:

- указатель на интерфейс ksLoop или ILoop.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом циклов грани и удаляются все предыдущие изменения в массиве.

Цикл грани(Интерфейсы ksLoop, lLoop)

Интерфейс цикла грани.

ksLoop - интерфейс Automation
lLoop - интерфейс COM

Описание:

Цикл - это объект топологии. Содержит информацию о ребрах и связях грани с другими гранями. Грань может содержать несколько циклов, например, если в ней есть отверстия.

Примечание:

Данный интерфейс можно получить от интерфейса массива циклов ksLoopCollection или lLoopCollection.

lLoop – методы

EdgeCollection – Получить массив ребер, ограничивающих грань

Интерфейс..**Синтаксис Automation**:

LPDISPATCH EdgeCollection();

Синтаксис COM:

LPEDGECOLLECTION EdgeCollection();

Возвращаемое значение:

- указатель на интерфейс ksEdgeCollection или lEdgeCollection.

GetLength – Получить общую длину ребер

Интерфейс..**Синтаксис Automation**:

double GetLength (unsigned long bitVector);

Синтаксис COM:

double GetLength (unsigned long bitVector);

Входные параметры:

bitVector - единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

общая длина ребер	- в случае успеха,
0	- в случае неудачи.

IsOuter – Определить, является ли цикл внешним

Интерфейс..**Синтаксис Automation**:

BOOL IsOuter();

Синтаксис COM:

BOOL IsOuter();

Возвращаемое значение:

TRUE	- цикл является внешним,
FALSE	- цикл не является внешним.

OrientedEdgeCollection – Получить массив ориентированных ребер

Интерфейс..Получить массив ориентированных ребер.

Синтаксис Automation:

LPDISPATCH OrientedEdgeCollection (LPDISPATCH edge);

Синтаксис COM:

LPORIENTEDEDGECOLLECTION OrientedEdgeCollection (LPEDGEDEFINITION edge);

Входные параметры:

edge	- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdgeCollection или IOrientedEdgeCollection.

Примечание:

Если параметр edge = NULL, то метод вернет весь массив. Если edge <> NULL, то будет возвращен массив ориентированных ребер, которые содержат данное ребро.

Интерфейсы вспомогательной геометрии

Интерфейсы объекта "Контрольная точка" – ksControlPointDefinition, IControlPointDefinition

Справка системы КОМПАС...

Интерфейс параметров объекта "Контрольная точка"

ksControlPointDefinition	- интерфейс Automation
IControlPointDefinition	- интерфейс COM

Описание:

Объект "Контрольная точка" создается в детали или сборке и используется для размещения этой детали или сборки в указанной вершине траектории трассы.

"Контрольная точка" может указываться в качестве вершины при создании или редактировании траектории.

В детали (или сборке) может находиться любое количество объектов "Контрольная точка".

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IControlPointDefinition – методы

GetPoint – Получить координаты точки

Интерфейс...**Синтаксис Automation**:

VARIANT_BOOL GetPoint (double * X, double * Y, double * Z);

Синтаксис COM:

BOOL GetPoint (double * X, double * Y, double * Z);

Выходные параметры:

X, Y, Z	- координаты точки.

GetVertex – Получить указатель на интерфейс опорной вершины

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetVertex();

Синтаксис COM:

LPENTITY GetVertex();

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

Примечание:

Возвращает указатель на интерфейс опорной вершины, на которой построена контрольная точка. Опорной вершиной может быть объект "вершина" (тип o3d_vertex).

SetVertex – Установить указатель на интерфейс опорной вершины

Интерфейс...**Синтаксис Automation**:

VARIANT_BOOL SetVertex (LPDISPATCH Val);

Синтаксис COM:

BOOL SetVertex (LPENTITY Val);

Входной параметр:

Val	- указатель на интерфейс опорной вершины ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Позволяет задать опорную вершину, на которой будет построена контрольная точка. Опорной вершиной может быть объект "вершина" (тип o3d_vertex).

Интерфейсы объекта "Присоединительная точка"ksConjunctivePointDefinition, IConjunctivePointDefinition

Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Интерфейс параметров объекта "Присоединительная точка".

ksConjunctivePointDefinition	- интерфейс Automation
IConjunctivePointDefinition	- интерфейс COM

Описание::

Объект "Присоединительная точка" создается в детали (или сборке) и используется для связывания элементов между собой и с траекторией трассы. "Присоединительная точка" может указываться в качестве вершины при создании или редактировании траектории.

В детали (или сборке) может находиться любое количество объектов "Присоединительная точка".

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IConjunctivePointDefinition – свойства

Direction – Направление

Интерфейс...Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Тип данных: BOOL

Синтаксис Automation:

Direction = iObject.Direction	Получить свойство(*)
iObject.Direction = Direction	Установить свойство (*)
Direction = iObject.GetDirection()	Получить свойство (**)
iObject.SetDirection (Direction)	Установить свойство (**)

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Примечание:

Позволяет задать конкретное направление вектора. Направление вектора в присоединительной точке зависит от выбора опорного объекта SetEdge.

IConjunctivePointDefinition – методы

GetVertex – Получить указатель на интерфейс опорной вершины

Интерфейс...Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Синтаксис Automation:

LPDISPATCH GetVertex();

Синтаксис COM:

LPENTITY GetVertex();

Возвращаемое значение:

- Указатель на интерфейс IEntity или ksEntity опорной вершины.

Примечания.

Возвращает указатель на интерфейс опорной вершины, на которой построена присоединительная точка. Опорной вершиной может быть объект "вершина" (тип o3d_vertex).

GetEdge – Получить указатель на интерфейс опорного объекта для определения вектора направления

Интерфейс...Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Синтаксис Automation:

LPDISPATCH GetEdge();

Синтаксис COM:

LPENTITY GetEdge();

Входные параметры:

Указатель на интерфейс ksEntity или IEntity опорного объекта.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечания.

Возвращает указатель на интерфейс опорного объекта, для определения вектора направления. Опорными объектами могут быть:

- грани,
- конструктивные плоскости,
- ребра,
- вспомогательные оси.

GetPoint – Получить указатель на интерфейс опорного объекта для определения вектора направления

Интерфейс...Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Синтаксис Automation:

VARIANT_BOOL GetPoint (double * X, double * Y, double * Z);

Синтаксис COM:

BOOL GetPoint (double * X, double * Y, double * Z);

Выходные параметры:

X, Y, Z	- координаты точки.

SetEdge – Установить указатель на интерфейс опорного объекта для определения вектора направления

Интерфейс...Справка системы КОМПАС...

`Postroenye_kontr_tochek.htm#connecting_point`

Синтаксис Automation:

VARIANT_BOOL SetEdge (LPDISPATCH Val);

Синтаксис COM:

BOOL SetEdge (LPENTITY Val);

Входные параметры:

Val	- Указатель на интерфейс ksEntity или IEntity опорного объекта.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечания.

Позволяет задать опорный объект, по которому будет определяться вектор направления. Опорными объектами могут быть:

- грани,
- конструктивные плоскости,
- ребра,
- вспомогательные оси.

Если в качестве опорного объекта выбрана грань или конструктивная плоскость, то вектор будет перпендикулярен опорному объекту. Если в качестве опорного объекта выбрано ребро или вспомогательная ось, то вектор будет параллелен опорному объекту. Чтобы задать конкретное (прямое или обратное) направление вектора относительно опорного объекта, необходимо установить соответствующее значение свойству Direction.

SetVertex – Установить указатель на интерфейс опорной вершины

Интерфейс...Справка системы КОМПАС...

Postroenye_kontr_tochek.htm#connecting_point

Синтаксис Automation:

VARIANT_BOOL SetVertex (LPDISPATCH Val);

Синтаксис COM:

BOOL SetVertex (LPENTITY Val);

Входные параметры:

Val - Указатель на интерфейс IEntity или ksEntity опорной вершины.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечания.

Позволяет задать опорную вершину, на которой будет построена присоединительная точка. Опорной вершиной может быть объект "вершина" (тип o3d_vertex).

Плоскость через ребро и вершину (Интерфейсы ksPlaneEdgePointDefinition,IPlaneEdgePointDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_edge_point

Интерфейс параметров плоскости, проходящей через ребро и вершину.

ksPlaneEdgePointDefinition - интерфейс Automation
IPlaneEdgePointDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneEdgePointDefinition – методы

GetEdge – Получить указатель на интерфейс ребра, через которое требуется построить плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_edge_point`

Синтаксис Automation:

LPDISPATCH GetEdge();

Синтаксис COM:

LPENTITY GetEdge();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetPoint – Получить указатель на интерфейс вершины, через которую требуется построить плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_edge_point`

Синтаксис Automation:

LPDISPATCH GetPoint();

Синтаксис COM:

LPENTITY GetPoint();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_edge_point`

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetEdge – Установить указатель на интерфейс ребра, через которое требуется построить плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_edge_point

Синтаксис Automation:

BOOL SetEdge (LPDISPATCH edge);

Синтаксис COM:

BOOL SetEdge (LPENTITY edge);

Входной параметр:

edge - указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPoint – Установить указатель на интерфейс вершины, через которую требуется построить плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_edge_point

Синтаксис Automation:

BOOL SetPoint (LPDISPATCH point);

Синтаксис COM:

BOOL SetPoint (LPENTITY point);

Входной параметр:

point - указатель на интерфейс вершины ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Средняя плоскость (Интерфейсы ksPlaneMiddleDefinition, IPlaneMiddleDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_sredn

Интерфейс параметров конструктивной плоскости "Средняя плоскость".

ksPlaneMiddleDefinition	- интерфейс Automation
IPlaneMiddleDefinition	- интерфейс COM

Описание::

Плоскость проходит через биссектрису для двух плоских или прямолинейных объектов. Может быть построена для двух прямолинейных или плоских объектов типа *Ребро*, *Конструктивная ось*, *Грань (плоская)*, *Конструктивная плоскость*.

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition

IPlaneMiddleDefinition – свойства

position – Положение плоскости

Интерфейс...Тип данных: BOOL

Значения свойства:

TRUE	- положение 1: позволяет построить биссекторную плоскость,
FALSE	- положение 2: позволяет построить плоскость, перпендикулярную биссекторной и проходящую через прямую или точку пересечения опорных объектов.

Синтаксис Automation:

position = iDocument3D.position		Получить свойство(*)
iDocument3D.position = position		Установить свойство (*)
position = iDocument3D.GetPosition()		Получить свойство (**)
iDocument3D.SetPosition (position)		Установить свойство (**)

IPlaneMiddleDefinition- методы

GetObject- Получить указатель на интерфейс опорного объекта с указанным номером

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetObject (long Number);

Синтаксис COM:

LPENTITY GetObject (long Number);

Входные параметры:

Number - номер объекта (1 или 2).

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity опорного объекта.

Примечание:

Возвращает указатель на интерфейс опорного объекта, на котором построена средняя плоскость. Опорным объектом может быть *Ребро*, *Конструктивная ось*, *Грань (плоская)*, *Конструктивная плоскость*.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

указатель на интерфейс ksSurface или ISurface.

SetObject – Установить указатель на интерфейс опорного объекта с указанным номером

Интерфейс...**Синтаксис Automation**:

VARIANT_BOOL SetObject (long Number, LPDISPATCH Val);

Синтаксис COM:

BOOL SetObject (long Number, LPENTITY Val);

Входные параметры:

Number - номер объекта (1 или 2).
Val - указатель на интерфейс ksEntity или IEntity опорного объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Позволяет задать опорный объект, на котором будет построена средняя плоскость. Опорным объектом может быть *Ребро*, *Конструктивная ось*, *Грань (плоская)*, *Конструктивная плоскость*.

Смещенная плоскость (Интерфейсы ksPlaneOffsetDefinition, IPlaneOffsetDefinition)

Справка системы КОМПАС...

`Postroenie_ploskostey.htm#smeshchennaja_ploskost`

Интерфейс параметров смещенной плоскости.

ksPlaneOffsetDefinition - интерфейс Automation
IPlaneOffsetDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneOffsetDefinition – свойства

direction– Направление смещения от базовой плоскости

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#smeshchennaja_ploskost`

Тип данных: BOOL

Синтаксис Automation:

direction = iPlaneOffset.direction	Получить свойство(*)
iPlaneOffset.direction = direction	Установить свойство (*)
direction = iPlaneOffset.GetDirection()	Получить свойство (**)
iPlaneOffset.SetDirection(direction)	Установить свойство (**)

Значения свойства:

TRUE	- прямое направление,
FALSE	- обратное направление.

Примечание:

Прямое направление совпадает с направлением нормали к базовой плоскости.

offset – Смещение (расстояние) от базовой плоскости

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#smeshchennaja_ploskost`

Тип данных: double

Синтаксис Automation:

offset = iPlaneOffset.offset	Получить свойство(*)
iPlaneOffset.offset = offset	Установить свойство (*)
offset = iPlaneOffset.GetOffset()	Получить свойство (**)
iPlaneOffset.SetOffset(offset)	Установить свойство (**)

IPlaneOffsetDefinition – методы

GetPlane – Получить указатель на интерфейс базовой плоскости

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#smeshchennaja_ploskost`

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#smeshchennaja_ploskost

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс математической поверхности ksSurface или ISurface.

SetPlane – Изменить указатель на интерфейс базовой плоскости

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#smeshchennaja_ploskost

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Плоскость через ребро параллельно/перпендикулярно грани (Интерфейсы ksPlaneLineToPlaneDefinition, IPlaneLineToPlaneDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran

Интерфейс параметров плоскости, проходящей через ребро параллельно или перпендикулярно грани.

ksPlaneLineToPlaneDefinition - интерфейс Automation
IPlaneLineToPlaneDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneLineToPlaneDefinition – свойства

parallel – Признак положения плоскости

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran
Тип данных: BOOL

Значения свойства:

TRUE	- параллельно грани,
FALSE	- перпендикулярно грани.

Синтаксис Automation:

parallel = iPlaneLineToPlane.parallel		Получить свойство(*)
iPlaneLineToPlane.parallel = parallel		Установить свойство (*)
parallel	=	Получить
iPlaneLineToPlane.GetParallel()		свойство (**)
iPlaneLineToPlane.SetParallel(parallel)		Установить свойство (**)

IPlaneLineToPlaneDefinition – методы

GetEdge – Получить указатель на интерфейс ребра, через которое будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran
Синтаксис Automation:

LPDISPATCH GetEdge();

Синтаксис COM:

LPDISPATCH GetEdge();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity.

GetPlane – Получить указатель на интерфейс плоского объекта, параллельно или перпендикулярно которому будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran`

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran`

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetEdge – Установить указатель на интерфейс ребра, через которое будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran`

Синтаксис Automation:

BOOL SetEdge (LPDISPATCH edge);

Синтаксис COM:

BOOL SetEdge (LPENTITY edge);

Входной параметр:

edge - указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPlane – Установить указатель на интерфейс плоского объекта, параллельно или перпендикулярно которому будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_rebro_i_gran`

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Плоскость через три вершины (Интерфейсы ksPlane3PointsDefinition, IPlane3PointsDefinition)

Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_3points`

Интерфейс параметров вспомогательной плоскости, проходящей через три точки.

ksPlane3PointsDefinition - интерфейс Automation
IPlane3PointsDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksPlane3PointsDefinition – методы

GetPoint – Получить указатель на интерфейс точки с указанным номеромt

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_3points`

Синтаксис Automation:

LPDISPATCH GetPoint (long number);

Синтаксис COM:

LPENTITY GetPoint (long number);

Входной параметр:

number - номер точки (от 1 до 3).

Возвращаемое значение:

- указатель на интерфейс точки ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetPoint – Изменить указатель на интерфейс точки с указанным номером

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_3points`

Синтаксис Automation:

BOOL SetPoint (long number, LPDISPATCH point);

Синтаксис COM:

BOOL SetPoint (long number, LPENTITY point);

Входные параметры:

point	- указатель на интерфейс точки ksEntity или IEntity,
number	- номер точки (от 1 до 3).

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Нормальная плоскость (Интерфейсы ksPlaneNormalToSurfaceDefinition, IPlaneNormalToSurfaceDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_norm

Интерфейс параметров нормальной плоскости.

ksPlaneNormalToSurfaceDefinition	- интерфейс Automation
IPlaneNormalToSurfaceDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneNormalToSurfaceDefinition – свойства

ang – Угол между плоскостью и опорным плоским объектом

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_norm

Тип данных: double

Синтаксис Automation:

ang = iPlaneNormalToSurface.ang		Получить свойство(*)
iPlaneNormalToSurface.ang = ang		Установить свойство (*)
ang	=	Получить
iPlaneNormalToSurface.GetAngle()		свойство (**)
iPlaneNormalToSurface.SetAngle(ang)		Установить свойство (**)

IPlaneNormalToSurfaceDefinition- методы

GetFace - Получить указатель на интерфейс цилиндрической или конической грани, к которой требуется построить нормальную плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_norm

Синтаксис Automation:

LPDISPATCH GetFace();

Синтаксис COM:

LPENTITY GetFace();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetPlane - Получить указатель на интерфейс плоского объекта, параллельно которому должна пройти новая плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_norm

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс объекта ksEntity или IEntity.

GetSurface - Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_norm

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetFace – Установить указатель на интерфейс цилиндрической или конической грани, к которой требуется построить нормальную плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_norm`

Синтаксис Automation:

BOOL SetFace (LPDISPATCH face);

Синтаксис COM:

BOOL SetFace (LPENTITY face);

Входной параметр:

plane	- указатель на интерфейс цилиндрической или конической грани ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetPlane – Установить указатель на интерфейс плоского объекта, параллельно которому должна пройти новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_norm`

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane	- указатель на интерфейс объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Плоскость перпендикулярная ребру (Интерфейсы ksPlanePerpendicularDefinition, IPlanePerpendicularDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Интерфейс параметров плоскости, проходящей через вершину перпендикулярно ребру.

ksPlanePerpendicularDefinition - интерфейс Automation
IPlanePerpendicularDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlanePerpendicularDefinition – методы

GetEdge – Получить указатель на интерфейс ребра, перпендикулярно которому будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

LPDISPATCH GetEdge();

Синтаксис COM:

LPENTITY GetEdge();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity.

GetPoint – Получить указатель на интерфейс вершины, через которую будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

LPDISPATCH GetPoint();

Синтаксис COM:

LPENTITY GetPoint();

Возвращаемое значение:

- указатель на интерфейс вершины ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс..Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_perpend_rebru`

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetEdge – Установить указатель на интерфейс ребра, перпендикулярно которому будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_perpend_rebru`

Синтаксис Automation:

BOOL SetEdge (LPDISPATCH edge);

Синтаксис COM:

BOOL SetEdge (LPENTITY edge);

Входной параметр:

edge - указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPoint – Установить указатель на интерфейс вершины, через которую будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_perpend_rebru`

Синтаксис Automation:

BOOL SetPoint (LPDISPATCH point);

Синтаксис COM:

BOOL SetPoint (LPENTITY point);

Входной параметр:

point - указатель на интерфейс вершины ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Плоскость через ребро параллельно/перпендикулярно другому ребру (Интерфейсы ksPlaneLineToEdgeDefinition, IPlaneLineToEdgeDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_parall_rebru

Интерфейс параметров плоскости, проходящей через ребро параллельно или перпендикулярно другому ребру.

ksPlaneLineToEdgeDefinition - интерфейс Automation
IPlaneLineToEdgeDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksPlaneLineToEdgeDefinition – свойства

parallel – Признак положения плоскости

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_parall_rebru

Тип данных: BOOL

Значения свойства:

TRUE - параллельно второму ребру,
FALSE - перпендикулярно второму ребру.

Синтаксис Automation:

parallel = iPlaneLineToEdge.parallel Получить свойство(*)

iPlaneLineToEdge.parallel = parallel		Установить свойство (*)
parallel	=	Получить свойство (**)
iPlaneLineToEdge.GetParallel()		
iPlaneLineToEdge.SetParallel(parallel)		Установить свойство (**)

ksPlaneLineToEdgeDefinition – методы

GetEdgeFirst – Получить указатель на интерфейс первого ребра, через которое будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_parall_rebru

Синтаксис Automation:

LPDISPATCH GetEdgeFirst();

Синтаксис COM:

LPENTITY GetEdgeFirst();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity.

GetEdgeSecond – Получить указатель на интерфейс второго ребра, параллельно или перпендикулярно которому будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

::/Postroenie_ploskostey.htm#ploskost_parall_rebru

Синтаксис Automation:

LPDISPATCH GetEdgeSecond();

Синтаксис COM:

LPENTITY GetEdgeSecond();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

::/Postroenie_ploskostey.htm#ploskost_parall_rebru

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetEdgeFirst – Установить указатель на интерфейс первого ребра, через которое будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_parall_rebru`

Синтаксис Automation:

BOOL SetEdgeFirst (LPDISPATCH edge1);

Синтаксис COM:

BOOL SetEdgeFirst (LPENTITY edge1);

Входной параметр:

edge1	- указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetEdgeSecond – Установить указатель на интерфейс второго ребра, параллельно или перпендикулярно которому будет проходить новая плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_parall_rebru`

Синтаксис Automation:

BOOL SetEdgeSecond (LPDISPATCH edge2);

Синтаксис COM:

BOOL SetEdgeSecond (LPENTITY edge2);

Входной параметр:

edge2	- указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Плоскость под углом к другой плоскости (Интерфейсы ksPlaneAngleDefinition, IPlaneAngleDefinition)

Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_angle`

Интерфейс параметров вспомогательной плоскости, построенной под углом к другой плоскости и проходящей через заданную ось или ребро.

ksPlaneAngleDefinition - интерфейс Automation
IPlaneAngleDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksPlaneAngleDefinition – свойства

angle – Угол наклона плоскости относительно базовой плоскости

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_angle`

Тип данных: double

Синтаксис Automation:

angle = iPlaneAngle.angle	Получить свойство(*)
iPlaneAngle.angle = angle	Установить свойство (*)
angle = iPlaneAngle.GetAngle()	Получить свойство (**)
iPlaneAngle.SetAngle(angle)	Установить свойство (**)

ksPlaneAngleDefinition – методы

GetAxis – Получить указатель на интерфейс базовой прямой (оси или ребра)

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_angle`

Синтаксис Automation:

LPDISPATCH GetAxis();

Синтаксис COM:

LPENTITY GetAxis();

Возвращаемое значение:

- указатель на интерфейс прямой ksEntity или IEntity.

GetPlane – Получить указатель на интерфейс базовой плоскости

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_angle`

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_angle`

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetAxis – Изменить указатель на интерфейс базовой прямой (оси или ребра)

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_angle

Синтаксис Automation:

BOOL SetAxis (LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis (LPENTITY axis);

Входной параметр:

axis	- указатель на интерфейс прямой ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetPlane – Изменить указатель на интерфейс базовой плоскости

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_angle

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane	- указатель на интерфейс плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Параллельная плоскость (Интерфейсы ksPlaneParallelDefinition, IPlaneParallelDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_tochky

Интерфейс параметров плоскости, проходящей через вершину параллельно другой плоскости.

ksPlaneParallelDefinition - интерфейс Automation
IPlaneParallelDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneParallelDefinition – методы

GetPlane – Получить указатель на интерфейс плоскости, параллельно которой будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_tochky

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

GetPoint – Получить указатель на интерфейс вершины, через которую будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_cherez_tochky

Синтаксис Automation:

LPDISPATCH GetPoint();

Синтаксис COM:

LPENTITY GetPoint();

Возвращаемое значение:

- указатель на интерфейс вершины ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_tochky`

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPENTITY GetSurface();

Возвращаемое значение:

- указатель на интерфейс ksSurface или ISurface.

SetPlane – Установить указатель на интерфейс плоскости, параллельно которой будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_tochky`

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPoint – Установить указатель на интерфейс вершины, через которую будет проходить новая плоскость

Интерфейс..Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_cherez_tochky`

Синтаксис Automation:

BOOL SetPoint (LPDISPATCH point);

Синтаксис COM:

BOOL SetPoint (LPENTITY point);

Входной параметр:

point - указатель на интерфейс вершины ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Касательная плоскость (Интерфейсы ksPlaneTangentToSurfaceDefinition, IPlaneTangentToSurfaceDefinition)

Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Интерфейс параметров касательной плоскости.

ksPlaneTangentToSurfaceDefinition - интерфейс Automation
IPlaneTangentToSurfaceDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

IPlaneTangentToSurfaceDefinition – свойства

angle – Угол между плоскостью и плоскостью, перпендикулярной опорному объекту

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Тип данных: double

Синтаксис Automation:

angle = iPlaneTangent.angle	Получить свойство(*)
iPlaneTangent.angle = angle	Установить свойство (*)
angle = iPlaneTangent.GetAngle()	Получить свойство (**)

iPlaneTangent.SetAngle (angle)	Установить свойство (**)

choosePlane – Изменение положения касательной плоскости относительно выбранной цилиндрической или конической грани

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_perpend_rebru`

Тип данных: long

Значения свойства:

1	- положение, предложенное по умолчанию,
0	- положение, противоположное предложенному по умолчанию.

Синтаксис Automation:

choosePlane = iPlaneTangentToSurface.choosePlane	Получить свойство(*)
iPlaneTangentToSurface.choosePlane = choosePlane	Установить свойство (*)
choosePlane = iPlaneTangentToSurface.GetChoosePlane()	Получить свойство (**)
iPlaneTangentToSurface.SetChoosePlane(choosePlane)	Установить свойство (**)

IPlaneTangentToSurfaceDefinition – методы

GetFace – Получить указатель на интерфейс цилиндрической или конической грани, к которой требуется построить касательную плоскость

Интерфейс...Справка системы КОМПАС...

`Postroenie_ploskostey.htm#ploskost_perpend_rebru`

Синтаксис Automation:

LPDISPATCH GetFace();

Синтаксис COM:

LPENTITY GetFace();

Возвращаемое значение:

- указатель на интерфейс грани ksEntity или IEntity.

GetPlane – Получить указатель на интерфейс плоского объекта, параллельно которому должна пройти новая плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

LPDISPATCH GetPlane();

Синтаксис COM:

LPENTITY GetPlane();

Возвращаемое значение:

- указатель на интерфейс плоской грани или плоскости ksEntity или IEntity.

GetSurface – Получить указатель на интерфейс математической поверхности

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

- указатель на интерфейс плоской грани или плоскости ksSurface или ISurface.

SetFace – Установить указатель на интерфейс цилиндрической или конической грани, к которой требуется построить касательную плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

BOOL SetFace (LPDISPATCH face);

Синтаксис COM:

BOOL SetFace (LPENTITY face);

Входной параметр:

plane - указатель на интерфейс грани ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetPlane – Установить указатель на интерфейс плоского объекта, параллельно которому должна пройти новая плоскость

Интерфейс...Справка системы КОМПАС...

Postroenie_ploskostey.htm#ploskost_perpend_rebru

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPENTITY plane);

Входной параметр:

plane - указатель на интерфейс плоской грани или плоскости ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Ось через ребро (Интерфейсы ksAxisEdgeDefinition, IAxisEdgeDefinition)

Справка системы КОМПАС...

Postroenie_osei.htm#osq_cherez_rebro

Интерфейс параметров вспомогательной оси, проходящей через прямолинейное ребро.

ksAxisEdgeDefinition - интерфейс Automation
IAxisEdgeDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksAxisEdgeDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...Справка системы КОМПАС...

::/Postroenie_osei.htm#osq_cherez_rebro

Синтаксис Automation:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- указатель на интерфейс ksCurve3D или iCurve3D.

GetEdge – Получить указатель на интерфейс базового ребра

Интерфейс..Справка системы КОМПАС...

::/Postroenie_osei.htm#osq_cherez_rebro

Синтаксис Automation:

LPDISPATCH GetEdge();

Синтаксис COM:

LPENTITY GetEdge();

Возвращаемое значение:

- указатель на интерфейс ребра ksEntity или IEntity.

SetEdge – Изменить указатель на интерфейс базового ребра

Интерфейс..Справка системы КОМПАС...

::/Postroenie_osei.htm#osq_cherez_rebro

Синтаксис Automation:

BOOL SetEdge (LPDISPATCH edge);

Синтаксис COM:

BOOL SetEdge (LPENTITY edge);

Входной параметр:

edge - указатель на интерфейс ребра ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Ось через две точки (Интерфейсы ksAxis2PointsDefinition,IAxis2PointsDefinition)

Справка системы КОМПАС...

Интерфейс параметров вспомогательной оси, проходящей через две точки.

ksAxis2PointsDefinition - интерфейс Automation
IAxis2PointsDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksAxis2PointsDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- указатель на интерфейс ksCurve3D или iCurve3D.

GetPoint – Получить указатель на интерфейс точки с указанным номером

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetPoint (long number);

Синтаксис COM:

LPENTITY GetPoint (long number);

Входной параметр:

number - номер точки (1 или 2).

Возвращаемое значение:

- указатель на интерфейс точки ksEntity или IEntity.

SetPoint – Изменить указатель на интерфейс точки с указанным номером

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetPoint (long number, LPDISPATCH point);

Синтаксис COM:

BOOL SetPoint (long number, LPENTITY point);

Входные параметры:

point - указатель на интерфейс точки ksEntity или IEntity,
number - номер точки (1 или 2).

Возвращаемое значение:

TRUE - в случае успешного завершения.

Ось на пересечении плоскостей (Интерфейсы ksAxis2PlanesDefinition, IAxis2PlanesDefinition)

Справка системы КОМПАС...

Интерфейс параметров вспомогательной оси на пересечении двух плоскостей.

ksAxis2PlanesDefinition - интерфейс Automation
IAxis2PlanesDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksAxis2PlanesDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
LPDISPATCH GetCurve3D();
Синтаксис COM:
LPCURVE3D GetCurve3D();
Возвращаемое значение:

- указатель на интерфейс ksCurve3D или iCurve3D.

GetPlane – Получить указатель на интерфейс базовой плоскости с указанным номером

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
LPDISPATCH GetPlane (long number);
Синтаксис COM:
LPENTITY GetPlane (long number);
Входной параметр:

number - номер плоскости (1 или 2).

Возвращаемое значение:

- указатель на интерфейс плоскости ksEntity или IEntity.

SetPlane – Изменить указатель на интерфейс базовой плоскости с указанным номером

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:
BOOL SetPlane(long number, LPDISPATCH plane);
Синтаксис COM:

BOOL SetPlane(long number, LPENTITY plane);

Входные параметры:

plane - указатель на интерфейс плоскости ksEntity или IEntity,
number - номер плоскости (1 или 2).

Возвращаемое значение:

TRUE - в случае успешного завершения.

Ось формообразующего элемента (Интерфейсы ksAxisOperationsDefinition, IAxisOperationsDefinition)

Справка системы КОМПАС...

Интерфейс параметров оси формообразующего элемента.

ksAxisOperationsDefinition - интерфейс Automation
IAxisOperationsDefinition - интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksAxisOperationsDefinition – методы

GetCurve3D – Получить указатель на интерфейс математической кривой

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

- указатель на интерфейс ksCurve3D или iCurve3D.

GetOperation – Получить базовый формообразующий элемент

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH GetOperation();

Синтаксис COM:

LPENTITY GetOperation();

Возвращаемое значение:

- указатель на интерфейс элемента ksEntity или IEntity.

SetOperation – Задать базовый формообразующий элемент

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

BOOL SetOperation (LPDISPATCH operation);

Синтаксис COM:

BOOL SetOperation (LPENTITY operation);

Входной параметр:

operation	- указатель на интерфейс базового формообразующего элемента ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Ось конической грани (Интерфейсы ksAxisConefaceDefinition, IAxisConefaceDefinition)

Справка системы КОМПАС...

Интерфейс параметров конструктивной оси конической грани.

ksAxisConefaceDefinition	- интерфейс Automation
IAxisConefaceDefinition	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetDefinition или IEntity::GetDefinition.

ksAxisConefaceDefinition – методы

GetCurve3D– Получить указатель на интерфейс математической кривой

Интерфейс..Справка системы КОМПАС...

Синтаксис Automation:
LPDISPATCH GetCurve3D();
Синтаксис COM:
LPCURVE3D GetCurve3D();
Возвращаемое значение:

- указатель на интерфейс ksCurve3D или iCurve3D.

GetFace – Получить указатель на базовую коническую поверхность

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:Postroenie_osei.htm#osq_konicheskoj_grani
LPDISPATCH GetFace();
Синтаксис COM:
LPENTITY GetFace();
Возвращаемое значение:

- указатель на интерфейс поверхности ksEntity или IEntity.

SetFace – Установить указатель на базовую коническую поверхность

Интерфейс..Справка системы КОМПАС...

Синтаксис Automation:
BOOL SetFace (LPDISPATCH face);
Синтаксис COM:
BOOL SetFace (LPENTITY face);
Входной параметр:

face - указатель на интерфейс поверхности ksEntity или IEntity.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Массив ориентированных ребер (Интерфейсы ksOrientedEdgeCollection, IOrientedEdgeCollection)

Интерфейс массива ориентированных ребер.

ksOrientedEdgeCollection - интерфейс Automation
IOrientedEdgeCollection - интерфейс COM

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- Интерфейс свойств ребра ksEdgeDefinition или IEdgeDefinition.
- Интерфейс цикла грани ksLoop или ILoop.

IOrientedEdgeCollection – методы

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPORIENTEDEDGE GetByIndex (long index);

Входной параметр:

index - номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

FindIt – Получить индекс элемента в массиве

Интерфейс...**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPORIENTEDEDGE entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity	- указатель на интерфейс компонента ksOrientedEdge или IOrientedEdge.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPORIENTEDEDGE First();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPORIENTEDEDGE Last();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPORIENTEDEDGE Next();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPORIENTEDEDGE Prev();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом ориентированных ребер и удаляются все предыдущие изменения в массиве.

Ориентированное ребро (Интерфейсы ksOrientedEdge, IOrientedEdge)

Интерфейс ориентированного ребра.

ksOrientedEdge - интерфейс Automation
IOrientedEdge - интерфейс COM

Описание:

Ориентированное ребро является объектом топологии. Содержит информацию о ребре и направлении относительно ребра.

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

Интерфейс массива ориентированных ребер ksOrientedEdgeCollection или IOrientedEdgeCollection.

IOrientedEdge – методы

GetAdjacentFace – Получить указатель на интерфейс грани, в цикл которой входит ребро

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetAdjacentFace (BOOL facePlus);

Синтаксис COM:

LPFACEDEFINITION GetAdjacentFace (BOOL facePlus);

Входные параметры:

facePlus - TRUE - ориентация грани относительно отрезка положительная,
- FALSE - ориентация грани относительно отрезка отрицательная.

Возвращаемое значение:

- указатель на интерфейс ksFaceDefinition или IFaceDefinition.

GetEdge – Получить указатель на интерфейс базового ребра

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetEdge();

Синтаксис COM:

LPEDGEDEFINITION GetEdge();

Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

GetNext – Получить указатель на интерфейс следующего ориентированного ребра

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetNext();

Синтаксис COM:

LPORIENTEDEDGE GetNext();

Возвращаемое значение:

- указатель на интерфейс ksOrientedEdge или IOrientedEdge.

GetOrientation – Получить направление относительно базового ребра

Интерфейс...**Синтаксис Automation**:

BOOL GetOrientation();

Синтаксис COM:

BOOL GetOrientation();

Возвращаемое значение:

TRUE	- направления совпадают,
FALSE	- направления не совпадают.

GetOwnerEntity – Получить указатель на интерфейс ребра

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetOwnerEntity();

Синтаксис COM:

LPENTITY GetOwnerEntity();

Возвращаемое значение:

- указатель на интерфейс ksEntity или IEntity.

GetSameSense – Получить направление относительно кривой

Интерфейс...**Синтаксис Automation**:

BOOL GetSameSense();

Синтаксис COM:

BOOL GetSameSense();

Возвращаемое значение:

TRUE	- направления совпадают,
FALSE	- направления не совпадают.

IsPole – Является ли ребро полюсным

Интерфейс...**Синтаксис**:

BOOL IsPole();

IsSeam – Является ли ребро швом

Интерфейс...**Синтаксис**:

BOOL IsSeam();

IsStraight – Является ли ребро прямым

Интерфейс...**Синтаксис**:

BOOL IsStraight();

Трехмерное тело (Интерфейсы ksBody, Body)

Интерфейс трехмерного тела.

ksBody	- интерфейс Automation
IBody	- интерфейс COM

Примечание:

1. Твердое тело состоит из массива граней.
2. Данный интерфейс может быть получен от интерфейса коллекции тел ksBodyCollection, IBodyCollection или от интерфейса компонента ksPart, IPart с помощью метода ksPart::GetMainBody, IPart::GetMainBody.

IBody – свойства

MultiBodyParts – Признак того, что компонент состоит из нескольких частей

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- компонент состоит из нескольких частей,
FALSE	- компонент не состоит из нескольких частей.

Синтаксис Automation:

MultiBodyParts = iBody.MultiBodyParts		Получить свойство (*)
MultiBodyParts iBody.GetMultiBodyParts()	=	Получить свойство (**)

Примечания:

Свойство позволяет узнать состоит ли компонент их нескольких частей.

В результате работы свойства можно получить компонент (деталь), состоящую из нескольких частей. С точки зрения конструктора, деталь должна представлять нечто целое. Частей быть не должно. Следовательно, нужно отыскать в компоненте операции, порождающие несколько частей, и выключить из расчета не нужные, чтобы образовалось целое тело.

IBody – методы

CheckIntersectionWithBody – Проверить наличия пересечений с другим телом

Интерфейс..**Синтаксис Automation**:

LPDISPATCH CheckIntersectionWithBody (LPDISPATCH otherBody,
BOOL checkTangent);

Синтаксис COM:

LPINTERSECTIONRESULT CheckIntersectionWithBody (LPBODY otherBody,
BOOL checkTangent);

Входные параметры:

otherBody	- тело, с которым проверяется пересечение,
checkTangent	- считать касания пересечениями.

Возвращаемое значение:

Интерфейс ksIntersectionResult	- в случае успеха,
NULL	- если пересечений нет.

CurveIntersection – Рассчитать пересечения с кривой

Интерфейс..**Automation**:

BOOL CurveIntersection (LPDISPATCH curve,
LPDISPATCH fases,
LPDISPATCH points);

Синтаксис COM:

BOOL CurveIntersection (LPCURVE3D curve,
LPFACECOLLECTION faces,
LPCOORDINATE3DCOLLECTION points);

Входные параметры:

curve - Указатель на интерфейс 3D кривой ksCurve3D.

Выходные параметры:

faces - Коллекция граней пересекаемых кривой ksFaceCollection,
points - Коллекция координат точек пересечений ksCoordinate3dCollection.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Координаты точек сортируются по параметру кривой t.
2. Параметры faces, points являются необязательными.
3. Заполняются только присланные коллекции.
4. Интерфейсы коллекций для заполнения нужно получить с помощью функций:

- ksFaceCollection, ksD ocument3D::GetInterface с параметром o3dType = o3d_faceCollection;
- ksCoordinate3dCollection, ksDocument3D::GetInterface с параметром o3dType = o3d_coordinate3dCollection.

FaceCollection – Получить указатель на интерфейс массива граней, образующих тело

Интерфейс..**Синтаксис Automation**:

LPDISPATCH FaceCollection();

Синтаксис COM:

LPFACECOLLECTION FaceCollection();

Возвращаемое значение:

- указатель на интерфейс ksFaceCollection или IFaceCollection.

GetFeature – Получить объект дерева, связанный с данным телом

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetFeature();

Синтаксис COM:

LPFEATURE GetFeature();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

GetGabarit – Получить габарит тела

Интерфейс...**Синтаксис Automation**:

BOOL GetGabarit (double *x1,
double *y1,
double *z1,
double * x2,
double *y2,
double *z2);

Синтаксис COM:

BOOL GetGabarit (double *x1,
double *y1,
double *z1,
double * x2,
double *y2,
double *z2);

Выходные параметры:

x1, y1, z1	- координаты первой вершины габаритного куба,
x2, y2, z2	- координаты второй вершины габаритного куба.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetIntersectionFacesWithBody – Определить взаимодействующие грани при пересечении данного тела с другим

Интерфейс..**Синтаксис Automation**:

BOOL GetIntersectionFacesWithBody (LPDISPATCH otherBody, VARIANT * intersectionFaces1,
VARIANT * intersectionFaces2,
VARIANT * connectedFaces1,
VARIANT * connectedFaces2);

Синтаксис COM:

BOOL GetIntersectionFacesWithBody (LPBODY otherBody, VARIANT * intersectionFaces1,
VARIANT * intersectionFaces2,
VARIANT * connectedFaces1,
VARIANT * connectedFaces2).

Входные параметры:

otherBody	- второе тело.

Выходные параметры:

intersectionFaces1	- пересекаемые грани первого тела,
intersectionFaces2	- пересекаемые грани второго тела,
connectedFaces1	- совпадающие грани первого тела,
connectedFaces2	- совпадающие грани второго тела.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Пересекаемые и совпадающие грани возвращаются в VARIANT-е так:

- Если грань одна, то VARIANT заполняется как VT_DISPATH в автоматизации и VT_UNKNOVN в COM.
- Если граней несколько, то VARIANT заполняется как массив SafeArray типа VT_ARRAY | VT_DISPATH в автоматизации и VT_ARRAY | VT_UNKNOVN в COM.

Количество граней в массивах intersectionFaces1 и intersectionFaces2 совпадают и образуют пары, т.е. грань с индексом i из массива intersectionFaces1 пересекается с гранью i из массива intersectionFaces2.

Одна и та же грань может встречаться в массиве несколько раз, если она пересекается с несколькими гранями.

Так же заполняются и массивы connectedFaces1 и connectedFaces2.

IsSolid – Является ли объект твердым телом или поверхностью

Интерфейс..**Синтаксис Automation**:

BOOL IsSolid();

Синтаксис COM:

BOOL IsSolid();

Возвращаемое значение:

TRUE	- объект является твердым телом,
FALSE	- объект является поверхностью.

Примечание:

Твердое тело может состоять из набора поверхностей.

MassInertiaParam – Получить указатель на интерфейс массово-центровочных характеристик

Интерфейс..**Синтаксис Automation**:

LPDISPATCH CalcMassInertiaProperties (long bitVector);

Синтаксис COM:

LPMASSINERTIAPARAM CalcMassInertiaProperties (unsigned int bitVector);

Входной параметр:

bitVector	- определяет размерность длины, размерность массы.

Возвращаемое значение:

- указатель на интерфейс ksIMassInertiaParam или IMassInertiaParam.

Примечания:

Варианты значений для задания значений bitVector находятся в интервале [ST_MIX_MM..ST_MIX_KG]

Пример:

(метры|кг - ST_MIX_MIST_MIX_KG).

Интерфейс частей тела (Интерфейсы ksBodyParts, IBodyParts)

Справка системы КОМПАС...

Интерфейс частей тела.

ksBodyParts	- интерфейс Automation
IBodyParts	- интерфейс COM

Описание:

Некоторые операции в 3D позволяют создавать несколько частей результирующего тела. Интерфейс позволяет получить информацию о частях тела и управлять их включением в расчет и выключением из расчета. Желательно включить части таким образом, чтобы тело оказалось целым.

Примечание:

Данный интерфейс можно получить, используя метод интерфейса элемента модели ksEntity::GetBodyParts или IEntity::GetBodyParts.

IBodyParts – свойства

AllSelected – Признак включения или выключения из расчета всех частей тела

Интерфейс..Тип данных:BOOL.

Значения свойства:

TRUE	- выбраны все части,
FALSE	- не все части выбраны.

Синтаксис Automation:

AllSelected = iPart.AllSelected	=	Получить свойство(*)
iPart.AllSelected = AllSelected	=	Установить свойство (*)
AllSelected = iPart.GetAllSelected()	=	Получить свойство (**)
iPart.SetAllSelected(AllSelected)		Установить свойство (**)

Примечание:

Если нужно включить или выключить все части из расчета или узнать, включены ли все части, то для ускорения используется это свойство. Для этой же цели можно анализировать или изменять каждую часть отдельно.

IBodyParts – методы

GetCount – Получить количество частей

Интерфейс..**Синтаксис Automation**:

long Count();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

количество частей	- в случае успешного завершения.

Примечание:

Позволяет получить количество частей тела для данной операции.

GetPartSelected – Получить для части тела с индексом признак включения

Интерфейс..**Синтаксис Automation**:

BOOL GetPartSelected(long Index);

Синтаксис COM:

BOOL GetPartSelected(long Index);

Входные параметры:

Index	- индекс выбираемой части.

Возвращаемое значение:

TRUE	- часть тела включена,
FALSE	- часть тела выключена.

Примечание:

Метод позволяет получить для части тела с индексом признак включения.

SetGreatPartsSelected – Включить часть тела по умолчанию (максимальную)

Интерфейс..**Синтаксис Automation**:

BOOL SetGreatPartsSelected();

Синтаксис COM:

BOOL SetGreatPartsSelected();

Входные параметры:

Index	- индекс выбираемой части.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет включить часть тела по умолчанию, как правило это большая часть.

SetPartSelected – Установить для части тела с индексом признак включения

Интерфейс..**Синтаксис Automation**:

BOOL SetPartSelected(long Index, BOOL Select);

Синтаксис COM:

BOOL SetPartSelected(long Index, BOOL Select);

Входные параметры:

Index	- индекс выбираемой части,
Select	- признак включения в расчет; TRUE - включить, FALSE - выключить часть из расчета.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет изменить признак включения части тела с индексом.

UserBodyPartsChoice – Запустить визуальный процесс выбора частей тела

Интерфейс...**Синтаксис Automation**:

BOOL UserBodyPartsChoice();

Синтаксис COM:

BOOL UserBodyPartsChoice();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Позволяет запустить визуальный процесс выбора частей тела.

Интерфейс массива трехмерных тел (Интерфейсы ksBodyCollection, IBodyCollection)

Интерфейс массива трехмерных тел.

ksBodyCollection	- интерфейс Automation
IBodyCollection	- интерфейс COM

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- Интерфейс булевой операции над твердыми телами ksAggregateDefinition или IAggregateDefinition.
- Интерфейс области применения для тел в операции ksChooseBodies или IChooseBodies.

- Интерфейс объекта Дерева построения ksFeature или IFeature.
- Интерфейс детали или подсборки в составе сборки ksPart или IPart.
- Интерфейс элемента модели (оси, плоскости, формообразующего элемента) ksEntityи-ли IEntity.

IBodyCollection – методы

Add – Добавить элемент в конец массива

Интерфейс..**Синтаксис Automation**:

BOOL Add(LPDISPATCH body);

Синтаксис COM:

BOOL Add(LPBODY body);

Входные параметры:

body	- указатель на интерфейс ksBody или IBody добавля-емого объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта body не должен быть равен 0.

AddAt – Добавить объект с заданным индексом в массив

Интерфейс..**Синтаксис Automation**:

BOOL AddAt(LPDISPATCH body, long index);

Синтаксис COM:

BOOL AddAt(LPBODY body, long index);

Входные параметры:

body	- указатель на интерфейс ksBody или IBody,
index	- индекс в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта body не должен быть равен 0.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс..**Синтаксис Automation**:

BOOL AddBefore(LPDISPATCH body, LPDISPATCH base);

Синтаксис COM:

BOOL AddBefore(LPBODY body, LPBODY base);

Входные параметры:

body	- указатель на интерфейс ksBody или IBody добавляемого объекта,
base	- указатель на интерфейс ksBody или IBody базового объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатели на интерфейсы добавляемого и базового объекта body и base не должны быть равны 0.

Clear – Очистить массив

Интерфейс..**Синтаксис Automation**:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объекты удаляются только из массива, в модель изменения не передаются.

DetachByBody – Удалить указанный объект из массива

Интерфейс..**Синтаксис Automation**:

BOOL DetachByBody(LPDISPATCH body);

Синтаксис COM:

BOOL DetachByBody(LPBODY body);

Входные параметры:

body	- указатель на интерфейс ksBody или IBody.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс удаляемого объекта body не должен быть равен 0.

DetachByIndex – Удалить объект из массива по индексу

Интерфейс..**Синтаксис Automation**:

BOOL DetachByIndex(long index);

Синтаксис COM:

BOOL DetachByIndex(long index);

Входные параметры:

index	- индекс удаляемого объекта в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Элемент массива из модели не удаляется.

FindIt – Получить индекс элемента в массиве

Интерфейс..**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPBODY entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity - указатель на интерфейс компонента ksBody или IBody.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс..**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPBODY First();

Возвращаемое значение:

- указатель на интерфейс ksBody или IBody.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPBODY GetByIndex (long index);

Входной параметр:

index - номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksBody или IBody.

GetCount – Получить количество элементов в массиве

Интерфейс..**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс..**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPBODY Last();

Возвращаемое значение:

- указатель на интерфейс ksBody или IBody.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс..**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPBODY Next();

Возвращаемое значение:

- указатель на интерфейс ksBody или IBody.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс..**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPBODY Prev();

Возвращаемое значение:

- указатель на интерфейс компонента ksBody или IBody.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом тел трехмерной детали или сборки и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс..**Синтаксис Automation**:

BOOL SetByIndex(LPDISPATCH body, long index);

Синтаксис COM:

BOOL SetByIndex(LPBODY body, long index);

Входные параметры:

body	- указатель на интерфейс ksBody или IBody присоединяемого объекта,
index	- индекс элемента в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейс параметров вершины ломаной Интерфейсы ksPolyLineVertexParam, IPolygonalLineVertexParam)

Справка системы КОМПАС...

CM_CCURVE_POLYLINE_BY_VERTEX.htm

Интерфейс параметров вершины ломаной.

ksPolyLineVertexParam	- интерфейс Automation
IPolygonalLineVertexParam	- интерфейс COM

IPolygonalLineVertexParam – свойства

buildingType – Способ построения сегмента ломаной

Интерфейс...

Тип данных: ksLineBuildingType

Синтаксис Automation:

buildingType	=	Получить
iObject.buildingType		свойство(*)
iObject.buildingType	=	Установить
buildingType		свойство (*)
buildingType	=	Получить
iObject.GetbuildingType()		свойство (**)
iObject.SetbuildingType		Установить
(buildingType)		свойство (**)

IPolygonalLineVertexParam – методы

GetAssociation – Получить ассоциированную вершину

Интерфейс...

Синтаксис Automation:

ksEntity * GetAssociation();

Синтаксис COM:

LPENTITY GetAssociation();

Возвращаемое значение:

Указатель на интерфейс объекта ksEntity или IEntity	- если объект задан,
NULL	- если объект не задан.

Примечание:

Тип возвращаемого объекта вершина o3d_vertex.

GetBuildingObject – Получить объект относительно которого ведется построение

Интерфейс...

Синтаксис Automation:

ksEntity * GetBuildingObject();

Синтаксис COM:

LPENTITY GetBuildingObject();

Возвращаемое значение:

Указатель на интерфейс объекта ksEntity или IEntity	- если объект задан,
NULL	- если объект не задан.

Примечание:

Задать объект, относительно которого ведется построение, можно для типа построения ksLineBuildingType, равного ksLBTParallel и ksLBTPerpendicular.

GetIndex – Получить индекс вершины

Интерфейс...

Синтаксис:

int GetIndex();

Возвращаемое значение:

Индекс вершины.	- если объект задан.

GetParamByDistance – Получить расстояние и радиус

Интерфейс...

Синтаксис:

BOOL GetParamByDistance (double * distance, double * radius);

Входные параметры:

distance	- расстояние до предыдущей вершины,
radius	- радиус.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Для вершины с индексом 0 расстояние выдается до вершины с индексом 1. Для остальных вершин расстояние выдается до вершины с индексом на 1 меньше, чем у данной вершины.
2. Получить расстояние можно для любого типа построения.
3. Установить расстояние можно, если тип построения ksLineBuildingType не равен ksLBTByPoint.

GetParamVertex – Получить параметры вершины

Интерфейс...

Синтаксис:

BOOL GetParamVertex (double *x, double *y, double *z, double *radius);

Входные параметры:

x	- x координата вершины,
y	- y координата вершины,
z	- z координата вершины,
radius	- радиус.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetVertex – Получить вершину

Интерфейс...

Синтаксис Automation:

ksEntity * GetVertex();

Синтаксис COM:

LPENTITY GetVertex();

Возвращаемое значение:

Указатель на интерфейс объекта ksEntity или IEntity	- если объект задан,
NULL	- если объект не задан.

Примечание:

Тип возвращаемого объекта - вершина o3d_vertex.

SetAssociation – Установить ассоциированную вершину

Интерфейс...

Синтаксис Automation:

BOOL SetAssociation (ksEntity * vertex);

Синтаксис COM:

BOOL SetAssociation(LPENTITY vertex);

Входные параметры:

vertex	- указатель на интерфейс объекта ksEntity или IEntity, если нужно установить ассоциацию,
NULL	- если нужно снять ассоциацию.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объект должен быть вершиной o3d_vertex.

SetBuildingObject – Установить объект относительно которого ведется построение

Интерфейс...

Синтаксис Automation:

BOOL SetBuildingObject (ksEntity * object);

Синтаксис COM:

BOOL SetBuildingObject (LPENTITY object);

Входные параметры:

object - указатель на интерфейс объекта ksEntity или IEntity

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Задать объект, относительно которого ведется построение, можно для типа построения ksLineBuildingType, равного ksLBTParallel и ksLBTPerpendicular.

SetParamByDistance – Установить расстояние и радиус

Интерфейс...

Синтаксис:

BOOL SetParamByDistance (double distance, double radius);

Входные параметры:

distance	- расстояние до предыдущей вершины,
radius	- радиус.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Установить расстояние можно, если тип построения ksLineBuildingType не равен ksLBTByPoint.
2. Нельзя установить расстояние для вершины с индексом 0.

SetParamByVertex – Установить параметры вершины по парамерам вершины другого объекта

Интерфейс...

Синтаксис Automation:

BOOL SetParamByVertex (ksEntity * vertex, double radius);

Синтаксис COM:

BOOL SetParamByVertex (LPENTITY vertex, double radius);

Входные параметры:

vertex	- указатель на интерфейс параметров вершины ksEntity или IEntity,
radius	- радиус.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetParamVertex – Установить параметры вершины

Интерфейс...

Синтаксис:

BOOL SetParamVertex (double x, double y, double z, double radius);

Входные параметры:

x	- x координата вершины,
y	- y координата вершины,
z	- z координата вершины,
radius	- радиус.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Интерфейсы параметров умолчательного объекта (IDefaultObject, ksDefaultObject)

Интерфейс параметров умолчательного объекта.

Иерархия:

IDispatch

IDefaultObject

Примечание:

1. Интерфейс можно получить для умолчательных объектов документа - базовых плоскостей и базовых осей.
2. Интерфейс возвращается функцией IEntity::GetDefinition и ksEntity::GetDefinition.

ksDefaultObject –методы

GetCurve3D – Получить интерфейс математической кривой для базовых осей

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetCurve3D();

Синтаксис COM:

LPCURVE3D GetCurve3D();

Возвращаемое значение:

Интерфейс математической кривой ksCurve3D или ICurve3D.

GetSurface – Получить интерфейс математической поверхности для базовой плоскости

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSurface();

Синтаксис COM:

LPSURFACE GetSurface();

Возвращаемое значение:

Интерфейс математической поверхности ksSurface или ISurface.

Математическая поверхность в трехмерном пространстве

Интерфейсы ksSurface, ISurface

Интерфейс математической поверхности в трехмерном пространстве.

ksSurface - интерфейс Automation
ISurface - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов ksPlaneAngleDefinition::GetSurface, ksPlaneMiddleDefinition::GetSurface, ksPlaneLineToPlaneDefinition::GetSurface, ksPlaneNormalToSurfaceDefinition::GetSurface, ksPlane3PointsDefinition::GetSurface ksPlanePerpendicularDefinition::GetSurface, ksPlaneTangentToSurfaceDefinition::GetSurface, ksSketchDefinition::GetSurface, ksPlaneEdgePointDefinition::GetSurface, ksPlaneLineToEdgeDefinition::GetSurface, ksPlaneParallelDefinition::GetSurface, ksPlaneOffsetDefinition::GetSurface, IDefaultObject::GetSurface, ksFaceDefinition::GetSurface.

ISurface – свойства

BoundaryCount – Количество контуров, задающих границу поверхности

Интерфейс...Тип данных: long

Синтаксис Automation:

BoundaryCount = Object.BoundaryCount		Получить свойство (*)
BoundaryCount Object.GetBoundaryCount()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

ISurface – методы

CurveIntersection – Рассчитать пересечения с кривой

Интерфейс...**Синтаксис Automation**:

BOOL CurveIntersection(LPDISPATCH curve, LPDISPATCH points, BOOL extSurf, BOOL extCurve);

Синтаксис COM:

BOOL CurveIntersection(LPCURVE3D curve, LPCOORDINATE3DCOLLECTION points, BOOL extSurf, BOOL extCurve);

Входные параметры:

curve	- указатель на интерфейс 3D кривой ksCurve3D,
extSurf	- TRUE - учитывать продолжение поверхности,
	- FALSE - не учитывать продолжение поверхности,
extCurve	- TRUE - учитывать продолжение кривой,
	- FALSE - не учитывать продолжение кривой.

Выходные параметры:

points - Коллекция координат точек пересечений ksCoordinate3dCollection.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание:

1. Координаты точек сортируются по параметру кривой t.
2. Параметр points является необязательным.
3. Заполняются только присланные коллекции.
4. Интерфейсы коллекций координат точек ksCoordinate3dCollection нужно получить с помощью функций ksDocument3D::GetInterface с параметром o3dType = o3d_coordinate3dCollection.

GetArea – Получить площадь поверхности

Интерфейс...**Синтаксис Automation**:

double GetArea (long bitVector);

Синтаксис COM:

double GetArea (unsigned long bitVector);

Входные параметры:

bitVector - единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

площадь поверхности - в случае успешного завершения,
0 - в случае неудачи.

GetBoundaryUVNurbs – Получить параметры NURBS-представления границы поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetBoundaryUVNurbs(BOOL uv,
BOOL unclamped,
long loopIndex,
long * degree,
long edgeIndex,
VARIANT * points,
VARIANT * weights,

VARIANT * knots,
double * tMin,
double * tMax);

Синтаксис COM:

BOOL GetBoundaryUVNurbs(BOOL uv,
BOOL unclamped,
long loopIndex,
long edgeIndex,
long * degree,
VARIANT * points,
VARIANT * weights,
VARIANT * knots,
double * tMin,
double * tMax);

Входные параметры:

uv	- TRUE - получить параметры границы поверхности в UV-параметрическом представлении исходной поверхности, - FALSE - получить параметры границы поверхности в 3D координатах,
unclamped	- TRUE - сомкнуть, если граница разомкнутая - FALSE - разомкнуть, если граница замкнутая,
loopIndex	- индекс цикла,
edgeIndex	индекс ребра в цикле (совпадает с индексом ребра в коллекции ориентированных ребер).

Выходные параметры:

degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
points	- массив параметров UV или координат вершин - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
weights	- веса точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
knots	- узлы точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
tMin, tMax	- минимальный и максимальный параметры.

Примечание:

Если признак UV равен TRUE, то в массиве points возвращаются параметры UV для Nurbs-представления границы. Параметры в массиве лежат в следующей последовательности:

u0, v0, u1, v1, ... ui, vi.

Если признак UV равен FALSE, то в массиве points возвращаются координаты вершин Nurbs-представления границы. Координаты точек в полученном массиве лежат в следующей последовательности:

x0, y0, z0, x1, y1, z1, ...xi, yi, zi.

- Если индекс ребра равен -1, формируется Nurbs-представление контура границы.
- Если индекс >= 0, то формируется Nurbs-представление ребра, входящего в цикл, параметр unclamped при этом игнорируется.

GetDerivativeU – Получить первую производную по U

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeU (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeU (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- первая производная по U.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeV– Получить первую производную по V

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- первая производная по V.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeUU – Получить вторую производную по UU

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeUU (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeUU (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- вторая производная по UU.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeVV – Получить вторую производную по VV

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeVV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeVV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- вторая производная по VV.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeUV – Получить вторую производную по UV

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeUV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeUV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- вторая производная по UV.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeUUU – Получить третью производную по UUU

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeUUU (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeUUU (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- третья производная по UUU.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeVVV – Получить третью производную по VVV

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeVVV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeVVV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- третья производная по VVV.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeUVV – Получить третью производную по UVV

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeUVV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeUVV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- третья производная по UVV.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetDerivativeUUV – Получить третью производную по UUV

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeUUV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeUUV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- третья производная по UUV.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetEdgesCount – Количество ребер в цикле

Интерфейс...Количество ребер в цикле.

Синтаксис Automation:

long GetEdgesCount(long loopIndex);

Синтаксис COM:

long GetEdgesCount(long loopIndex);

GetGabarit – Получить габаритный параллелепипед поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetGabarit (double * x1, double *y1, double *z1, double * x2, double *y2, double *z2);

Синтаксис COM:

BOOL GetGabarit (double * x1, double *y1, double *z1, double * x2, double *y2, double *z2);

Выходные параметры:

x1, y1, z1, x2, y2, z2	- координаты вершин габаритного параллелепипеда.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Координаты вершин параллелепипеда возвращаются в системе координат компонента.

GetNormal – Получить направление нормали

Интерфейс...**Синтаксис Automation**:

BOOL GetNormal (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetNormal (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- вектор нормали.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetNurbsSurfaceParam – Получить NURBS-представление поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetNurbsSurfaceParam();

Синтаксис COM:

LPNURBS3DPARAM GetNurbsSurfaceParam();

Возвращаемое значение:

- указатель на интерфейс ksNurbsSurfaceParam или INurbsSurfaceParam.

Примечание:

Метод позволяет получить параметры поверхности в NURBS-представлении. Для получения параметров поверхности в NURBS-представлении создается математическая Nurbs-поверхность, аналогичная заданной. Если заданная кривая является NURBS-поверхности, то возвращаются ее параметры без создания NURBS-представления.

GetParamUMin – Получить начальное значение параметра U в параметрическом представлении поверхности

Интерфейс...**Синтаксис Automation**:

double GetParamUMin();

Синтаксис COM:

double GetParamUMin();

Возвращаемое значение:

Значение параметра U.	- в случае успеха,
0	- в случае неудачи.

GetParamUMax – Получить конечное значение параметра U в параметрическом представлении поверхности

Интерфейс...**Синтаксис Automation**:

double GetParamUMax();

Синтаксис COM:

double GetParamUMax();

Возвращаемое значение:

Значение параметра U	- в случае успеха,
0	- в случае неудачи.

GetParamVMin – Получить начальное значение параметра V в параметрическом представлении поверхности

Интерфейс...**Синтаксис Automation**:

double GetParamVMin();

Синтаксис COM:

double GetParamVMin();

Возвращаемое значение:

Значение параметра V	- в случае успеха,
0	- в случае неудачи.

GetParamVMax – Получить конечное значение параметра V в параметрическом представлении поверхности

Интерфейс...**Синтаксис Automation**:

double GetParamVMax();

Синтаксис COM:

double GetParamVMax();

Возвращаемое значение:

Значение параметра V	- в случае успеха,
0	- в случае неудачи.

GetPoint – Получить координаты точки на поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetPoint (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- координаты точки на поверхности.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetSurfaceParam – Получить параметры плоскости, конуса, цилиндра, тора, сферы, NURBS или NULL

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetSurfaceParam();

Синтаксис COM:

LPDISPATCH GetSurfaceParam();

Возвращаемое значение:

- указатель на интерфейс параметра поверхности.

Примечание:

В соответствии с типом объекта метод вернет указатель на интерфейс параметра:

- плоскости	ksPlaneParam или IPlaneParam,
- конуса	ksConeParam или IConeParam,
- цилиндра	ksCylinderParam или ICylinderParam,
- тора	ksTorusParam или ITorusParam,
- сферы	ksSphereParam или ISphereParam,

- NURBS-поверхности	ksNurbsSurfaceParam или INurbsSurfaceParam.

Зная тип объекта, можно базовый интерфейс привести к конечному (IDispatch или IUnknown). Для определения типа объекта можно воспользоваться методами IsPlane, IsCone, IsCylinder, IsTorus, IsSphere, IsNurbsSurface.

GetTangentVectorU – Получить направление касательного вектора по U

Интерфейс...**Синтаксис Automation**:

BOOL GetTangentVectorU (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetTangentVectorU (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV	- значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z	- касательный вектор по U.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

GetTangentVectorV – Получить направление касательного вектора по V

Интерфейс...**Синтаксис Automation**:

BOOL GetTangentVectorV (double paramU, double paramV, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetTangentVectorV (double paramU, double paramV, double * x, double *y, double *z);

Входные параметры:

paramU, paramV - значения параметров U и V в параметрическом представлении поверхности.

Выходные параметры:

x, y, z - касательный вектор по V.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Граничные значения параметров U и V получаются при помощи методов GetParamUMin и GetParamUMax и GetParamVMin и GetParamVMax. Координаты точки возвращаются в системе координат компонента.

IsClosedU – Определить, является ли поверхность замкнутой по U

Интерфейс...**Синтаксис Automation**:

BOOL IsClosedU();

Синтаксис COM:

BOOL IsClosedU();

Возвращаемое значение:

TRUE - кривая замкнута,
FALSE - кривая разомкнута.

IsClosedV – Определить, является ли поверхность замкнутой по V

Интерфейс...**Синтаксис Automation**:

BOOL IsClosedV();

Синтаксис COM:

BOOL IsClosedV();

Возвращаемое значение:

TRUE - кривая замкнута,
FALSE - кривая разомкнута.

IsCone – Определить, является ли поверхность конической

Интерфейс...**Синтаксис Automation**:

BOOL IsCone();

Синтаксис COM:

BOOL IsCone();

Возвращаемое значение:

TRUE	- поверхность является конической,
FALSE	- поверхность не является конической.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetSurfaceParam вернет указатель на интерфейс параметра конуса ksConeParam или IConeParam.

IsCylinder – Определить, является ли поверхность цилиндрической

Интерфейс...**Синтаксис Automation**:

BOOL IsCylinder();

Синтаксис COM:

BOOL IsCylinder();

Возвращаемое значение:

TRUE	- поверхность является цилиндрической,
FALSE	- поверхность не является цилиндрической.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetSurfaceParam вернет указатель на интерфейс параметра цилиндра ksCylinderParam или IPlaneCylinder.

IsNurbsSurface – Определить, является ли поверхность NURBS

Интерфейс...**Синтаксис Automation**:

BOOL IsNurbsSurface();

Синтаксис COM:

BOOL IsNurbsSurface();

Возвращаемое значение:

TRUE	- поверхность является NURBS,

FALSE - поверхность не является NURBS.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра NURBS ksNurbsSurfaceParam или INurbsSurfaceParam.

IsPlane – Определить, является ли грань плоской

Интерфейс...**Синтаксис Automation**:

BOOL IsPlane();

Синтаксис COM:

BOOL IsPlane();

Возвращаемое значение:

TRUE - поверхность является плоской,
FALSE - поверхность не является плоской.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetSurfaceParam вернет указатель на интерфейс параметра плоскости ksPlaneParam или IPlaneParam.

IsRevolved – Определить, является ли поверхность поверхностью вращения

Интерфейс...**Синтаксис Automation**:

BOOL IsRevolved();

Синтаксис COM:

BOOL IsRevolved();

Возвращаемое значение:

TRUE - поверхность является поверхностью вращения,
FALSE - поверхность не является поверхностью вращения.

IsSphere – Определить, является ли поверхность сферой

Интерфейс...**Синтаксис Automation**:

BOOL IsSphere();

Синтаксис COM:

BOOL IsSphere();

Возвращаемое значение:

TRUE	- поверхность является сферической,
FALSE	- поверхность не является сферической.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetSurfaceParam вернет указатель на интерфейс параметра сферы ksSphereParam или ISphereParam.

IsSwept – Определить, является ли поверхность поверхностью по траектории

Интерфейс...**Синтаксис Automation**:

BOOL IsSwept();

Синтаксис COM:

BOOL IsSwept();

Возвращаемое значение:

TRUE	- поверхность является поверхностью по траектории,
FALSE	- поверхность не является поверхностью по траектории.

IsTorus – Определить, является ли поверхность тором

Интерфейс...**Синтаксис Automation**:

BOOL IsTorus();

Синтаксис COM:

BOOL IsTorus();

Возвращаемое значение:

TRUE	- поверхность является тороидальной,
FALSE	- поверхность не является тороидальной.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetSurfaceParam вернет указатель на интерфейс параметра тора ksTorusParam или ITorusParam.

NearPointProjection – Получить ближайшую проекцию точки на поверхность

Интерфейс...**Синтаксис Automation**:

BOOL NearPointProjection (double x, double y, double z, double * u, double * v, BOOL ext);

Синтаксис COM:

BOOL NearPointProjection (double x, double y, double z, double * u, double * v, BOOL ext);

Входные параметры:

x, y, z	- координаты исходной пространственной точки,
ext	- если проекция точки вне поверхности: - TRUE проецировать на продолжение поверхности, - FALSE найти ближайшую граничную точку поверхности. Если проекция точки внутри поверхности, значение этого параметра не учитывается.

Выходные параметры:

u, v	- параметрические координаты точки на поверхности.

Возвращаемое значение:

TRUE	- если проекция точки попала на саму поверхность или если ext равен TRUE,
FALSE	- если проекция точки не попала на саму поверхность.

Tessellation – Получить указатель на интерфейс триангуляции

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetTessellation();

Синтаксис COM:

LPTESSELLATION GetTessellation();

Возвращаемое значение:

- Указатель на интерфейс ksTessellation или ITessellation.

Триангуляция (аппроксимирующая поверхность грани), Интерфейсы ksTessellation, ITessellation

Интерфейс триангуляции (аппроксимирующей поверхности грани).

ksTessellation - интерфейс Automation
ITessellation - интерфейс COM

Описание:

Существуют два вида триангуляции:

- обычная, которая рассчитывается для отображения;
- для производства прочностных расчетов.

По умолчанию выдается обычная триангуляция - рассчитанная аппроксимирующая поверхность для отображения. Если необходима триангуляция для расчетов, то следует указать ограничение размера ребра триангуляционной пластины.

1. Триангуляционная сетка может быть построена с учетом нескольких параметров:

- Установить ограничение размера ребра для триангуляциооной пластины ksTessellation::SetFacetSize.
- Установить ограничение прогиба поверхности триангуляциооной пластины ksTessellation::SetFacetSag.
- Установить ограничение углового отклонения поверхности ksTessellation::SetFacetAngle.

Если ни один из параметров не задан (равны 0), возвращаются параметры рассчитанной в модели сетки.

2. В Компас при расчете триангуляции ограничение размера ребра рассчитывается в зависимости от габаритов модели.

Получить значение рассчитанной величины ограничения размера ребра можно с помощью функции IPart7::GetMaxSag.

Для расчетов рекомендуемое значение шага ребра составляет до 2% от максимального значения габарита. Для визуализации должно быть задано расстояние от аппроксимирующей хорды до кривой. Это значение должно составлять 0,04% от максимального габарита детали.

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

- Интерфейс свойств грани ksFaceDefinition или ksFaceDefinition.
- Интерфейс математической поверхности в трехмерном пространстве ksSurface или ISurface.

ITessellation – методы

GetFacet – Получить интерфейс триангуляционной пластины

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetFacet();

Синтаксис COM:

LPFACET GetFacet();

Возвращаемое значение:

- указатель на интерфейс ksFacet или IFacet.

GetFacetAngle– Получить ограничение углового отклонения поверхности

Интерфейс...**Синтаксис Automation**:

double GetFacetAngle();

Синтаксис COM:

double GetFacetAngle();

GetFacetsCount – Получить количество триангуляционных пластин

Интерфейс...**Синтаксис Automation**:

long GetFacetsCount();

Синтаксис COM:

long GetFacetsCount();

Возвращаемое значение:

Количество триангуляционных пластин	- в случае успеха,
-1	- в случае неудачи.

GetFacetData – Получить данные триангуляционной пластины по индексу в массиве триангуляционных пластин

Интерфейс...**Синтаксис Automation**:

BOOL GetFacetData (long index, LPDISPATCH facet);

Синтаксис COM:

BOOL GetFacetData (int index, LPFACET facet);

Входные параметры:

index	- индекс триангуляционной пластины в массиве,
facet	- указатель на интерфейс ksFacet или IFacet.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс триангуляционной пластины не должен быть равен 0.

GetFacetNormals – Получить параметры нормалей вершин триангуляционной сетки

Интерфейс...**Синтаксис Automation**:

BOOL GetFacetNormals (VARIANT * normals);

Синтаксис COM:

BOOL GetFacetNormals (VARIANT * normals);

Выходные параметры:

normals	- массив SafeArray вещественных чисел - нормалей вершин триангуляционной сетки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. В массиве нормалей вершин триангуляционной сетки координаты нормалей лежат в последовательности x, y, z нормаль для первой точки, x, y, z нормаль второй точки и т.д.
2. Функция используется совместно с GetFacetPoints.

GetFacetPoints – Получить параметры вершин триангуляционной сетки

Интерфейс...**Синтаксис Automation**:

BOOL GetFacetPoints (VARIANT * points, VARIANT * indexes);

Синтаксис COM:

BOOL GetFacetPoints (VARIANT * points, VARIANT * indexes);

Выходные параметры:

points	- массив SafeArray вещественных чисел - координат вершин триангуляционной сетки,
indexes	- массив SafeArray целых чисел - индексов вершин триангуляционной сетки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. В массиве индексов вершин триангуляционной сетки хранятся индексы точек из массива координат точек points. Для получения параметров вершины триангуляционной сетки нужно получить индекс точки из массива indexes.

2. Координаты точек в массиве points лежат в последовательности x, y, z первой точки, x, y, z второй точки и т.д.

GetFacetSize – Получить ограничение размера ребра для триангуляционной пластины

Интерфейс...**Синтаксис Automation**:

double GetFacetSize();

Синтаксис COM:

double GetFacetSize();

Возвращаемое значение:

- ограничение размера ребра для триангуляционной пластины.

GetFacetParams – Получить параметрические координаты вершин триангуляционных сетки

Интерфейс...**Синтаксис Automation**:

BOOL GetFacetParams(VARIANT * params);

Синтаксис COM:

BOOL GetFacetParams(VARIANT * params);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Выходной параметр:

params	- SafeArray - массив вещественных чисел - параметрических координат вершин триангуляционных сетки. Для формирование параметрических координат триангуляционной сетки требуется установить признак ksTessellation::SetNeedParams.

Примечание:

В массиве параметрические координаты вершин лежат в последовательности:

u, v для первой точки, u, v второй точки и т.д.

GetFacetSag – Получить ограничение прогиба поверхности триангуляционной пластины

Интерфейс...**Синтаксис Automation**:

double GetFacetSag();

Синтаксис COM:

double GetFacetSag();

GetNeedParams – Получить признак необходимости заполнения параметров вершин

Интерфейс...**Синтаксис Automation**:

BOOL GetNeedParams();

Синтаксис COM:

BOOL GetNeedParams();

GetNormal – Получить координаты нормали триангуляционной сетки

Интерфейс...**Синтаксис Automation**:

BOOL GetNormal (long index, double * x, double * y, double * z);

Синтаксис COM:

BOOL GetNormal (int index, float * x, float * y, float * z);

Входные параметры:

index	- индекс вершины в пластине.

Выходные параметры:

x, y, z	- координаты нормали.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetPoint – Получить координаты вершины триангуляционной сетки

Интерфейс...**Синтаксис Automation**:

BOOL GetPoint (long index, double * x, double * y, double * z);

Синтаксис COM:

BOOL GetPoint (int index, float * x, float * y, float * z);

Входные параметры:

index	- индекс вершины в массиве вершин.

Выходные параметры:

x, y, z	- координаты вершины.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetPointsCount – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

long GetPointsCount();

Синтаксис COM:

long GetPointsCount();

Возвращаемое значение:

Количество вершин триангуляционной сетки	- в случае успеха,
-1	- в случае неудачи.

Refresh – Обновить триангуляцию если изменилась геометрия

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

SetFacetAngle – Задать ограничение углового отклонения триангуляционной пластины

Интерфейс...**Синтаксис Automation**:

BOOL SetFacetAngle(double angle);

Синтаксис COM:

BOOL SetFacetAngle(double angle);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Выходной параметр:

angle	- величина углового отклонения.

Примечание:

1. Триангуляционная сетка может быть построена с учетом нескольких параметров:

- Установить ограничение размера ребра для триангуляциооной пластины ksTessellation::SetFacetSize.
- Установить ограничение прогиба поверхности триангуляциооной пластины ksTessellation::SetFacetSag.
- Установить ограничение углового отклонения поверхности ksTessellation::SetFacetAngle.

Если ни один из параметров не задан (равны 0), возвращаются параметры рассчитанной в модели сетки.

2. В Компас при расчете триангуляции ограничение размера ребра рассчитывается в зависимости от габаритов модели.

Получить значение расчитанной величины ограничения размера ребра можно с помощью метода IPart7::GetMaxSag.

SetFacetSag – Задать ограничение прогиба поверхности триангуляционной пластины (Если 0, обычная триангуляция)

Интерфейс...**Синтаксис Automation**:

BOOL SetFacetSag(double sag);

Синтаксис COM:

BOOL SetFacetSag(double sag);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Выходной параметр:

sag	- величина максимально допустимого прогиба на расстоянии шага.

Примечание:

1. Триангуляционная сетка может быть построена с учетом нескольких параметров:

- Установить ограничение размера ребра для триангуляционной пластины ksTessellation::SetFacetSize.
- Установить ограничение прогиба поверхности триангуляционной пластины ksTessellation::SetFacetSag.
- Установить ограничение углового отклонения поверхности ksTessellation::SetFacetAngle.

   Если ни один из параметров не задан (равны 0), возвращаются параметры рассчитанной в модели сетки.

2. В Компас при расчете триангуляции ограничение размера ребра рассчитывается в зависимости от габаритов модели.

   Получить значение рассчитанной величины ограничения размера ребра можно с помощью метода IPart7::GetMaxSag.

SetFacetSize – Изменить ограничение размера ребра для триангуляционной пластины

Интерфейс...**Синтаксис Automation**:

BOOL SetFacetSize (double sag);

Синтаксис COM:

BOOL SetFacetSize (double sag);

Входные параметры:

sag	0	- обычная триангуляция (для отображения),
sag	>0	- триангуляция для прочностных расчетов.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Триангуляционная сетка может быть построена с учетом нескольких параметров:

- Установить ограничение размера ребра для триангуляционной пластины ksTessellation::SetFacetSize.
- Установить ограничение прогиба поверхности триангуляционной пластины ksTessellation::SetFacetSag.
- Установить ограничение углового отклонения поверхности ksTessellation::SetFacetAngle.

   Если ни один из параметров не задан (равны 0), возвращаются параметры рассчитанной в модели сетки.

2. В Компас при расчете триангуляции ограничение размера ребра рассчитывается в зависимости от габаритов модели.

Получить значение рассчитанной величины ограничения размера ребра можно с помощью метода IPart7::GetMaxSag.

3. Рекомендуемое значение шага ребра для расчетов - до 2% от максимального значения габарита.

SetNeedParams – Задать необходимость заполнения параметров вершин

Интерфейс...**Синтаксис Automation**:

BOOL SetNeedParams(BOOL need);

Синтаксис COM:

BOOL SetNeedParams(BOOL need);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Триангуляционная пластина (Интерфейсы ksFacet, IFacet)

Интерфейс триангуляционной пластины.

ksFacet	- интерфейс Automation
IFacet	- интерфейс COM

Описание::

Триангуляционная пластина это треугольная или четырехугольная пластина, из которых состоит аппроксимирующая поверхность.

Примечание:

Данный интерфейс можно получить от следующих интерфейсов:

Интерфейс триангуляции (аппроксимирующей поверхности грани) ksTessellation или ITessellation.

IFacet – методы

GetPoint – Получить координаты вершины

Интерфейс...**Синтаксис Automation**:

BOOL GetPoint (long index, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint (int index, float * x, float *y, float *z);

Входной параметр:

index	- индекс вершины в пластине.

Выходные параметры:

x, y, z	- координаты вершины.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetPointsCount – Получить количество вершин пластины

Интерфейс...**Синтаксис Automation**:

long GetPointsCount();

Синтаксис COM:

int GetPointsCount();

Возвращаемое значение:

количество вершин пластины	- в случае успеха,
0	- в случае неудачи.

GetNormal – Получить координаты нормали

Интерфейс...**Синтаксис Automation**:

BOOL GetNormal (long index, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetNormal (int index, float * x, float *y, float *z);

Входной параметр:

index	- индекс вершины в пластине.

Выходные параметры:

x, y, z	- координаты вершины.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetTessellationIndex – Получить индекс вершины в массиве вершин триангуляционной сетки

Интерфейс...**Синтаксис Automation**:

long GetTessellationIndex (long index);

Синтаксис COM:

int GetTessellationIndex (int index);

Входной параметр:

index	- индекс вершины в пластине.

Возвращаемое значение:

Индекс вершины в массиве вершин триангуляционной сетки	- в случае успеха,
-1	- в случае неудачи.

Параметры сферы (Интерфейсы ksSphereParam, ISphereParam)

Интерфейс параметров сферы.

ksSphereParam	- интерфейс Automation
ISphereParam	- интерфейс COM

Описание:

Сфера определяется системой координат и радиусом.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

ISphereParam – свойства

radius – Радиус сферы

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iSphereParam.radius	Получить свойство (*)
radius = iSphereParam.GetRadius()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

ISphereParam – методы

GetPlacement – Получить систему координат сферы

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры конической поверхности (Интерфейсы ksConeParam, IConeParam)

Интерфейс параметров конической поверхности.

ksConeParam	- интерфейс Automation
IConeParam	- интерфейс COM

Описание:

Коническая поверхность определяется системой координат, радиусом, высотой и углом образующей. Оси X и Y системы координат лежат в плоскости основания конуса.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

IConeParam – свойства

angle – Угол образующей поверхности

Интерфейс...Тип данных: double

Синтаксис Automation:

angle = iConeParam.angle		Получить свойство (*)
angle iConeParam.GetAngle()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

height – Высота поверхности

Интерфейс...Тип данных: double

Синтаксис Automation:

height = iConeParam.height		Получить свойство (*)
height iConeParam.GetHeight()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

radius – Радиус ортогонального сечения, проходящего через начало системы координат поверхности

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iConeParam.radius		Получить свойство (*)
radius iConeParam.GetRadius()	=	Получить свойство (**)

Примечания:

1. Свойство доступно только для чтения.
2. Систему координат конической поверхности можно получить с помощью методов ksConeParam::GetPlacement и IConeParam::GetPlacement.

IConeParam – методы

GetPlacement – Получить систему координат конической поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры тора (Интерфейсы ksTorusParam, ITorusParam)

Интерфейс параметров тора.

ksTorusParam	- интерфейс Automation
ITorusParam	- интерфейс COM

Описание:

Тор определяется системой координат и радиусом центров и радиусом образующей.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

ITorusParam – свойства

generatrixRadius – Радиус образующей

Интерфейс...Тип данных: double

Синтаксис Automation:

radius iTorusParam.generatrixRadius	=	Получить свойство (*)
iTorusParam.GetGeneratrixRadius()		Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

radius – Радиус центров

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iTorusParam.radius	Получить свойство (*)
radius = iTorusParam.GetRadius()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

ITorusParam – методы

GetPlacement – Получить систему координат тора

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры плоскости (Интерфейсы ksPlaneParam, IPlaneParam)

Интерфейс параметров плоскости.

ksPlaneParam - интерфейс Automation
IPlaneParam - интерфейс COM

Описание:

Плоскость определяется системой координат. Оси X и Y системы координат лежат в плоскости.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam

IPlaneParam – методы

GetPlacement – Получить систему координат плоскости

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры цилиндрической поверхности (Интерфейсы ksCylinderParam, ICylinderParam)

Интерфейс параметров цилиндрической поверхности.

ksCylinderParam - интерфейс Automation
ICylinderParam - интерфейс COM

Описание:

Цилиндрическая поверхность определяется системой координат, радиусом и высотой. Оси X и Y лежат в плоскости основания.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

ICylinderParam – свойства

height – Высота поверхности

Интерфейс...Тип данных: double

Синтаксис Automation:

height = iCylinderParam.height		Получить свойство (*)
height iCylinderParam.GetHeight()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

radius – Радиус основания

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iCylinderParam.radius		Получить свойство (*)
radius iCylinderParam.GetRadius()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

ICylinderParam – методы

GetPlacement – Получить систему координат основания

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры NURBS – поверхности (Интерфейсы ksNurbsSurfaceParam, INurbsSurfaceParam)

Интерфейс параметров NURBS–поверхности.

ksNurbsSurfaceParam	- интерфейс Automation
INurbsSurfaceParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

ksNurbsSurfaceParam – свойства

BoundaryCount – Количество контуров, образующих границу поверхности

Интерфейс...Тип данных: long

Синтаксис Automation:

BoundaryCount = Object.BoundaryCount		Получить свойство (*)
BoundaryCount Object.GetBoundaryCount()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

INurbsSurfaceParam – методы

GetBoundaryUVNurbs – Получить параметры NURBS-представления границы поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetBoundaryUVNurbs (BOOL uv,
BOOL closed,
long loopIndex,
long * degree,
long edgeIndex,
VARIANT * points,
VARIANT * weights,
VARIANT * knots,
double * tMin,
double * tMax)

Синтаксис COM:

BOOL GetBoundaryUVNurbs (BOOL uv,
BOOL closed,
long loopIndex,
long edgeIndex,
long * degree,
VARIANT * points,
VARIANT * weights,
VARIANT * knots,
double * tMin,
double * tMax)

Входные параметры:

Uv	- TRUE - получить параметры границы поверхности в UV- параметрическом представлении исходной поверхности,
	- FALSE - получить параметры границы поверхности в 3D координатах,
Closed	- TRUE - сомкнуть, если граница разомкнутая,
	- FALSE - разомкнуть, если граница замкнутая,
loopIndex	- индекс цикла,
edgeIndex	- индекс ребра в цикле (совпадает с индексом ребра в коллекции ориентированных ребер).

Выходные параметры:

degree	- порядок NURBS (степень полинома + 1), от 3 до 10,

points	- массив параметров UV или координат вершин - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
weights	- веса точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
knots	- узлы точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
tMin, tMax	- минимальный и максимальный параметры.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

- Если признак UV равен TRUE, то в массиве points возвращаются параметры UV для Nurbs-представления границы. Параметры в массиве лежат в следующей последовательности:

  u0, v0, u1, v1, ... ui, vi.

- Если признак UV равен FALSE, то в массиве points возвращаются координаты вершин Nurbs-представления границы. Координаты точек в полученном массиве лежат в следующей последовательности:

  x0, y0, z0, x1, y1, z1, ...xi, yi, zi.

- Если индекс ребра равен -1, формируется Nurbs-представление контура границы.
- Если индекс >= 0, то формируется Nurbs-представление ребра, входящего в цикл; параметр unclamped при этом игнорируется.

GetClose – Получить тип замыкания сплайна

Интерфейс...**Синтаксис Automation**:

BOOL GetClose (BOOL paramU);

Синтаксис COM:

BOOL GetClose (BOOL paramU);

Входные параметры:

paramU	TRUE	- для параметра U,
paramU	FALSE	- для параметра V.

Возвращаемое значение:

TRUE	- сплайн замкнутый,
FALSE	- сплайн разомкнутый.

Примечание:

Значение paramU позволяет выбрать параметр представления поверхности U или V, для которого выполняется метод.

GetDegree – Получить степень сплайна

Интерфейс...**Синтаксис Automation**:

short GetDegree (BOOL paramU);

Синтаксис COM:

short GetDegree (BOOL paramU);

Входные параметры:

paramU	TRUE	- для параметра U,
paramU	FALSE	- для параметра V.

Возвращаемое значение:

- порядок NURBS (степень полинома + 1), от 3 до 10.

Примечание:

Значение paramU позволяет выбрать параметр представления поверхности U или V, для которого выполняется метод.

GetEdgesCount – Количество ребер в цикле

Интерфейс...**Синтаксис Automation**:

long GetEdgesCount (long loopIndex)

Синтаксис COM:

long GetEdgesCount (long loopIndex)

Входные параметры:

loopIndex	- номер цикла.

Возвращаемое значение:

- количество ребер в цикле.

Примечание:

Свойство доступно только для чтения.

GetMinMaxParameters – Получить параметры поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetMinMaxParameters (BOOL closedV, BOOL closedU, double * uMin, double * uMax, double * vMin, double * vMax)

Синтаксис COM:

BOOL GetMinMaxParameters (BOOL closedV, BOOL closedU, double * uMin, double * uMax, double * vMin, double * vMax)

Входные параметры:

closedV	- TRUE - получение параметров для замкнутого по V представления, - FALSE - получение параметров для разомкнутого по V представления,
closedU	- TRUE - получение параметров для замкнутого по U представления, - FALSE - получение параметров для разомкнутого по V представления.

Выходные параметры:

uMin	- значение минимального параметра по U,
uMax	- значение максимального параметра по U,
vMin	- значение минимального параметра по V,
vMax	- значение максимального параметра по V.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetNurbsParams – Получить параметры NURBS–поверхности

Интерфейс...**Синтаксис Automation**:

BOOL GetNurbsParams(BOOL unClampedV,
BOOL unClampedU,
long * degreeV,
long * degreeU,
long * nPV, long * nPU,
VARIANT * points,
VARIANT * weights,
VARIANT * knotsV,
VARIANT * knotsU);

Синтаксис COM:

BOOL GetNurbsParams(BOOL unClampedV,
BOOL unClampedU,
long * degreeV,
long * degreeU
long * nPV,
long * nPU,
VARIANT * points,
VARIANT * weights,

VARIANT * knotsV,
VARIANT * knotsU);

Входные параметры:

unClampedV	- TRUE - вернуть параметры для замкнутого по V представления, - FALSE - вернуть параметры для разомкнутого по V представления,
unClampedU	- TRUE вернуть параметры для замкнутого по U представления, - FALSE вернуть параметры для разомкнутого по U представления.

Выходные параметры:

degreeV, degreeU	- порядок NURBS (степень полинома + 1) по V и U, от 3 до 10,
nPV, nPU	- количество точек по V и U,
points	- массив координат вершин - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
weights	- веса в вершинах - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
knotsV, knotsU	- узлы точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8.

GetPeriodic – Получить периодичность сплайна

Интерфейс...**Синтаксис Automation**:

BOOL GetPeriodic (BOOL paramU);

Синтаксис COM:

BOOL GetPeriodic (BOOL paramU);

Входные параметры:

paramU	TRUE	- для параметра U,
paramU	FALSE	- для параметра V.

Возвращаемое значение:

TRUE	- сплайн периодический,
FALSE	- сплайн непериодический.

Примечание:

Значение paramU позволяет выбрать параметр представления поверхности U или V, для которого выполняется метод.

NurbsKnotCollection – Получить указатель на интерфейс массива узлов для NURBS-поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetKnotCollection (BOOL paramU);

Синтаксис COM:

LPNURBSKNOTCOLLECTION GetKnotCollection (BOOL paramU);

Входные параметры:

paramU	TRUE	- для параметра U,
paramU	FALSE	- для параметра V.

Возвращаемое значение:

- указатель на интерфейс ksNurbsKnotCollection или INurbsKnotCollection.

Примечание:

Значение paramU позволяет выбрать параметр представления поверхности U или V, для которого выполняется метод.

NurbsPoint3DCollCollection – Получить указатель на интерфейс массива массивов точек для NURBS-поверхности

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPointCollection();

Синтаксис COM:

LPNURBSPOINT3DCOLLCOLLECTION GetPointCollection();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3DCollCollection или INurbsPoint3DCollCollection.

Параметры точки для трехмерных NURBS (Интерфейсы ksNurbsPoint3dCollCollection, INurbsPoint3dCollCollection)

Интерфейс массива массивов точек для трехмерных NURBS поверхностей.

ksNurbsPoint3dCollCollection	- интерфейс Automation
INurbsPoint3dCollCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksNurbsSurfaceParam::NurbsPoint3DCollCollection.

INurbsPoint3dCollCollection – методы

Add – Добавить объект в конец массива

Интерфейс...**Синтаксис Automation**:

BOOL Add (LPDISPATCH obj);

Синтаксис COM:

BOOL Add (LPNURBSPOINT3DCOLLECTION obj);

Входной параметр:

obj	- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection параметров точки для трехмерной NURBS.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта obj не должен быть равен 0.

AddAt – Добавить объект с заданным индексом в массив

Интерфейс...**Синтаксис Automation**:

BOOL AddAt (LPDISPATCH obj, long index);

Синтаксис COM:

BOOL AddAt (LPNURBSPOINT3DCOLLECTION obj, long index);

Входные параметры:

obj	- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection параметров точки для трехмерной NURBS,
index	- индекс в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта obj не должен быть равен 0.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс...**Синтаксис Automation**:

BOOL AddBefore (LPDISPATCH obj, LPDISPATCH base);

Синтаксис COM:

BOOL AddBefore (LPNURBSPOINT3DCOLLECTION obj, LPNURBSPOINT3DCOLLECTION base);

Входные параметры:

obj	- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection добавляемого объекта,
base	- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection базового объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатели на интерфейсы добавляемого и базового объекта obj и base не должны быть равны 0.

Clear – Очистить массив

Интерфейс...**Синтаксис Automation**:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объекты удаляются только из массива, в модель изменения не передаются.

DetachByBody – Удалить указанный объект из массива

Интерфейс...**Синтаксис Automation**:

BOOL DetachByBody (LPDISPATCH obj);

Синтаксис COM:

BOOL DetachByBody (LPNURBSPOINT3DCOLLECTION obj);

Входные параметры:

obj - указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection удаляемого объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Указатель на интерфейс удаляемого объекта obj не должен быть равен 0.

DetachByIndex – Удалить объект из массива по индексу

Интерфейс...**Синтаксис Automation**:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входной параметр:

index - индекс удаляемого объекта в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Элемент массива из модели не удаляется.

FindIt – Получить индекс элемента в массиве

Интерфейс...**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
-1 - если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPNURBSPOINT3DCOLLECTION entity);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,

SYS_MAX_UINT - если элемент не найден.

Входные параметры:

entity - указатель на интерфейс компонента ksNurbsPoint3dCollection или INurbsPoint3dCollection.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION First();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION GetByIndex (long index);

Входной параметр:

index - номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION Last();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION Next();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION Prev();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dCollection или INurbsPoint3dCollection.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом точек поверхности и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс...**Синтаксис Automation**:

BOOL SetByIndex (LPDISPATCH obj, long index);

Синтаксис COM:

BOOL SetByIndex (LPNURBSPOINT3DCOLLECTION obj, long index);

Входные параметры:

obj	- указатель на интерфейс ksINurbsPoint3dCollection или INurbsPoint3dCollection присоединяемого объекта,
index	- индекс элемента в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Массив узлов для трехмерных NURBS – поверхностей (Интерфейсы ksNurbsKnotCollection, INurbsKnotCollection)

Интерфейс массива узлов для трехмерных NURBS поверхностей.

ksNurbsKnotCollection	- интерфейс Automation
INurbsKnotCollection	- интерфейс COM

Примечание:

Данный интерфейс можно получить, используя метод интерфейса ksNurbs3dParam ksNurbs3dParam::NurbsKnotCollection или интерфейса ksNurbsSurfaceParam ksNurbsSurfaceParam::NurbsKnotCollection.

INurbsKnotCollection – методы

Add – Добавить элемент в конец массива

Интерфейс...**Синтаксис Automation**:

BOOL Add (double obj);

Синтаксис COM:

BOOL Add (double obj);

Входной параметр:

obj - значение узла.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

AddAt – Добавить объект с заданным индексом в массив

Интерфейс...**Синтаксис Automation**:

BOOL AddAt (double obj, long index);

Синтаксис COM:

BOOL AddAt (double obj, long index);

Входные параметры:

obj - значение узла,
index - индекс в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс...**Синтаксис Automation**:

BOOL AddBefore (double obj, double base);

Синтаксис COM:

BOOL AddBefore (double obj, double base);

Входные параметры:

obj	- значение добавляемого узла,
base	- значение базового узла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Clear – Очистить массив

Интерфейс...**Синтаксис Automation**:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объекты удаляются только из массива, в модель изменения не передаются.

DetachByBody – Удалить указанный объект из массива

Интерфейс...**Синтаксис Automation**:

BOOL DetachByBody (double obj);

Синтаксис COM:

BOOL DetachByBody (double obj);

Входные параметры:

obj	- значение удаляемого узла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

DetachByIndex– Удалить объект из массива по индексу

Интерфейс...**Синтаксис Automation**:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входной параметр:

index	- индекс удаляемого объекта в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Элемент массива из модели не удаляется.

First – Получить первый объект в массиве

Интерфейс...**Синтаксис Automation**:

double First();

Синтаксис COM:

double First();

Возвращаемое значение:

- первый объект в массиве.

GetByIndex – Получить элемент массива по индексу

Интерфейс...**Синтаксис Automation**:

double GetByIndex (long index);

Синтаксис COM:

double GetByIndex (long index);

Входной параметр:

index	- номер объекта в массиве.

Возвращаемое значение:

- значение узла.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

Last – Получить последний объект в массиве

Интерфейс...**Синтаксис Automation**:

double Last();

Синтаксис COM:

double Last();

Возвращаемое значение:

- последний объект в массиве.

Next – Получить следующий объект в массиве

Интерфейс...**Синтаксис Automation**:

double Next();

Синтаксис COM:

double Next();

Возвращаемое значение:

- следующий объект в массиве.

Prev – Получить предыдущий объект в массиве

Интерфейс...**Синтаксис Automation**:

double Prev();

Синтаксис COM:

double Prev();

Возвращаемое значение:

- предыдущий объект в массиве.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом узлов ребра (поверхности) и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс...**Синтаксис Automation**:

BOOL SetByIndex (double obj, long index);

Синтаксис COM:

BOOL SetByIndex (double obj, long index);

Входные параметры:

obj	- значение присоединяемого узла,
index	- индекс элемента в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Параметры точки для трехмерных NURBS(Интерфейсы ksNurbsPoint3dParam, INurbsPoint3dParam)

Интерфейс параметров точки для трехмерных NURBS.

ksNurbsPoint3dParam	- интерфейс Automation
INurbsPoint3dParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam .

INurbsPoint3dParam – свойства

weight – Вес точки

Интерфейс..Тип данных: double

Синтаксис Automation:

weight = iNurbsPoint3dParam.weight	Получить свойство (*)
weight = iNurbsPoint3dParam.GetWeight()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

INurbsPoint3dParam – методы

GetPoint – Получить координаты базовой точки

Интерфейс..**Синтаксис Automation**:

BOOL GetPoint (double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint (double * x, double *y, double *z);

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Выходные параметры:

x, y, z	- координаты базовой точки.

Параметры элемента вращения (Интерфейсы ksRotatedParam, IRotatedParam)

Справка системы КОМПАС...

Интерфейс параметров элемента вращения.

ksRotatedParam	- интерфейс Automation
IRotatedParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksSurface::GetSurfaceParam.

IRotatedParam – свойства

angleNormal – Угол вращения в прямом направлении

Интерфейс...Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

angleNormal = iRotatedParam.angleNormal	=	Получить свойство(*)
iRotatedParam.angleNormal = angleNormal	=	Установить свойство (*)
angleNormal = iRotatedParam.GetAngleNormal()	=	Получить свойство (**)
iRotatedParam.SetAngleNormal(angleNormal)		Установить свойство (**)

angleReverse – Угол вращения в обратном направлении

Интерфейс..Справка системы КОМПАС...

Тип данных: double

Синтаксис Automation:

angleReverse = iRotatedParam.angleReverse	=	Получить свойство(*)
iRotatedParam.angleReverse = angleReverse	=	Установить свойство (*)
angleReverse = iRotatedParam.GetAngleReverse()	=	Получить свойство (**)
iRotatedParam.SetAngleReverse(angleReverse)		Установить свойство (**)

direction – Направление вращения

Интерфейс..Справка системы КОМПАС...

Тип данных: long

Значения свойства:

Типы направлений...

Синтаксис Automation:

direction iRotatedParam.direction	=	Получить свойство(*)
iRotatedParam.direction direction	=	Установить свойство (*)
direction iRotatedParam.GetDirection()	=	Получить свойство (**)
iRotatedParam.SetDirection(direction)		Установить свойство (**)

Примечание:

Прямое направление - вдоль нормали к плоскости эскиза и всегда против часовой стрелки.

Нормаль, проведенная к поверхности тела, всегда выходит из тела.

toroidShape – Признак тороида

Интерфейс..Справка системы КОМПАС...

Тип данных: BOOL

Значения свойства:

TRUE	- тороид,
FALSE	- сфероид.

Синтаксис Automation:

toroidShape iRotatedParam.toroidShape	=	Получить свойство(*)
iRotatedParam.toroidShape toroidShape	=	Установить свойство (*)
toroidShape iRotatedParam.GetToroidShape()	=	Получить свойство (**)
iRotatedParam.SetToroidShape (toroidShape		Установить свойство (**)

Математическая кривая в трехмерном пространстве (Интерфейсы ksCurve3D, ICurve3D)

Интерфейс математической кривой в трехмерном пространстве.

ksCurve3D	- интерфейс Automation
ICurve3D	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейсов ksAxisConefaceDefinition::GetCurve3D, IDefaultObject::GetCurve3D, ksAxisEdgeDefinition::GetCurve3D, ksAxis2PlanesDefinition::GetCurve3D, ksAxisOperationsDefinition::GetCurve3D, ksAxis2PointsDefinition::GetCurve3D, ksEdgeDefinition::GetCurve3D.

ICurve3D – методы

CalculatePolygon – Рассчитать полигон

Интерфейс...**Синтаксис Automation**:

VARIANT CalculatePolygon(double step);

Синтаксис COM:

BOOL CalculatePolygon(VARIANT * points,
double step);

Входные параметры:

step	- шаг для расчета полигона.

Возвращаемое значение:

	- массив SafeArray координат точек на кривой, полученных с шагом step.

Примечание:

1. Если шаг задан равным 0, то используется шаг применяемый в системе КОМПАС, который рассчитывается с учетом габаритов детали.
2. Если шаг задан -1, то используется шаг применяемый в системе КОМПАС при отрисовке кривых. Он рассчитывается с учетом следующих параметров:

- габариты детали и масштаба отображения модели.
- тип возвращаемого массива VT_ARRAY | VT_R8
- значения координат в массиве лежат в последовательности x1, y1, z1, x2, y2, z2 ... xn, yn, zn.

GetCurveParam – Получить параметры линии, окружности, эллипса, NURBS или NULL

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetCurveParam();

Синтаксис COM:

LPUNKNOWN GetCurveParam();

Возвращаемое значение:

- указатель на интерфейс параметра кривой.

Примечание:

В соответствии с типом объекта метод вернет указатель на интерфейс параметра:

- отрезка	ksILineSeg3dParam или ILineSeg3dParam,
- дуги окружности	ksArc3dParam или IArc3dParam,
- окружности	ks ICircle3dParam или ICircle3dParam,
- эллипса	ksEllipse3dParam или IEllipse3dParam,
- NURBS	kIsNurbs3dParam или INurbs3dParam.

Зная тип объекта, можно базовый интерфейс привести к конечному (IDispatch или IUnknown). Для определения типа объекта можно воспользоваться методами IsLineSeg, IsArc,IsCircle, IsEllipse, IsNurbs.

GetDerivativeT – Получить первую производную по T

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeT (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeT (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- первая производная по T.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координата точки возвращается в координатах компонента.

GetDerivativeTT – Получить вторую производную по T

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeTT (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeTT (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- вторая производная по T.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координата точки возвращается в координатах компонента.

GetDerivativeTTT – Получить третью производную по T

Интерфейс...**Синтаксис Automation**:

BOOL GetDerivativeTTT (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetDerivativeTTT (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- третья производная по T.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координата точки возвращается в координатах компонента.

GetGabarit – Получить габаритный параллелепипед кривой

Интерфейс...**Синтаксис Automation**:

BOOL GetGabarit (double * x1,
double *y1,
double *z1,
double * x2,
double *y2,
double *z2);

Синтаксис COM:

BOOL GetGabarit (double * x1,
double *y1,
double *z1,
double * x2,
double *y2,
double *z2);

Выходные параметры:

x1, y1, z1, x2, y2, z2 - координаты вершин габаритного параллелепипеда.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Координаты вершин параллелепипеда возвращаются в системе координат компонента.

GetLength – Получить длину кривой

Интерфейс...**Синтаксис Automation**:

double GetLength (unsigned long bitVector);

Синтаксис COM:

double GetLength (unsigned long bitVector);

Входные параметры:

bitVector - единицы измерения в интервале [ST_MIX_MM..ST_MIX_M].

Возвращаемое значение:

длина кривой	- в случае успешного завершения,
0	- в случае неудачи.

GetMetricLength – Получить метрическую длину кривой

Интерфейс...**Синтаксис Automation**:

double GetMetricLength (double startParam, double endParam);

Синтаксис COM:

double GetMetricLength (double startParam, double endParam);

Входные параметры:

startParam	- начальное значение параметра T в параметрическом представлении кривой,
endParam	- конечное значение параметра T в параметрическом представлении кривой.

Возвращаемое значение:

метрическая длина кривой	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax.

GetNormal – Получить направление нормали

Интерфейс...**Синтаксис Automation**:

BOOL GetNormal (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetNormal (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- вектор нормали.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координата точки возвращается в координатах компонента.

GetNurbs3dParam – Получить NURBS-представление кривой

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetNurbs3dParam();

Синтаксис COM:

LPNURBS3DPARAM GetNurbs3dParam();

Возвращаемое значение:

- указатель на интерфейс ksNurbs3dParam/ INurbs3dParam.

Примечание:

Метод позволяет получить параметры кривой в NURBS-представлении. Для получения параметров кривой в NURBS-представлении создается математическая NURBS-кривая, аналогичная заданной. Если заданная кривая является NURBS-кривой, то возвращаются ее параметры без создания NURBS-представления.

GetParamMin – Получить начальное значение параметра T в параметрическом представлении кривой

Интерфейс...**Синтаксис Automation**:

double GetParamMin();

Синтаксис COM:

double GetParamMin();

Возвращаемое значение:

Значение параметра T - в случае успеха,
0 - в случае неудачи.

GetParamMax – Получить конечное значение параметра T в параметрическом представлении кривой

Интерфейс...**Синтаксис Automation**:

double GetParamMax();

Синтаксис COM:

double GetParamMax();

Возвращаемое значение:

Значение параметра T - в случае успеха,
0 - в случае неудачи.

GetPoint – Получить координаты точки на кривой

Интерфейс...**Синтаксис Automation**:

BOOL GetPoint (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- координаты точки на кривой.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координаты точки возвращаются в системе координат компонента.

GetTangentVector – Получить направление касательного вектора

Интерфейс...**Синтаксис Automation**:

BOOL GetTangentVector (double paramT, double * x, double *y, double *z);

Синтаксис COM:

BOOL GetTangentVector (double paramT, double * x, double *y, double *z);

Входной параметр:

paramT	- значение параметра T в параметрическом представлении кривой.

Выходные параметры:

x, y, z	- касательный вектор.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Граничные значения параметра T получаются при помощи методов GetParamMin и GetParamMax. Координата точки возвращается в координатах компонента.

IsArc – Определить, является ли кривая дугой окружности

Интерфейс...**Синтаксис Automation**:

BOOL IsArc();

Синтаксис COM:

BOOL IsArc();

Возвращаемое значение:

TRUE	- кривая является дугой окружности,
FALSE	- кривая не является дугой окружности.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра дуги окружности ksArc3dParam или IArc3dParam.

IsCircle – Определить, является ли кривая окружностью

Интерфейс...**Синтаксис Automation**:

BOOL IsCircle();

Синтаксис COM:

BOOL IsCircle();

Возвращаемое значение:

TRUE	- кривая является окружностью,
FALSE	- кривая не является окружностью.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра окружности ksCircle3dParam или ICircle3dParam.

IsClosed – Определить, является ли кривая замкнутой

Интерфейс...**Синтаксис Automation**:

BOOL IsClosed();

Синтаксис COM:

BOOL IsClosed();

Возвращаемое значение:

TRUE	- кривая замкнута,
FALSE	- кривая разомкнута.

IsDegenerate – Определить, является ли кривая вырожденной

Интерфейс...**Синтаксис Automation**:

BOOL IsDegenerate();

Синтаксис COM:

BOOL IsDegenerate();

Возвращаемое значение:

TRUE	- кривая вырожденная,
FALSE	- кривая невырожденная.

IsEllipse – Определить, является ли кривая эллипсом

Интерфейс...**Синтаксис Automation**:

BOOL IsEllipse();

Синтаксис COM:

BOOL IsEllipse();

Возвращаемое значение:

TRUE	- кривая является эллипсом,
FALSE	- кривая не является эллипсом.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра эллипса ksEllipse3dParamили IEllipse3dParam.

IsLineSeg – Определить, является ли кривая отрезком

Интерфейс...**Синтаксис Automation**:

BOOL IsLineSeg();

Синтаксис COM:

BOOL IsLineSeg();

Возвращаемое значение:

TRUE - кривая является отрезком,
FALSE - кривая не является отрезком.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра отрезка ksLineSeg3dParam или ILineSeg3dParam.

IsNurbs – Определить, является ли кривая NURBS

Интерфейс...**Синтаксис Automation**:

BOOL IsNurbs();

Синтаксис COM:

BOOL IsNurbs();

Возвращаемое значение:

TRUE - кривая является NURBS,
FALSE - кривая не является NURBS.

Примечание:

Данный метод удобно использовать для приведения базового интерфейса (IUnknown в COM и IDispatch в Automation) к конечному интерфейсу трехмерных объектов. Зная тип объекта, можно привести его к соответствующему интерфейсу. Метод GetCurveParam вернет указатель на интерфейс параметра NURBS ksNurbs3dParam или INurbs3dParam.

IsPeriodic – Определить, является ли замкнутая кривая периодической

Интерфейс...**Синтаксис Automation**:

BOOL IsPeriodic();

Синтаксис COM:

BOOL IsPeriodic();

Возвращаемое значение:

TRUE - кривая периодична,
FALSE - кривая не периодична.

IsPlanar – Определить, является ли кривая плоской

Интерфейс...**Синтаксис Automation**:

BOOL IsPlanar();

Синтаксис COM:

BOOL IsPlanar();

Возвращаемое значение:

TRUE	- кривая плоская,
FALSE	- кривая неплоская.

NearPointProjection – Получить ближайшую проекцию точки на поверхность

Интерфейс...**Синтаксис Automation**:

BOOL NearPointProjection (double x, double y, double z, double * t, BOOL ext);

Синтаксис COM:

BOOL NearPointProjection (double x, double y, double z, double * t, BOOL ext);

Входные параметры:

x, y, z	- координаты исходной пространственной точки,
ext	- если проекция точки вне кривой: - TRUE проецировать на продолжение кривой, - FALSE найти ближайшую граничную точку кривой. Если проекция точки внутри кривой, значение этого параметра не учитывается.

Выходные параметры:

t	- параметрические координаты точки на кривой.

Возвращаемое значение:

TRUE	- если проекция точки попала на саму кривую или если ext равен TRUE,
FALSE	- если проекция точки не попала на саму кривую.

Параметры отрезка (Интерфейсы ksLineSeg3dParam,ILineSeg3dParam)

Интерфейс параметров отрезка.

ksLineSeg3dParam	- интерфейс Automation
ILineSeg3dParam	- интерфейс COM

Описание:

Отрезок определяется координатами начальной и конечной точек.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksCurve3D::GetCurveParam.

ILineSeg3dParam – методы

GetPointFirst – Получить координаты первой точки отрезка

Интерфейс...**Синтаксис Automation**:

BOOL GetPointFirst (double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPointFirst (double * x, double *y, double *z);

Выходные параметры:

x, y, z	- координаты первой точки отрезка.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Значения x, y, z возвращаются в системе координат компонента.

GetPointLast – Получить координаты второй точки отрезка

Интерфейс...**Синтаксис Automation**:

BOOL GetPointLast (double * x, double *y, double *z);

Синтаксис COM:

BOOL GetPointLast (double * x, double *y, double *z);

Выходные параметры:

x, y, z	- координаты первой точки отрезка.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Значения x, y, z возвращаются в системе координат компонента.

Параметры окружности (Интерфейсы ksCircle3dParam,ICircle3dParam)

Интерфейс параметров окружности.

ksCircle3dParam	- интерфейс Automation
ICircle3dParam	- интерфейс COM

Описание:

Окружность определяется координатами центра и радиусом.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksCurve3D::GetCurveParam.

ICircle3dParam – свойства

radius – Радиус окружности

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iCircle3dParam.radius	Получить свойство (*)
radius = iCircle3dParam.GetRadius()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

ICircle3dParam – методы

GetPlacement – Получить указатель на интерфейс положения окружности

Интерфейс..**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры эллипса (Интерфейсы ksEllipse3dParam, IEllipse3dParam)

Интерфейс параметров эллипса.

ksEllipse3dParam - интерфейс Automation
IEllipse3dParam - интерфейс COM

Описание:

Эллипс определяется координатами центра и двумя радиусами.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksCurve3D::GetCurveParam.

IEllipse3dParam – свойства

majorRadius – Больший радиус эллипса

Интерфейс...

Тип данных: double

Синтаксис Automation:

majorRadius = iEllipse3dParam.majorRadius	Получить свойство (*)
majorRadius = iEllipse3dParam.GetMajorRadius()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

minorRadius – Меньший радиус эллипса

Интерфейс...

Тип данных: double

Синтаксис Automation:

minorRadius = iEllipse3dParam.minorRadius	Получить свойство (*)
minorRadius = iEllipse3dParam.GetMinorRadius()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

IEllipse3dParam – методы

GetPlacement – Получить указатель на интерфейс положения эллипса

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры трехмерной дуги(Интерфейсы ksArc3dParam,IArc3dParam)

Интерфейс параметров трехмерной дуги.

ksArc3dParam	- интерфейс Automation
IArc3dParam	- интерфейс COM

Описание:

Трехмерная дуга определяется положением плоскости, в которой эта дуга лежит, углом раствора и радиусом. Угол раствора отсчитывается от оси OX плоскости дуги.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksCurve3D::GetCurveParam.

IArc3dParam – свойства

angle – Угол раствора дуги

Интерфейс...Тип данных: double

Синтаксис Automation:

angle = iArc3dParam.angle		Получить свойство (*)
angle iArc3dParam.GetAngle()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

radius – Радиус дуги

Интерфейс...Тип данных: double

Синтаксис Automation:

radius = iArc3dParam.radius		Получить свойство (*)
radius iArc3dParam.GetRadius()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

IArc3dParam – методы

GetPlacement – Получить систему координат дуги

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Параметры трехмерных NURBS (Интерфейсы ksNurbs3dParam,INurbs3dParam)

Интерфейс параметров трехмерных NURBS.

ksNurbs3dParam	- интерфейс Automation
INurbs3dParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksCurve3D::GetCurveParam.

INurbs3dParam – свойства

close – Замкнутость сплайна

Интерфейс...Тип данных: BOOL

Синтаксис Automation:

close = iNurbs3dParam.close		Получить свойство (*)
close iNurbs3dParam.GetClose()	=	Получить свойство (**)

Значения свойства:

TRUE	- сплайн замкнутый,
FALSE	- сплайн разомкнутый.

Примечания:

Свойство доступно только для чтения.

degree – Порядок NURBS (степень полинома + 1), от 3 до 10

Интерфейс...Тип данных: short

Синтаксис Automation:

degree iNurbs3dParam.degree	=	Получить свойство (*)
degree iNurbs3dParam.GetDegree()	=	Получить свойство (**)

Примечания:

Свойство доступно только для чтения.

periodic – Периодичность сплайна

Интерфейс...Тип данных: BOOL

Синтаксис Automation:

periodic iNurbs3dParam.periodic	=	Получить свойство (*)
periodic iNurbs3dParam.GetPeriodic()	=	Получить свойство (**)

Значения свойства:

TRUE	- сплайн периодический,
FALSE	- сплайн непериодический.

Примечания:

Свойство доступно только для чтения.

INurbs3dParam – методы

GetMinMaxParameters – Получить параметры кривой

Интерфейс...**Синтаксис Automation**:

BOOL GetMinMaxParameters (BOOL closed, double * tMin, double * tMax);

Синтаксис COM:

BOOL GetMinMaxParameters (BOOL closed, double * tMin, double * tMax);

Входные параметры:

Closed	- TRUE - получение параметров для замкнутого представления, - FALSE - получение параметров для разомкнутого представления.

Выходные параметры:

tMin	- значение минимального параметра,
tMax	- значение максимального параметра.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetNurbsPoints3DParams – Получить параметры точек для Nurbs 3D

Интерфейс...**Синтаксис Automation**:

BOOL GetNurbsPoints3DParams(BOOL unClamped, VARIANT * points, VARIANT * weights, VARIANT * knots);

Синтаксис COM:

BOOL GetNurbsPoints3DParams(BOOL unClamped, VARIANT * points, VARIANT * weights, VARIANT * knots);

Входные параметры:

unClamped - true - вернуть параметры точек для замкнутого представления,
- false - вернуть параметры точек для разомкнутого представления.

Выходные параметры:

points - координаты точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
weights - веса точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8,
knots - узлы точек - массив SafeArray вещественных чисел VT_ARRAY | VT_R8.

Примечания:

Координаты точек в полученном массиве лежат в следующей последовательности:
x0, y0, z0, x1, y1, z1, ...xi, yi, zi.

NurbsKnotCollection – Получить указатель на интерфейс массива узлов для трехмерного NURBS

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetKnotCollection();

Синтаксис COM:

LPNURBSKNOTCOLLECTION GetKnotCollection();

Возвращаемое значение:

- указатель на интерфейс ksNurbsKnotCollection или INurbsKnotCollection массива узлов для трехмерных NURBS поверхностей.

NurbsPoint3DCollection – Получить указатель на интерфейс массива точек для трехмерного NURBS

Интерфейс...LPDISPATCH GetPointCollection();

Синтаксис COM:

LPNURBSPOINT3DCOLLECTION GetPointCollection();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3DCollection или INurbsPoint3DCollection массива точек для трехмерных NURBS.

Массив точек для трехмерных NURBS (Интерфейсы ksNurbsPoint3dCollection, INurbsPoint3dCollection)

Интерфейс массива точек для трехмерных NURBS.

ksNurbsPoint3dCollection	- интерфейс Automation
INurbsPoint3dCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksNurbs3dParam::NurbsPoint3DCollection.

INurbsPoint3dCollection – методы

Add – Добавить объект в конец массива

Интерфейс...**Синтаксис Automation**:

BOOL Add (LPDISPATCH obj);

Синтаксис COM:

BOOL Add (LPNURBSPOINT3DPARAM obj);

Входной параметр:

obj	- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam параметров точки для трехмерной NURBS.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта obj не должен быть равен 0.

AddAt – Добавить объект с заданным индексом в массив

Интерфейс...**Синтаксис Automation**:

BOOL AddAt (LPDISPATCH obj, long index);

Синтаксис COM:

BOOL AddAt (LPNURBSPOINT3DPARAM obj, long index);

Входные параметры:

obj	- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam параметров точки для трехмерной NURBS,

index - индекс в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Указатель на интерфейс добавляемого объекта obj не должен быть равен 0.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс...**Синтаксис Automation**:

BOOL AddBefore (LPDISPATCH obj, LPDISPATCH base);

Синтаксис COM:

BOOL AddBefore (LPNURBSPOINT3DPARAM obj, LPNURBSPOINT3DPARAM base);

Входные параметры:

obj - указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam добавляемого объекта,
base - указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam базового объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Указатели на интерфейсы добавляемого и базового объекта obj и base не должны быть равны 0.

Clear – Очистить массив

Интерфейс...**Синтаксис Automation**:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Объекты удаляются только из массива, в модель изменения не передаются.

DetachByBody – Удалить указанный объект из массива

Интерфейс...**Синтаксис Automation**:

BOOL DetachByBody (LPDISPATCH obj);

Синтаксис COM:

BOOL DetachByBody (LPNURBSPOINT3DPARAM obj);

Входные параметры:

obj	- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam удаляемого объекта.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Указатель на интерфейс удаляемого объекта obj не должен быть равен 0.

DetachByIndex – Удалить объект из массива по индексу

Интерфейс...**Синтаксис Automation**:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входной параметр:

index	- индекс удаляемого объекта в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Элемент массива из модели не удаляется.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...**Синтаксис Automation**:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPNURBSPOINT3DPARAM GetByIndex (long index);

Входной параметр:

index	- номер объекта в массиве.

Возвращаемое значение:

указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam.

GetCount – Получить количество элементов в массиве

Интерфейс...**Синтаксис Automation**:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

FindIt – Получить индекс элемента в коллекции

Интерфейс...**Синтаксис Automation**:

long FindIt (LPDISPATCH entity);

Входные параметры:

entity	- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam точки для трехмерного NURBS.

Возвращаемое значение:

индекс элемента в массиве.	- если элемент найден в коллекции,
-1	- элемент в коллекции не найден.

Синтаксис COM:

unsigned long FindIt (LPNURBSPOINT3D entity);

Входные параметры:

entity - указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam точки для трехмерного NURBS.

Возвращаемое значение:

индекс элемента в массиве. - если элемент найден в коллекции,

SYS_MAX_UINT - элемент в коллекции не найден.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH First();

Синтаксис COM:

LPNURBSPOINT3DPARAM First();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Last();

Синтаксис COM:

LPNURBSPOINT3DPARAM Last();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Next();

Синтаксис COM:

LPNURBSPOINT3DPARAM Next();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...**Синтаксис Automation**:

LPDISPATCH Prev();

Синтаксис COM:

LPNURBSPOINT3DPARAM Prev();

Возвращаемое значение:

- указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam.

Refresh – Обновить массив

Интерфейс...**Синтаксис Automation**:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом точек ребра (поверхности), и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс...**Синтаксис Automation**:

BOOL SetByIndex (LPDISPATCH obj, long index);

Синтаксис COM:

BOOL SetByIndex (LPNURBSPOINT3DPARAM obj, long index);

Входные параметры:

obj - указатель на интерфейс ksNurbsPoint3dParam или INurbsPoint3dParam присоединяемого объекта,
index - индекс элемента в массиве.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Массив узлов для трехмерных NURBS поверхностей (Интерфейсы ksNurbsKnotCollection, INurbsKnotCollection)

Объект Дерева построения (Интерфейсы ksFeature, IFeature)

Справка системы КОМПАС...

kompas.chm::/659_81_3_Derevo_modeli.htm

Интерфейс объекта Дерева построения.

ksFeature	- интерфейс Automation
IFeature	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов следующих интерфейсов:

- Интерфейса объекта Дерева построения ksFeatureCollection,
- Интерфейса детали или подсборки в составе сборки ksPart.

IFeature – свойства

excluded – Признак включения или выключения из расчета

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/962_117_10_Iskljuchenie_obwekto.htm

Тип данных: BOOL

Синтаксис Automation:

excluded = iFeature.excluded	Получить свойство (*)
iFeature.excluded = excluded	Установить свойство (*)
excluded = iFeature.GetExcluded()	Получить свойство (**)
iFeature.SetExcluded(excluded)	Установить свойство (**)

name – Имя объекта

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

name = iFeature.name		Получить свойство (*)
name = iFeature.GetName()		Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

objectError – Признак ошибки объекта

Интерфейс...

Тип данных: long.

Синтаксис Automation:

objectError object.objectError	=	Получить свойство (*)
objectError object.GetObjectError()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

type – Тип объекта модели

Интерфейс...

Тип данных: short

Синтаксис Automation:

type = iFeature.type	Получить свойство (*)
type = iFeature.GetType()	Получить свойство (**)

Примечание:

1. Свойство доступно только для чтения.
2. Тип возвращаемого объекта может принимать значения:

o3d_part	- деталь,	
o3d_entity	- объект,	
o3d_mateConstraint	- сопряжение,	
o3d_body	-тело.	IBody или ksBody.

Выбирается из Obj3dType.

updateStamp – Значение, определяющее время изменения геометрии

Интерфейс...

Тип данных: long

Синтаксис Automation:

updateStamp = iFeature.updateStamp	=	Получить свойство (*)
updateStamp = iFeature.GetUpdateStamp(	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

Feature – методы

AttributeCollection – Получить массив атрибутов

Интерфейс...

Синтаксис Automation:

ksAttribute3DCollection* AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb);

Синтаксис COM:

LPATTRIBUTE3DCOLLECTION AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb);

Входные параметры:

key1, key2, key3, key4	- ключи для поиска по ключам либо 0,
Numb	- номер типа атрибута для поиска по номеру либо 0.

Возвращаемое значение:

- указатель на массив атрибутов ksAttribute3DCollection.

Примечание:

Атрибут - это дополнительная неграфическая информация, связанная с объектом или несколькими объектами чертежа. Такая информация может быть представлена в виде числа, строки текста, а также таблицы с фиксированным или переменным числом строк.

В дальнейшем значения атрибутов могут использоваться для поиска объектов, а также обрабатываться различными приложениями (например, системой проектирования спецификаций, различными расчетными программами и т.п.).

Типы (описания структуры) атрибутов могут храниться как непосредственно в документе, так и в специальных файлах (библиотеках типов атрибутов). Эти файлы имеют расширение lat.

AttributeCollectionEx – Получить массив атрибутов из источника вставки

Интерфейс...

Синтаксис Automation:

LPDISPATCH AttributeCollectionEx (long key1,
long key2,
long key3,
long key4,
double numb,
LPDISPATCH sourcePart);

Синтаксис COM:

LPATTRIBUTE3DCOLLECTION AttributeCollectionEx (long key1,
long key2,
long key3,
long key4,
double numb,
LPPART sourcePart);

Входные параметры:

key1, key2, key3, key4	- ключи для поиска по ключам либо 0,
Numb	- номер типа атрибута для поиска по номеру либо 0,
sourcePart	- указатель на интерфейс вставки-источника ksPart или IPart.

Возвращаемое значение:

- указатель на интерфейс коллекции атрибутов ksAttribute3DCollection или IAttribute3DCollection.

Примечание:

Атрибут - это дополнительная неграфическая информация, связанная с объектом или несколькими объектами чертежа. Такая информация может быть представлена в виде числа, строки текста, а также таблицы с фиксированным или переменным числом строк.

В дальнейшем значения атрибутов могут использоваться для поиска объектов, а также обрабатываться различными приложениями (например, системой проектирования спецификаций, различными расчетными программами и т.п.).

Типы (описания структуры) атрибутов могут храниться как непосредственно в документе, так и в специальных файлах (библиотеках типов атрибутов). Эти файлы имеют расширение lat.

BodyCollection – Получить указатель на интерфейс массива трехмерных тел

Интерфейс...

Синтаксис Automation:

LPBODYCOLLECTION BodyCollection();

Синтаксис COM:

LPBODYCOLLECTION BodyCollection();

Возвращаемое значение:

- указатель на интерфейс массива тел ksBodyCollection или IBodyCollection.

EntityCollection – Получить массив объектов (грани, ребра, вершины)

Интерфейс...

Синтаксис Automation:

ksEntityCollection * EntityCollection (short objType);

Синтаксис COM:

LPENTITYCOLLECTION EntityCollection (short objType);

Входные параметры:

objType - тип объектов (грани, ребра, вершины).

Возвращаемое значение:

указатель на интерфейс ksEntityCollection или IEntityCollection - в случае успеха,
NULL - в случае неудачи.

GetObject – Получить объект модели, связанный с объектом дерева

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/919_109_2_Ierarkhicheskaja_para.htm

Синтаксис Automation:

LPDISPATCH GetObject();

Синтаксис COM:

LPUNKNOWN GetObject();

Возвращаемое значение:

- указатель на интерфейс объекта.

Примечание:

Тип возвращаемого объекта выбирается из Obj3dType. Может принимать значение:

o3d_part	- деталь,	
o3d_entity	- объект,	
o3d_mateConstraint	- сопряжение,	
o3d_body	-тело	IBody или kIsBody

GetOwnerFeature – Получить указатель на интерфейс объекта, в состав которого входит данный объект

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/919_109_2_Ierarkhicheskaja_para.htm

Синтаксис Automation:

LPDISPATCH GetOwnerFeature();

Синтаксис COM:

LPFEATURE GetOwnerFeature();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

IsModified – Определить, модифицирована ли модель с момента последнего перестроения

Интерфейс...

Синтаксис Automation:

BOOL IsModified (BOOL recursive);

Синтаксис COM:

BOOL IsModified (BOOL recursive);

Входной параметр:

recursive	TRUE	Определять по вложенным объектам.

Возвращаемое значение:

TRUE	- модель изменена с момента последнего перестроения,
FALSE	- модель не изменена.

IsRollBacked – Лежит ли объект ниже конца построения модели

Интерфейс...

Синтаксис Automation:

BOOL IsRollBacked();

Синтаксис COM:

BOOL IsRollBacked();

Возвращаемое значение:

TRUE	- объект лежит ниже конца построения модели,
FALSE	- объект лежит выше конца построения модели.

IsValid – Получить индикатор доступности объекта

Интерфейс...

Синтаксис Automation:

BOOL IsValid();

Синтаксис COM:

BOOL IsValid();

Возвращаемое значение:

TRUE	- объект доступен,
FALSE	- объект недоступен.

Примечание:

Метод позволяет получить информацию о том, что элемент дерева построения доступен для редактирования. Элемент дерева построения может стать недоступным, например, если он исключен из расчета или указатель окончания построения перемещен в дереве выше этого элемента.

SubFeatureCollection – Получить массив объектов, входящих в данный объект

Интерфейс...

Синтаксис Automation:

LPDISPATCH SubFeatureCollection (BOOL through, BOOL libObject);

Синтаксис COM:

LPFEATURECOLLECTION SubFeatureCollection (BOOL through, BOOL libObject);

Входные параметры:

Through	TRUE	- выдавать все объекты, включая скрытые,
libObject	TRUE	- выдавать состав библиотечных элементов.

Возвращаемое значение:

- указатель на интерфейс ksFeatureCollection или IFeatureCollection.

VariableCollection – Получить указатель на интерфейс массива внешних переменных

Интерфейс...

Справка системы КОМПАС...

`kompas.chm::/1868_175_5_vneshn_peremen.htm`

Синтаксис Automation:

LPDISPATCH VariableCollection();

Синтаксис COM:

LPVARIABLECOLLECTION VariableCollection();

Возвращаемое значение:

- указатель на интерфейс ksVariableCollection или IVariableCollection.

Примечание:

Изменения в полученном массиве не отображаются в модели немедленно. Чтобы изменения вступили в силу, необходимо вызвать метод ksPart::RebuildModel.

VariableCollectionEx – Получить массив переменных из источника

Интерфейс...

Синтаксис Automation:

LPDISPATCH VariableCollectionEx(BOOL source);

Синтаксис COM:

LPVARIABLECOLLECTION VariableCollectionEx(BOOL source);

Входные параметры:

Source	- признак получения списка переменных и значений из источника.

Возвращаемое значение:

- указатель на интерфейс коллекции переменных ksVariableCollection или IvariableCollection.

Примечание:

Метод позволяет получать и изменять переменные в исходном документе детали без открытия документа в отдельном окне или вслепую.

Массив объектов Дерева построения (Интерфейсы ksFeatureCollection, IFeatureCollection)

Интерфейс массива объектов Дерева построения.

ksFeatureCollection	- интерфейс Automation
IFeatureCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов:

- ksFeature::SubFeatureCollection,
- ksMacro3DDefinition::FeatureCollection,
- ksDocument3D::FeatureCollection.

ksFeatureCollection – методы

Add – Добавить объект в массив

Интерфейс...

Синтаксис Automation:

BOOL Add (ksFeature* obj);

Синтаксис COM:

BOOL Add (LPFEATURE obj);

Входные параметры:

obj - указатель на интерфейс компонента ksFeature или IFeature.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

AddAt – Добавить в массив объект с заданным индексом

Интерфейс...

Синтаксис Automation:

BOOL AddAt (ksFeature* obj, long index);

Синтаксис COM:

BOOL AddAt (LPFEATURE obj, long index);

Входные параметры:

obj - указатель на интерфейс компонента ksFeature или IFeature,
index - индекс нового элемента.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

Примечание.

При добавлении тела в макроэлемент в макроэлемент добавляются все операции данного тела на момент создания макроэлемента. Само тело в макроэлемент не добавляется.

AddBefore – Добавить объект перед указанным объектом в массиве

Интерфейс...

Синтаксис Automation:

BOOL AddBefore (ksFeature* obj, ksFeature* base);

Синтаксис COM:

BOOL AddBefore (LPFEATURE obj, LPFEATURE base);

Входные параметры:

obj - указатель на интерфейс компонента ksFeature или IFeature,
base - указатель на элемент массива.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

AttributeCollection – Получить массив атрибутов

Интерфейс...

Синтаксис Automation:

ksAttribute3DCollection* AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb);

Синтаксис COM:

LPATTRIBUTE3DCOLLECTION AttributeCollection (long key1,
long key2,
long key3,
long key4,
double numb);

Входные параметры:

key1, key2, key3, key4	- ключи для поиска по ключам либо 0,
numb	- номер типа атрибута для поиска по номеру либо 0.

Возвращаемое значение:

Указатель на массив атрибутов ksAttribute3DCollection

Примечание:

Атрибут - это дополнительная неграфическая информация, связанная с объектом или несколькими объектами чертежа. Такая информация может быть представлена в виде числа, строки текста, а также таблицы с фиксированным или переменным числом строк.

Значения атрибутов могут использоваться для поиска объектов, а также обрабатываться различными приложениями, например, системой проектирования спецификаций, различными расчетными программами и т.п.

Типы (описания структуры) атрибутов могут храниться как непосредственно в документе, так и в специальных файлах - библиотеках типов атрибутов. Эти файлы имеют расширение lat.

Структура параметров табличного атрибута.

Clear – Очистить динамический массив объектов

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

DetachByBody – Удалить указанный объект из массива

Интерфейс...

Синтаксис Automation:

BOOL DetachByBody (LPDISPATCH obj);

Синтаксис COM:

BOOL DetachByBody (LPFEATURE obj);

Входные параметры:

obj	- указатель на интерфейс удаляемого объекта ksFeature или IFeature.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Указатель на интерфейс удаляемого объекта obj не должен быть 0.

DetachByIndex – Удалить объект из массива по индексу

Интерфейс...

Синтаксис Automation:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входные параметры:

index	- индекс элемента.

Возвращаемое значение:

TRUE - в случае успеха,
FALSE - в случае неудачи.

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
-1 - если элемент не найден.

Синтаксис COM:

unsigned long FindIt(LPFEATURE entity);

Возвращаемое значение:

индекс элемента - если элемент найден в массиве,
SYS_MAX_UINT - если элемент не найден.

Входные параметры:

entity - указатель на интерфейс компонента ksFeature или IFeature.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPFEATURE First();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPFEATURE GetByIndex (long index);

Входной параметр:

index	- номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

GetByName – Получить указатель на интерфейс объекта в массиве по имени

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByName (BSTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Синтаксис COM:

LPFEATURE GetByName (LPOLESTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Входные параметры:

name	- имя объекта,
testFullNa me	- признак полного имени: TRUE - name - полное имя, FALSE - имя name может быть частью полного имени,
testIgnore Case	- признак игнорирования регистра символов: TRUE - игнорировать регистр, FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

GetCount – Получить количество элементов в массиве

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:
long GetCount();
Возвращаемое значение:

- количество элементов массива.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...
Синтаксис Automation:
LPDISPATCH Last();
Синтаксис COM:
LPFEATURE Last();
Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature .

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...
Синтаксис Automation:
LPDISPATCH Next();
Синтаксис COM:
LPFEATURE Next();
Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...
Синтаксис Automation:
LPDISPATCH Prev();
Синтаксис COM:
LPFEATURE Prev();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

Refresh – Обновить массив

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом объектов дерева и удаляются все предыдущие изменения в массиве.

SetByIndex – Заменить объект с указанным индексом в массиве на присланный объект

Интерфейс...

Синтаксис Automation:

BOOL SetByIndex (ksFeature* obj, long index);

Синтаксис COM:

BOOL SetByIndex (LPFEATURE obj, long index);

Входные параметры:

obj	- указатель на интерфейс компонента ksFeature или IFeature,
index	- индекс элемента в массиве.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Массив внешних переменных (Интерфейсы ksVariableCollection, IVariableCollection)

Справка системы КОМПАС...

kompas.chm::/1868_175_5_vneshn_peremen.htm

Интерфейс динамического массива внешних переменных.

ksVariableCollection	- интерфейс Automation
IVariableCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов ksFeature::VariableCollectionEx, ksFeature::VariableCollection, ksPart::VariableCollection.

IVariableCollection – методы

AddNewVariable – Добавить новую переменную

Интерфейс..

Синтаксис Automation:

LPDISPATCH AddNewVariable(BSTR name, double value, BSTR note);

Синтаксис COM:

LPVARIABLE AddNewVariable(LPOLESTR name, double value, LPOLESTR note);

Входные параметры:

name	- имя переменной,
value	- значение переменной,
Note	- комментарий к переменной.

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

Примечание:

Переменную можно добавлять только для ksFeature и IFeature.

GetByIndex – Получить указатель на интерфейс элемента массива по индексу

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPVARIABLE GetByIndex (long index);

Входной параметр:

index	- индекс элемента в массиве.

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

GetByName – Получить указатель на интерфейс элемента массива по имени

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetByName (BSTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Синтаксис COM:

LPVARIABLE GetByName (LPOLESTR name,

BOOL testFullName,

BOOL testIgnoreCase);

Входные параметры:

name	- имя объекта,
testFullNa	- признак полного имени:
me	TRUE - name - полное имя,
	FALSE - имя name может быть частью полного имени,
testIgnore	- признак игнорирования регистра символов имени:
Case	TRUE - игнорировать регистр,
	FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

GetCount – Получить количество элементов массива

Интерфейс..

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество элементов массива.

First – Получить указатель на интерфейс первого элемента массива

Интерфейс..

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPVARIABLE First();

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

Last – Получить указатель на интерфейс последнего элемента массива

Интерфейс..

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPVARIABLE Last();

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

Next – Получить указатель на интерфейс следующего элемента массива

Интерфейс..

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPVARIABLE Next();

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

Prev – Получить указатель на интерфейс предыдущего элемента массива

Интерфейс..

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPVARIABLE Prev();

Возвращаемое значение:

- указатель на интерфейс внешней переменной модели ksVariable или IVariable.

Refresh – Обновить массив внешних переменных

Интерфейс..

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

После вызова данной функции удаляются все изменения в массиве внешних переменных компонента, которые не были закреплены вызовом функции RebuildModel. Таким образом, возникает полное соответствие массива внешних переменных в памяти с массивом внешних переменных компонента.

Вызов данной функции эквивалентен отказу пользователя от перестроения модели после изменения значений внешних переменных (модель не перестраивается, а измененные значения переменных "забываются", переменные принимают прежние значения).

RemoveVariable – Удалить переменную

Интерфейс..

Синтаксис Automation:

BOOL RemoveVariable (BSTR name);

Синтаксис COM:

BOOL RemoveVariable (LPOLESTR name);

Входные параметры:

name	- имя переменной.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

1. Переменную можно удалить только для ksFeature и IFeature.
2. Массив автоматически обновляется в случае удачного завершения.

Внешняя переменная (Интерфейсы ksVariable, IVariable)

Справка системы КОМПАС...

`KOMPAS.chm::/1868_175_5_vneshn_peremen.htm`

Интерфейс внешней параметрической переменной модели.

ksVariable	- интерфейс Automation
Ivariable	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейса ksVariableCollection.

IVariable – свойства

displayName – Отображаемое имя переменной

Интерфейс…

Тип данных: BSTR

Синтаксис Automation:

displayName = iVariable.displayName		Получить свойство (*)
displayName iVariable.GetdisplayName()	=	Получить свойство (**)

Примечание.

Свойство только для чтения.

Expression – Выражение

Интерфейс…

Тип данных: строка

Синтаксис Automation:

expression = iVariable.expression	Получить свойство (*)
iVariable.expression = expression	Установить свойство (*)
iVariable.GetExpression()	Получить свойство (**)
iVariable.SetExpression(expression)	Установить свойство (**)

Примечание:

Свойство позволяет получить или задать строковое выражение для расчета значения переменной.

Перечень элементов, которые могут быть использованы в выражении см. в разделе:

Операторы, функции, константы

`KOMPAS.chm::/operator_funktion_const.htm`
справочной системы КОМПАС 3D.

Обратите внимание на то, что оператор «=» в выражениях не используется.

External – Внешняя переменная

Интерфейс…

Тип данных: BOOL

Синтаксис Automation:

external = iVariable.external	Получить свойство (*)
iVariable.external = external	Установить свойство (*)
external = iVariable.GetExternal()	Получить свойство (**)
iVariable.SetExternal(external)	Установить свойство (**)

Information – Информационная переменная

Интерфейс…

Тип данных: Long

Синтаксис Automation:

Information = Object.Information	Получить свойство (*)
Object.Information = Information	Установить свойство (*)
Information = Object.GetInformation()	Получить свойство (**)
Object.SetInformation(Information)	Установить свойство (**)

Примечание.

Свойство позволяет задать и получить признак "информационная" для переменных модели.

linkDocName – Имя документа, в котором хранится ссылочная переменная

Интерфейс…

Тип данных: строка

Синтаксис Automation:

linkDocName = iVariable.linkDocName		Получить свойство (*)
linkDocName iVariable.GetLinkDocName()	=	Получить свойство (**)

Примечание:

Свойство только для чтения.

linkVarName – Имя переменной, на которую задана ссылка

Интерфейс…

Тип данных: строка

Синтаксис Automation:

linkVarName = iVariable.linkVarName		Получить свойство (*)
linkVarName iVariable.GetLinkVarName()	=	Получить свойство (**)

Примечание:

Свойство только для чтения.

name – Имя переменной

Интерфейс…

Тип данных: строка

Синтаксис Automation:

name = iVariable.name	Получить свойство (*)
name = iVariable.GetName()	Получить свойство (**)

Примечание:

Свойство только для чтения.

note – Комментарий к переменной

Интерфейс…

Тип данных: строка

Синтаксис Automation:

note = iVariable.note	Получить свойство (*)
iVariable.note = note	Установить свойство (*)
note = iVariable.GetNote()	Получить свойство (**)
iVariable.SetNote(name)	Установить свойство (**)

parameterNote – Имя параметра переменной

Интерфейс…

Тип данных: строка

Синтаксис Automation:

parameterNote iVariable.parameterNote	=	Получить свойство (*)
parameterNote iVariable.GetParameterNote()	=	Получить свойство (**)

Примечание:

Свойство только для чтения.

Pseudonym – Псевдоним переменной

Интерфейс…

Тип данных: строка

Синтаксис Automation:

pseudonym = iVariable.pseudonym	Получить свойство (*)
iVariable.pseudonym = pseudonym	Установить свойство (*)
pseudonym = iVariable.GetPseudonym()	Получить свойство (**)
iVariable.SetPseudonym(pseudonym)	Установить свойство (**)

value – Значение переменной

Интерфейс…

Тип данных: double

Синтаксис Automation:

value = iVariable.value	Получить свойство (*)
iVariable.value = value	Установить свойство (*)
value = iVariable.GetValue()	Получить свойство (**)
iVariable.SetValue(value)	Установить свойство (**)

IVariable – методы

SetLink – Установить ссылку на переменную

Интерфейс…

Синтаксис Automation:

BOOL SetLink (BSTR doc,BSTR name);

Синтаксис COM:

BOOL SetLink (LPOLESTR doc,LPOLESTR name);

Входные параметры:

Name - имя переменной, на которую будет сделана ссылка,

Doc - полный путь к документу, в котором лежит переменная с именем name.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Чтобы удалить установленную ссылку, необходимо задать следующие значения параметров:

name = NULL,

doc = NULL.

Интерфейс массива атрибутов объектов модели (Интерфейсы ksAttribute3DCollection, IAttribute3DCollection)

Интерфейс массива атрибутов объектов модели.

ksAttribute3DCollection - интерфейс Automation
IAttribute3DCollection - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов ksFeatureCollection::AttributeCollection, ksFeature::AttributeCollectionEx, ksFeature::AttributeCollection, ksDocument3D::AttributeCollection.

IAttribute3DCollection – методы

FindIt – Получить индекс элемента массива

Интерфейс...

Синтаксис Automation:

long GetByIndex (ksAttribute3D* obj);

Синтаксис COM:

unsigned long GetByIndex (LPATTRIBUTE3D obj);

Возвращаемое значение:

индекс элемента массива.

First – Получить указатель на интерфейс первого атрибута

Интерфейс...

Синтаксис Automation:

ksAttribute3D* First();

Синтаксис COM:
LPATTRIBUTE3D First();
Возвращаемое значение:

указатель на интерфейс ksAttribute3D или IAttribute3D.

GetCount – Получить количество элементов массива атрибутов

Интерфейс...
Синтаксис Automation:
long GetCount();
Синтаксис COM:
unsigned long GetCount();
Возвращаемое значение:

количество элементов массива.

GetByIndex – Получить указатель на интерфейс атрибута по индексу

Интерфейс...
Синтаксис Automation:
ksAttribute3D* GetByIndex (long index);
Синтаксис COM:
LPATTRIBUTE3D GetByIndex (long index);
Возвращаемое значение:

указатель на интерфейс ksAttribute3D или IAttribute3D.

Last – Получить указатель на интерфейс последнего атрибута

Интерфейс...
Синтаксис Automation:
ksAttribute3D* Last();
Синтаксис COM:
LPATTRIBUTE3D Last();
Возвращаемое значение:

указатель на интерфейс ksAttribute3D или IAttribute3D.

Next – Получить указатель на интерфейс следующего атрибута

Интерфейс...

Синтаксис Automation:

ksAttribute3D* Next();

Синтаксис COM:

LPATTRIBUTE3D Next();

Возвращаемое значение:

указатель на интерфейс ksAttribute3D или IAttribute3D.

Prev – Получить указатель на интерфейс предыдущего атрибута

Интерфейс...

Синтаксис Automation:

ksAttribute3D* Prev();

Синтаксис COM:

LPATTRIBUTE3D Prev();

Возвращаемое значение:

указатель на интерфейс ksAttribute3D или IAttribute3D.

Refresh – Обновить массив атрибутов

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом атрибутов объекта в документе.

Select – Выделить атрибуты данного типа или с данными ключами

Интерфейс...

Синтаксис

BOOL Select (long key1,
long key2,
long key3,
long key4,
double numb,
int objType);

Входные параметры:

key1, key2, key3, key4	- ключи для поиска по ключам либо 0,
numb	- номер типа атрибута для поиска по номеру либо 0,
objType	- тип объекта: все объекты дерева построений, кроме компонентов, сопряжений, группы сопряжений.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Работа с атрибутами объектов модели (Интерфейсы ksAttribute3D, IAttribute3D)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_ATR_LST_BODY_ATR.htm

Интерфейс атрибута объектов модели.

ksAttribute3D	- интерфейс Automation
IAttribute3D	- интерфейс COM

Примечание:

Данный интерфейс может быть получен следующими способами:

- использованием методов интерфейсов ksAttribute3DCollection, ksAttributeObject.
- использованием экспортных функций ksCreateAttr3DExW, ksCreateAttr3DEx, ksCreateAttr3D, ksChoiceAttr3D.

IAttribute3D – свойства

reference – Указатель на атрибут

Интерфейс...

Тип данных: long

Синтаксис Automation:

reference = iAttribute3D.reference		Получить свойство (*)
reference iAttribute3D.GetReference()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

IAttribute3D – методы

FeatureCollection – Получить массив объектов заданного типа, имеющих данный атрибут

Интерфейс...

Синтаксис Automation:

ksFeatureCollection* FeatureCollection(int objType);

Синтаксис COM:

LPFEATURECOLLECTION FeatureCollection(int objType);

Входные параметры:

objType	- тип трехмерного объекта.

Возвращаемое значение:

указатель на интерфейс ksFeatureCollection.	- массив, содержащий объекты дерева построений, на которые установлен данный атрибут, и которые имеют заданный тип.

Примечание:

Если objType = o3d_unknown - выдавать все объекты.

GetNameType – Получить имя типа атрибута

Интерфейс...

Синтаксис Automation:

BSTR GetNameType();

Синтаксис COM:

LPOLESTR GetNameType();

Входные параметры:

objType	- тип трехмерного объекта.

Возвращаемое значение:

Имя файла библиотеки типов - в случае успешного завершения.

Измерение расстояния и угла между двумя примитивами (Интерфейсы ksMeasurer, IMeasurer)

Справка системы КОМПАС...

kompas.chm::/976_Glava120_Izmerenija.htm

Интерфейс для измерений расстояния и угла между двумя примитивами.

ksMeasurer - интерфейс Automation
IMeasurer - интерфейс COM

Примечание:

1. Интерфейс применим для граней, ребер, вершин.
2. Данный интерфейс может быть получен с использованием метода интерфейса ksPart::GetMeasurer.

IMeasurer – свойства

angle – Угол между объектами

Интерфейс..

Тип данных: double

Синтаксис Automation:

angle = iMeasurer.angle	Получить свойство (*)
angle = iMeasurer.GetAngle()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

distance – Расстояние между объектами

Интерфейс..

Тип данных: double

Синтаксис Automation:

distance = iMeasurer.distance	Получить свойство (*)

distance = iMeasurer.GetDistance()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

extendObject1 – Признак вычисления за пределами первого объекта

Интерфейс..

Тип данных: BOOL

Синтаксис Automation:

extendObject1 = iMeasurer.extendObject1	Получить свойство (*)
iMeasurer.extendObject1 = extendObject1	Установить свойство (*)
extendObject1 = iMeasurer.GetExtendObject1()	Получить свойство (**)
iMeasurer.SetExtendObject1(extendObject1)	Установить свойство (**)

Примечание:

Если значение свойства равно TRUE, то следует вычислять за пределами второго объекта.

extendObject2 – Признак вычисления за пределами второго объекта

Интерфейс..

Тип данных: BOOL

Синтаксис Automation:

extendObject2 = iMeasurer.extendObject2	Получить свойство (*)
iMeasurer.extendObject2 = extendObject2	Установить свойство (*)
extendObject2 = iMeasurer.GetExtendObject2()	Получить свойство (**)
iMeasurer.SetExtendObject2(extendObject2)	Установить свойство (**)

Примечание:

Если значение свойства равно TRUE, то следует вычислять за пределами первого объекта.

MinDistance –Минимальное расстояние

Интерфейс..

Тип данных: double

Синтаксис Automation:

MinDistance = IMeasurer.MinDistance	Получить свойство (*)
MinDistance = IMeasurer.GetMinDistance()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

unit – Единицы измерения

Интерфейс..

Тип данных: unsigned long

Синтаксис Automation:

unit = iMeasurer.unit	Получить свойство (*)
iMeasurer.unit = unit	Установить свойство (*)
unit = iMeasurer.GetUnit()	Получить свойство (**)
iMeasurer.SetUnit(unit)	Установить свойство (**)

Примечание:

Свойство находится в интервале [ST_MIX_MM..ST_MIX_M].

IMeasurer – методы

Calc – Произвести вычисления

Интерфейс..

Синтаксис Automation:

BOOL Calc();

Синтаксис COM:

BOOL Calc();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetMaxPoint1 – Получить первую точку отрезка максимального расстояния

Интерфейс..

Синтаксис Automation:

BOOL GetMaxPoint1 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetMaxPoint1 (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetMaxPoint2 – Получить вторую точку отрезка максимального расстояния

Интерфейс..

Синтаксис Automation:

BOOL GetMaxPoint2 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetMaxPoint2 (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetMinPoint1 – Получить первую точку отрезка минимального расстояния

Интерфейс..

Синтаксис Automation:

BOOL GetMinPoint1 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetMinPoint1 (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetMinPoint12 – Получить вторую точку отрезка минимального расстояния

Интерфейс..

Синтаксис Automation:

BOOL GetMinPoint2 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetMinPoint2 (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetNormalPoint1 – Получить первую точку отрезка расстояния по нормали

Интерфейс..

Синтаксис Automation:

BOOL GetNormalPoint1 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetNormalPoint1 (double *x,
double*y,
double*z);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetNormalPoint2 – Получить вторую точку отрезка расстояния по нормали

Интерфейс..

Синтаксис Automation:

BOOL GetNormalPoint2 (double *x,
double *y,
double *z);

Синтаксис COM:

BOOL GetNormalPoint2 (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

GetPoint1 – Получить ближайшую точку на первом объекте или на продолжении объекта

Интерфейс..

Синтаксис Automation:

BOOL GetPoint1 (double *x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint1 (double *x, double *y, double *z);

Входной параметр:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Значения x, y, z возвращаются в системе координат компонента.

GetPoint2 – Получить ближайшую точку на втором объекте или на продолжении объекта

Интерфейс..

Синтаксис Automation:

BOOL GetPoint2 (double *x, double *y, double *z);

Синтаксис COM:

BOOL GetPoint2 (double *x, double *y, double *z);

Входной параметр:

x, y, z - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Значения x, y, z возвращаются в системе координат компонента.

GetObject1 – Получить первый объект

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetObject1();

Синтаксис COM:

LPUNKNOWN GetObject1();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown первого объекта.

Примечание:

Возвращаемым значением может быть указатель на интерфейс грани, ребра, вершины. Зная тип объекта, можно привести его интерфейс к конечному 3D интерфейсу.

GetObject2 – Получить второй объект

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetObject2();

Синтаксис COM:

LPUNKNOWN GetObject2();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown второго объекта.

Примечание:

Возвращаемым значением может быть указатель на интерфейс грани, ребра, вершины. Зная тип объекта, можно привести его интерфейс к конечному 3D интерфейсу.

IsAngleValid – Определить применимость расчета угла

Интерфейс..

Синтаксис Automation:

BOOL IsAngleValid();

Синтаксис COM:

BOOL IsAngleValid();

Возвращаемое значение:

TRUE	- для данных объектов угол имеет смысл,
FALSE	- для данных объектов угол смысла не имеет.

MaxDistance – Получить максимальное расстояние

Интерфейс..

Синтаксис Automation:

double MaxDistance();

Синтаксис COM:

double MaxDistance();

Возвращаемое значение:

- Максимальное расстояние между объектами (двумя гранями, гранью и ребром и т. п.)

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

MeasureResult – Получить результат измерения

Интерфейс..

Синтаксис Automation:

long MeasureResult();

Синтаксис COM:

ksMeasureResultEnum MeasureResult();

Возвращаемое значение:

- Результат измерения расстояния и угла между поверхностями из перечисления ksMeasureResultEnum.

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

NormalDistance – Получить расстояние по нормали (межосевое или расстояние между гранями)

Интерфейс..

Синтаксис Automation:

double NormalDistance();

Синтаксис COM:

double NormalDistance();

Возвращаемое значение:

- Максимальное расстояние по нормали между объектами (двумя гранями, гранью и ребром и т. п.).

Примечание:

Метод доступен после вызова метода ksMeasurer::Calc (Automation) или IMeasurer::Calc (COM).

SetObject1 – Задать первый объект

Интерфейс..

Синтаксис Automation:

BOOL SetObject1 (LPDISPATCH obj);

Синтаксис COM:

BOOL SetObject1 (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown первого объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Примечание:

Входным параметром может быть грань, ребро, вершина.

SetObject2 – Задать второй объект

Интерфейс..

Синтаксис Automation:

BOOL SetObject2 (LPDISPATCH obj);

Синтаксис COM:

BOOL SetObject2 (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown первого объекта.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Входным параметром может быть грань, ребро, вершина.

Расчет координат точек при S– образном соединении прямолинейных ребер (Интерфейс ISTrackingPointsMeasurer)

COM интерфейс для расчета координат точек при S– образном соединении прямолинейных ребер.

ISTrackingPointsMeasurer - интерфейс COM

Примечание:

1. Интерфейс для расчета координат точки начала скругления сегмента ломаной при изменении величины радиуса, образующего скругление.
2. Данный интерфейс может быть получен с использованием метода интерфейса IPart::GetMathematic3D с параметром type = o3d_sTrackingPointsMeasurer.

ISTrackingPointsMeasurer – методы

Calculate – Рассчитать точки

Интерфейс...

Синтаксис COM:

int Calculate();

Возвращаемое значение:

1	- успешное завершение,
0	- в случае неудачи.

GetResultPoint1 – Получить рассчитанную точку 1

Интерфейс...

Синтаксис COM:

void GetResultPoint1(double * X,
double * Y,
double * Z);

Выходные параметры:

X,Y,Z	- координаты точки 1, рассчитанные функцией Calculate.

GetResultPoint2 – Получить рассчитанную точку 2

Интерфейс...

Синтаксис COM:

void GetResultPoint2(double * X,
double * Y,
double * Z);

Выходные параметры:

X,Y,Z	- координаты точки 2, рассчитанные функцией Calculate.

SetPoint1 – Установить точку первого сегмента

Интерфейс...

Синтаксис COM:

void SetPoint1(double X,
double Y,
double Z);

Входные параметры:

X,Y,Z	- координаты точки первого сегмента.

SetPoint2 – Установить точку второго сегмента

Интерфейс...

Синтаксис COM:

void SetPoint2(double X,
double Y,
double Z);

Входные параметры:

X,Y,Z - координаты точки второго сегмента.

SetRadius1 – Установить радиус расчета для первой точки

Интерфейс...

Синтаксис COM:

void SetRadius1(double Val);

Входные параметры:

Val - радиус расчета для первой точки.

SetRadius2 – Установить радиус расчета для первой точки

Интерфейс...

Синтаксис COM:

void SetRadius2(double Val);

Входные параметры:

Val - радиус расчета для второй точки.

Визуальные свойства материала (Интерфейсы ksColorParam, IColorParam)

Справка системы КОМПАС...

`kompas.chm::/688_82_9_Upravlenie_cvetom_i_sv.htm`

Интерфейс параметров цвета и визуальных свойств материала.

ksColorParam - интерфейс Automation
IColorParam - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksPart::ColorParam.

IColorParam – свойства

ambient – Общий свет

Интерфейс...

Тип данных:double.

Значения свойства:

Общий свет (от 0 до 1).

Синтаксис Automation:

ambient = iColorParam.ambient	Получить свойство (*)
iColorParam.ambient = ambient	Установить свойство (*)
ambient = iColorParam.GetAmbient()	Получить свойство (**)
iColorParam.SetAmbient (ambient)	Установить свойство (**)

color – Цвет

Интерфейс...

Тип данных: long

Синтаксис Automation:

color = iColorParam.color	Получить свойство (*)
iColorParam.color = color	Установить свойство (*)
color = iColorParam.GetColor()	Получить свойство (**)
iColorParam.SetColor(color)	Установить свойство (**)

diffuse – Диффузия

Интерфейс...

Тип данных: double

Значения свойства:

Диффузия (от 0 до 1).

Синтаксис Automation:

Diffuse = iColorParam.diffuse	Получить свойство (*)
iColorParam.diffuse = diffuse	Установить свойство (*)
Diffuse = iColorParam.GetDiffuse()	Получить свойство (**)
iColorParam.SetDiffuse (diffuse)	Установить свойство (**)

emission – Излучение

Интерфейс...

Тип данных: double

Значения свойства:

Излучение (от 0 до 1).

Синтаксис Automation:

emission = iColorParam.emission	Получить свойство (*)
iColorParam.emission = emission	Установить свойство (*)
emission = iColorParam.GetEmission()	Получить свойство (**)
iColorParam.SetEmission(emission)	Установить свойство (**)

shininess – Блеск

Интерфейс...

Тип данных: double

Значения свойства:

Блеск (от 0 до 1).

Синтаксис Automation:

shininess = iColorParam.shininess	Получить свойство (*)
iColorParam.shininess = shininess	Установить свойство (*)

shininess = iColorParam.GetShininess()	Получить свойство (**)
iColorParam.SetShininess (shininess)	Установить свойство (**)

specularity – Зеркальность

Интерфейс...

Тип данных: double

Значения свойства:

Зеркальность (от 0 до 1).

Синтаксис Automation:

specularity = iColorParam.specularity	Получить свойство (*)
iColorParam.specularity = specularity	Установить свойство (*)
specularity = iColorParam.GetSpecularity()	Получить свойство (**)
iColorParam.SetSpecularity (specularity)	Установить свойство (**)

transparency – Прозрачность

Интерфейс...

Тип данных: double

Значения свойства:

Прозрачность (от 0 до 1).

Синтаксис Automation:

transparency = iColorParam.transparency	Получить свойство (*)
iColorParam.transparency = transparency	Установить свойство (*)
transparency = iColorParam.GetTransparency()	Получить свойство (**)
iColorParam.SetTransparency (transparency)	Установить свойство (**)

useColor – Используемый цвет (цвет источника, цвет хозяина, цвет слоя, собственный цвет)

Интерфейс...

Тип данных: UseColor.

Синтаксис Automation:

useColor = Object.useColor	Получить свойство (*)
Object.useColor.color = useColor	Установить свойство (*)
useColor = Object.GetUseColor()	Получить свойство (**)
Object.SetUseColory(useColor)	Установить свойство (**)

IColorParam – методы

Clear – Очистить свойства цвета объекта

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Примечание:

Цвет будет установлен как у детали.

Интерфейс результатов редактирования объекта (Интерфейсы ksObject3DNotifyResult, IObject3DNotifyResult)

Интерфейс результатов редактирования объекта документа-модели.

ksObject3DNotifyResult	- интерфейс Automation
IObject3DNotifyResult	- интерфейс COM

В Automation получить интерфейс результата можно получить при помощи метода ksPart::GetObject3DNotifyResult.

В COM получить интерфейс результата можно при помощи метода IPart::GetObject3DNotifyResult.

Примечание:

В процессе обработки события Object3DNotifyEnum можно получить информацию о редактировании объекта.

IObject3DNotifyResult – методы

GetFeatureCollection – Получить массив удаляемых объектов

Интерфейс...

Синтаксис Automation:

ksFeatureCollection * GetFeatureCollection();

Синтаксис COM:

IFeatureCollection * GetFeatureCollection();

Возвращаемое значение:

- указатель на интерфейс массива удаляемых объектов ksFeatureCollection (IFeatureCollection)

GetNotifyType – Получить тип события

Интерфейс...

Синтаксис Automation:

long GetNotifyType();

Синтаксис COM:

long GetNotifyType();

Возвращаемое значение:

- тип события.

GetPlacement – Получить координаты точки в системе координат листа

Интерфейс...

Синтаксис Automation:

ksPlacement * GetPlacement();

Синтаксис COM:

IPlacement * GetPlacement();

Возвращаемое значение:

- указатель на интерфейс исходных локальных систем координат ksPlacement (IPlacement).

GetProcessType – Тип процесса

Синтаксис:

long GetProcessType();

Возвращаемое значение:

Тип процесса из перечисления ProcessTypeEnum - в случае успеха,

Источник событий объектов документа – модели Object3DNotify

Интерфейс событий объектов документа-модели...

Источник событий для объектов документа–модели.

В Automation источник событий Object3DNotify можно получить при помощи метода ksPart::GetObject3DNotify.

Примечание:

Интерфейс позволяет следить и управлять редактированием объектов документа-модели.

Трехмерное тело

Интерфейсы ksBody, IBody

Массив граней

Интерфейсы ksFaceCollection, IFaceCollection

Грань

Интерфейсы ksFaceDefinition,IFaceDefinition

Интерфейсы, получаемые от ksFaceDefinition

Массив циклов грани

Интерфейсы ksLoopCollection, ILoopCollection

Цикл грани

Интерфейсы ksLoop, ILoop

Массив ребер (Интерфейсы ksEdgeCollection, IEdgeCollection)

Интерфейс массива ориентированных ребер.

ksEdgeCollection	- интерфейс Automation
IEdgeCollection	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов:

- ksFaceDefinition::EdgeCollection,
- ksPolyLineDefinition::EdgeCollection,
- ksEdgeDefinition::EdgeCollection,
- ksLoop::EdgeCollection.

IEdgeCollection – методы

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt(LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPEDGEDEFINITION entity);

Входные параметры:

entity	- указатель на интерфейс компонента ksEdgeDefinition или IEdgeDefinition.

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

First – Получить указатель на интерфейс первого объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:
LPEDGEDEFINITION First();
Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

GetByIndex – Получить указатель на интерфейс объекта в массиве по индексу

Интерфейс...
Синтаксис Automation:
LPDISPATCH GetByIndex (long index);
Синтаксис COM:
LPEDGEDEFINITION GetByIndex (long index);
Входной параметр:

index - номер объекта в массиве.

Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

GetCount – Получить количество элементов в массиве

Интерфейс...
Синтаксис Automation:
long GetCount();
Синтаксис COM:
long GetCount();
Возвращаемое значение:

- количество элементов массива.

Last – Получить указатель на интерфейс последнего объекта в массиве

Интерфейс...
Синтаксис Automation:
LPDISPATCH Last();

Синтаксис COM:

LPEDGEDEFINITION Last();

Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

Next – Получить указатель на интерфейс следующего объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPEDGEDEFINITION Next();

Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPEDGEDEFINITION Prev();

Возвращаемое значение:

- указатель на интерфейс ksEdgeDefinition или IEdgeDefinition.

Refresh – Обновить массив

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

После вызова данной функции возникает полное соответствие массива с массивом ребер грани и удаляются все предыдущие изменения в массиве.

Триангуляция (аппроксимирующая поверхность грани)

Интерфейсы ksTessellation, ksTessellation

Триангуляционная пластина

Интерфейсы ksFacet, ksFacet

Интерфейсы, получаемые от ksEdgeDefinition

Массив ориентированных ребер

Интерфейсы ksOrientedEdgeCollection, IOrientedEdgeCollection

Ориентированное ребро

Интерфейсы ksOrientedEdge, IOrientedEdge

Массив ребер

Интерфейсы ksEdgeCollection, IEdgeCollection

Триангуляция (аппроксимирующая поверхность грани)

Интерфейсы ksTessellation, ksTessellation

Интерфейс результатов пересечения тел (Интерфейсы ksIntersectionResult, IIntersectionResult)

Интерфейс результатов пересечения тел.

ksIntersectionResult - интерфейс Automation
IIntersectionResult - интерфейс COM

Примечание:

1. Интерфейс используется при проверке наличия пересечения тел в сборке. См. функцию ksBody::CheckIntersectionWithBody.
2. Два тела в сборке могут иметь более одного пересечения. См. также Типы пересечений
3. Данный интерфейс можно получить от интерфейсов трехмерного тела ksBody или IBody.

IIntersectionResult – методы

GetCount – Получить количество пересечений

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество пересечений.

GetIntersectionType – Получить тип пересечения по индексу

Интерфейс...

Синтаксис Automation:

long GetIntersectionType (long index);

Синтаксис COM:

long GetIntersectionType(long index);

Входные параметры:

index - индекс пересечения.

Возвращаемое значение:

- тип пересечения.

Примечание:

Два тела в сборке могут иметь более одного пересечения. См. также Типы пересечений

МЦХ тела вращения или выдавливания (Интерфейсы ksMassInertiaParam, IMassInertiaParam)

Справка системы КОМПАС...

kompas.chm::/609_Glava73_Masso-centrovochnye.htm

Интерфейс параметров расчета массово-центровочных характеристик.

ksMassInertiaParam - интерфейс Automation
IMassInertiaParam - интерфейс COM

Аналог данного интерфейса при использовании API экспортных функций - MassInertiaParamStruct.

Примечание:

1. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct. Аналогом интерфейса при использовании API экспортных функций является MassInertiaParamStruct. Наполнение данными производится в методе ksMathematic2D::ksCalcMassInertiaProperties, аналог экспортной функции ksCalcMassInertiaProperties.
2. В трехмерной автоматизации указатель на интерфейс ksMassInertiaParam можно получить при помощи метода ksPart::CalcMassInertiaProperties или ksBody::MassInertiaParam . В 3D COM указатель на интерфейс IMassInertiaParam получается методом IPart::CalcMassInertiaProperties или IBody::MassInertiaParam.
3. Размерность длины и размерность массы, возвращаемых интерфейсом, данных находятся в интервале [ST_MIX_MM..ST_MIX_KG]. Она устанавливается при получении интерфейса, либо через SetBitVectorValue.

   Смотрите также:KompasObject

IMassInertiaParam– свойства

f – Площадь

Интерфейс...

Тип данных: double

Синтаксис Automation:

f = iMassInertiaParam.f		Получить свойство (*)
f iMassInertiaParam.GetF()	=	Получить свойство (**)

Примечание:

1. Свойство доступно только для чтения.
2. Свойство применимо только для трехмерных объектов.

Ix, Iy, Iz – Осевые моменты инерции в глобальной (исходной) системе координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

Ix = iMassInertiaParam.Ix	Получить свойство (*)

Ix iMassInertiaParam.GetLx()	=	Получить свойство (**)

Примечание:

Свойства доступны только для чтения.

Ixy, Ixz, Iyz – Центробежные моменты инерции в глобальной (исходной) системе координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

Ixy = iMassInertiaParam.Ixy		Получить свойство (*)
Ixy iMassInertiaParam.GetLxy()	=	Получить свойство (**)

Примечание:

Свойства только для чтения.

jx, jy, jz – Осевые моменты инерции в центральной системе координат

Интерфейс...

Осевые моменты инерции в центральной системе координат.

Тип данных: double

Синтаксис Automation:

jx = iMassInertiaParam.jx		Получить свойство (*)
jx iMassInertiaParam.GetJx()	=	Получить свойство (**)

Примечание:

Свойства доступны только для чтения.

jxjy, jxz, jyz – Центробежные моменты инерции в центральной системе координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

jxy = iMassInertiaParam.jxy		Получить свойство (*)
jxy iMassInertiaParam.GetJxy()	=	Получить свойство (**)

Примечание:

Свойства доступны только для чтения.

jx0z, jy0z, jx0y – Плоскостные моменты инерции

Интерфейс...

Тип данных: double

Синтаксис Automation:

jx0z = iMassInertiaParam.jx0z		Получить свойство (*)
jx0z iMassInertiaParam.GetJx0z()	=	Получить свойство (**)

Примечание:

Свойства доступны только для чтения.

jx$_j$0, jy0, jz0 – Главный центральный момент инерции

Интерфейс...

Главный центральный момент инерции (в главной центральной системе координат – с началом в центре масс и осями, ориентированными так, что все центробежные моменты равны нулю).

Тип данных: double

Синтаксис Automation:

jx0 = iMassInertiaParam.jx0		Получить свойство (*)
jx0 iMassInertiaParam.GetJx0()	=	Получить свойство (**)

Примечание:

1. Свойство доступно только для чтения.
2. Свойство применимо только для трехмерных объектов.

m – Масса тела

Интерфейс...

Тип данных: double

Синтаксис Automation:

m = iMassInertiaParam.m		Получить свойство (*)
m iMassInertiaParam.GetM()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

r – Плотность материала

Интерфейс...

Тип данных: double

Синтаксис Automation:

r = iMassInertiaParam.r		Получить свойство (*)
r iMassInertiaParam.GetR()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

v – Объем тела

Интерфейс...

Тип данных: double

Синтаксис Automation:

v = iMassInertiaParam.v		Получить свойство (*)
v iMassInertiaParam.GetV()	=	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

xc, yc, zc – Координаты центра масс в глобальной (исходной) системе координат

Интерфейс...

Тип данных: double

Синтаксис Automation:

xc = iMassInertiaParam.xc		Получить свойство (*)
xc iMassInertiaParam.GetXc()	=	Получить свойство (**)

Примечание:

Свойства доступны только для чтения.

IMassInertiaParam – методы

GetAxisX, GetAxisY, GetAxisZ – Получить вектора направлений главных центральных осей инерции

Интерфейс...

Синтаксис Automation:

BOOL GetAxisX(double *x, double *y, double *z);

BOOL GetAxisY(double *x, double *y, double *z);

BOOL GetAxisZ(double *x, double *y, double *z);

Синтаксис COM:

BOOL GetAxisX(double *x, double *y, double *z);

BOOL GetAxisY(double *x, double *y, double *z);

BOOL GetAxisZ(double *x, double *y, double *z);

Выходные параметры:

x, y, z	- вектор направления.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод применим только для трехмерных объектов.

SetBitVectorValue – Установить значение битового вектора

Интерфейс...

Синтаксис Automation:

BOOL SetBitVectorValue (long bitVector, BOOL state);

Синтаксис COM:

BOOL SetBitVectorValue (long bitVector, BOOL state);

Входные параметры:

bitVector	- битовый вектор, состоящий из флагов, определяющих размерность длины, размерность массы и тип тела (тело вращения или выдавливания) для расчета МЦХ,
state	- состояние битового вектора: TRUE - включен, FALSE - выключен.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Размерности и типы тел для расчета МЦХ...

Запрос к системе (Интерфейсы ksRequestInfo3D, IRequestInfo)

Интерфейс параметров запроса к системе.

ksRequestInfo3D	- интерфейс Automation
IRequestInfo	- интерфейс COM

Примечания:

1. Данный интерфейс позволяет установить параметры процесса указания местоположения точки или объектов (UserGetPlacementAndEntity):
2. Указатель на данный интерфейс можно получить методом ksDocument3D::GetRequestInfo.
3. Методами данного интерфейса можно получить/изменить:

- строку подсказки в процессе;
- функцию обратной связи для фильтрации объектов в процессе указания;
- функцию обратной связи процесса.

4. Методами данного интерфейса можно получить:

- указатель на интерфейс локальной системы координат ksPlacement,

▼ указатель на интерфейс ksEntityCollection - массив указанных в процессе объектов.

IRequestInfo – свойства

commandsString – Строка для создания меню командного окна

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetCommandsString и IRequestInfo::SetCommandsString.

Тип данных: строка

Синтаксис Automation:

commandsString	=	Получить свойство (*)
IRequestInfo.commandsString		
IRequestInfo.commandsString	=	Установить свойство (*)
commandsString		
commandsString	=	Получить свойство (**)
IRequestInfo.GetCommandsString()		
IRequestInfo.SetCommandsString(com		Установить свойство (**)
mandsString)		

cursorId – Идентификатор курсора

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetCursorId и IRequestInfo::SetCursorIdString.

Тип данных: long

Синтаксис Automation:

cursorId = IRequestInfo.cursorId	Получить свойство (*)
IRequestInfo.cursorId = cursorId	Установить свойство (*)
cursorId = IRequestInfo.GetCursorId()	Получить свойство (**)
IRequestInfo.SetCursorId(cursorId)	Установить свойство (**)

Примечание:

Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (ldefin2d.h):

OCR_SELECT

OCR_SNAP

OCR_DEFAULT

OCR_CATCH

cursorName – Имя курсора

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetCursorName и IRequestInfo::SetCursorNameString.

Тип данных: строка

Синтаксис Automation:

cursorName = IRequestInfo.cursorName	Получить свойство (*)
IRequestInfo.cursorName = cursorName	Установить свойство (*)
cursorName = IRequestInfo.GetCursorName()	Получить свойство (**)
IRequestInfo.SetCursorName(cursorName)	Установить свойство (**)

Примечание:

Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (Idefin2d.h):

OCR_SELECT

OCR_SNAP

OCR_DEFAULT

OCR_CATCH

DynamicFiltering – Режим динамической фильтрации

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetDynamicFiltering и IRequestInfo::SetDynamicFiltering.

Тип данных: BOOL

Значения свойства:

TRUE	- режим динамической фильтрации включен.

Синтаксис Automation:

DynamicFiltering	=	Получить
iObject.DynamicFiltering		свойство (*)
iObject.DynamicFiltering	=	Установить
DynamicFiltering		свойство (*)
DynamicFiltering	=	Получить
iObject.GetDynamicFiltering()		свойство (**)
iObject.SetDynamicFiltering		Установить
(DynamicFiltering)		свойство (**)

Синтаксис COM:

DynamicFiltering	=	Получить
iObject.GetDynamicFiltering()		свойство (*)
iObject.SetDynamicFiltering		Получить
(DynamicFiltering)		свойство (**)

Примечания:

Свойство распространяется на работу процесса указания местоположения или объектов ksDocument3D::UserGetPlacementAndEntity.

Перед запуском процесса определяются параметры запроса к системе - интерфейс ksRequestInfo3D, в котором можно определить функцию фильтрации.

Если свойство DynamicFiltering, определяющее режим динамической фильтрации, равно FALSE (умолчательный режим), то функция фильтрации вызывается только когда курсор проходит над объектом, чтобы библиотека могла сообщить системе подходит данный объект или нет. Если функция фильтрации возвращает TRUE, объект библиотеке подходит.

Если свойство DynamicFiltering равно TRUE, то функция фильтрации вызывается не только когда курсор проходит над объектом, но и когда объекта под курсором нет. В таком случае передается сигнал библиотеке, что объекта под курсором нет. Возвращаемое значение функции фильтрации игнорируется.

Если свойство включено, то метод фильтрации объектов, заданный функцией ksRequestInfo3D::SetFilterCallBack, будет вызываться динамически, не зависимо от того установлен курсор над объектом или нет.

Если свойство выключено, то метод будет вызываться, только если курсор находится над объектом.

menuId – Идентификатор меню

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetMenuId и IRequestInfo::SetMenuId.

Тип данных: long

Синтаксис Automation:

menuId = IRequestInfo.menuId	Получить свойство (*)
IRequestInfo.menuId = menuId	Установить свойство (*)
menuId = IRequestInfo.GetMenuId()	Получить свойство (**)
IRequestInfo.SetMenuId (menuId)	Установить свойство (**)

processParam – Параметры процесса

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetProcessParam и IRequestInfo::SetProcessParam.

Тип данных: указатель на интерфейс IProcessParam

Синтаксис Automation:

processParam iObject.ProcessParam	=	Получить свойство (*)
iObject.ProcessParam processParam	=	Установить свойство (*)
processParam iObject.GetProcessParam()	=	Получить свойство (**)
iObject.SetProcessParam (processParam)		Установить свойство (**)

Примечания:

Если в интерфейсе параметров запроса к системе указатель на интерфейс параметров процесса не NULL, то при выполнении процесса UserGetPlacementAndEntity, элементы управления из IProcessParam попадут на Панель свойств процесса.

prompt – Строка–подсказка процесса

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetPrompt и IRequestInfo::SetPrompt.

Тип данных: строка

Синтаксис Automation:

prompt = IRequestInfo.prompt	Получить свойство (*)
IRequestInfo.prompt = prompt	Установить свойство (*)

prompt = IRequestInfo.GetPrompt()	Получить свойство (**)
IRequestInfo.SetPrompt(prompt)	Установить свойство (**)

selectionBandMode – Режим использования прямоугольной рамки для выделения объектов

Интерфейс...

Тип данных: из перечисления ksSelectionBandMode

Синтаксис Automation:

selectionBandMode = Object.selectionBandMode	=	Получить свойство (*)
Object.selectionBandMode = selectionBandMode	=	Установить свойство (*)
selectionBandMode = Object.GetSelectionBandMode()	=	Получить свойство (**)
Object.SetSelectionBandMode(selectionBandMode)		Установить свойство (**)

ShowCommandWindow – Показывать командное окно

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

ShowCommandWindow = Object.ShowCommandWindow	=	Получить свойство (*)
Object.ShowCommandWindow = ShowCommandWindow	=	Установить свойство (*)
ShowCommandWindow = Object.GetShowCommandWindow()	=	Получить свойство (**)
Object.SetShowCommandWindow(ShowCommandWindow)		Установить свойство (**)

Синтаксис COM:

ShowCommandWindow = Object.GetShowCommandWindow();	=	Получить свойство

Object.SetShowCommandWindow(ShowCommandWindow);	Установить свойство

title – Строка–заголовок командного окна процесса

Интерфейс...

Аналог данного свойства при использовании COM - методы IRequestInfo::GetTitle и IRequestInfo::SetTitle.

Тип данных: строка

Синтаксис Automation:

title = IRequestInfo.title	Получить свойство (*)
IRequestInfo.title = title	Установить свойство (*)
title = IRequestInfo.GetTitle()	Получить свойство (**)
IRequestInfo.SetTitle(title)	Установить свойство (**)

IRequestInfo – методы

GetProcessingGroupObjectsCallBack – Получить имя (в Automation) или адрес (в COM) функции обратной связи для обработки объектов, пришедших при селектировании рамкой

Интерфейс...

Синтаксис Automation:

BSTR GetProcessingGroupObjectsCallBack();

Возвращаемое значение:

- имя функции.

Синтаксис COM:

void* GetProcessingGroupObjectsCallBack();

Возвращаемое значение:

- адрес функции.

GetObjectsFilter3D – Способ фильтрации 3D объектов в процессе

Интерфейс...

Синтаксис Automation:

BOOL GetObjectsFilter3D(long filterType);

Синтаксис COM :

BOOL GetObjectsFilter3D(long filterType);

Входные параметры:

filterType	- тип объектов для фильтрации в процессе из перечисления ksProcessObjectsFilter3DEnum.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

CreatePhantom – Создать фантом

Интерфейс...

Синтаксис Automation:

BOOL CreatePhantom();

Синтаксис COM:

BOOL CreatePhantom();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

GetCallBack – Получить имя (в Automation) или адрес (в COM) функции обратной связи процесса

Интерфейс...

Синтаксис Automation:

BSTR GetCallBack();

Возвращаемое значение:

- имя функции.

Синтаксис COM:

USERSELECTCALLBACKPROC GetCallBack();

Возвращаемое значение:

- адрес функции.

GetCallBackFeature – Получить интерфейс на объект дерева в функции обратной связи для процесса ksDocument3D_UserGetPlacementAndEntity

Интерфейс...

Получить интерфейс на объект дерева в функции обратной связи для процесса ksDocument3D_UserGetPlacementAndEntity.

Синтаксис Automation:

LPDISPATCH GetCallBackFeature();

Синтаксис COM:

LPFEATURE GetCallBackFeature();

Возвращаемое значение:

- указатель на интерфейс объекта дерева ksFeature или IFeature.

GetCommandsString – Получить строку меню, если меню было задано строкой

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::Prompt.

Синтаксис COM:

LPOLESTR GetCommandsString();

Возвращаемое значение:

- строка меню.

GetCurrentCommand – Получить номер текущей команды из командного окна

Интерфейс...

Синтаксис Automation:

long GetCurrentCommand();

Синтаксис COM:

long GetCurrentCommand();

Возвращаемое значение:

- номер текущей команды.

GetCursorId – Получить идентификатор курсора

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::CursorId.

Синтаксис COM:

long GetCursorId();

Возвращаемое значение:

- идентификатор курсора.

GetCursorName – Получить имя курсора

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::CursorName.

Синтаксис COM:

LPOLESTR GetCursorName();

Возвращаемое значение:

- имя курсора.

GetDynamicFiltering – Получить свойство режима динамической фильтрации

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::DynamicFiltering.

Синтаксис COM:

GetDynamicFiltering();

Возвращаемое значение:

TRUE	- режим динамической фильтрации включен,
FALSE	- режим динамической фильтрации выключен.

Примечания:

Метод позволяет получить флаг режима динамической фильтрации (TRUE - включен).

Метод распространяется на работу процесса указания местоположения или объектов IDocument3D::UserGetPlacementAndEntity.

Перед запуском процесса определяются параметры запроса к системе - интерфейс IRequestInfo, в котором можно определить функцию фильтрации.

Если свойство DynamicFiltering, определяющее режим динамической фильтрации, равно FALSE (умолчательный режим), то функция фильтрации вызывается только когда курсор проходит над объектом, чтобы библиотека могла сообщить системе, подходит данный объект или нет. Если функция фильтрации возвращает TRUE, объект библиотеке подходит.

Если свойство DynamicFiltering равно TRUE, то функция фильтрации вызывается не только когда курсор проходит над объектом, но и когда объекта под курсором нет. В таком случае передается сигнал библиотеке, что объекта под курсором нет. Возвращаемое значение функции фильтрации игнорируется.

GetEntityCollection – Получить указатель на интерфейс массива объектов, указанных в процессе

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetEntityCollection();

Синтаксис COM:

LPENTITYCOLLECTION GetEntityCollection();

Возвращаемое значение:

- указатель на интерфейс ksEntityCollection или IEntityCollection.

Примечание:

В возвращаемом массиве контроль выключен, что позволяет добавлять в массив нулевые указатели.

GetFilterCallBack – Получить имя (в Automation) или адрес (в COM) функции обратной связи для фильтрации объектов

Интерфейс...

Синтаксис Automation:

BSTR GetFilterCallBack();

Возвращаемое значение:

- имя функции.

Синтаксис COM:

USERSELECTFILTERPROC GetFilterCallBack();

Возвращаемое значение:

- адрес функции.

GetMateConstraintCollection – Получить указатель на интерфейс массива временных сопряжений для фантома

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMateConstraintCollection();

Синтаксис COM:

LPMATECONSTRAINTCOLLECTION GetMateConstraintCollection();

Возвращаемое значение:

- указатель на интерфейс ksMateConstraintCollection или ImateConstraintCollection.

GetMenuId – Получить идентификатор меню

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::MenuId.

Синтаксис COM:

long GetMenuId();

Возвращаемое значение:

- идентификатор меню.

GetIPhantom – Получить указатель на интерфейс фантома

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetIPhantom();

Синтаксис COM:

LPPART GetIPhantom();

Возвращаемое значение:

- указатель на интерфейс ksPart или IPart.

GetPlacement – Получить указатель на интерфейс локальной системы координат

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

GetProcessParam – Получить указатель на интерфейс параметров процесса

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::processParam.

Синтаксис COM:

LPUNKNOWN GetProcessParam();

Возвращаемое значение:

- Указатель на интерфейс IProcessParam.

Примечания:

Если в интерфейсе параметров запроса к системе указатель на интерфейс параметров процесса не NULL, то при выполнении процесса UserGetPlacementAndEntity элементы управления из IProcessParam попадут на Панель свойств процесса.

GetPrompt – Получить строку-подсказку процесса

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::Prompt.

Синтаксис COM:

GetPrompt();

Возвращаемое значение:

- строка подсказки.

GetTitle – Получить заголовок

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::title.

Синтаксис COM:

LPOLESTR GetTitle();

Возвращаемое значение:

- заголовок.

SetCallBack – Изменить имя (в Automation) или адрес (в COM) функции обратной связи процесса

Интерфейс...

Синтаксис Automation:

BOOL SetCallBack (BSTR methodName,

long hInst,

LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- имя функции,
hInst	- HINSTANCE модуля, в котором находится функция,
dispatchOCX	- интерфейс, в котором находится функция.

Синтаксис COM:

BOOL SetCallBack (USERSELECTCALLBACKPROC callBack);

Входной параметр:

callBack - адрес функции.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Тип функции обратной связи для выбора объекта:

typedef int (__stdcall * USERSELECTCALLBACKPROC)(LPENTITY, LPREQUESTINFO);

Прототип CallBack-функции:

Синтаксис Automation (пример для Visual C):

int WINAPI SELECTCALLBACKPROC (LPDISPATCH entity,

LPDISPATCH info);

Синтаксис COM:

int __stdcall SelectCallBackProc (LPENTITY entity, LPREQUESTINFO info);

Входные параметры:

entity	- указатель на интерфейс ksEntity или IEntity,
info	- указатель на интерфейс kIsRequestInfo3D или IRequestInfo.

Функция вызывается по нажатию кнопки мыши на подсвеченном объекте.

Получить доступ к массиву объектов, указываемых в процессе, можно вызовом метода GetEntityCollection у параметра info.

Если входной параметр entity равен 0, то в процессе была указана точка и рекомендуется заполнить массив нулевыми указателями.

Если entity не равен 0, то при необходимости можно добавить данный объект в массив.

CallBack-функция должна вернуть 1, если содержимое массива выбранных элементов было изменено (в этом случае будет обновлена подсветка выбранных объектов в модели), и 0, если массив не изменился (в модели никаких изменений не произойдет).

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64.Рекомендуется использовать функцию ksRequestInfo3D::SetCallBackEx.

SetCallBackEx – Установить функцию обратной связи

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackEx(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Аналог функции в COM см. IRequestInfo3D::SetCallBack.

Прототип FilterCallBack-функции:

Синтаксис Automation (пример для Visual C):

int WINAPI SELECTCALLBACKPROC(LPDISPATCH entity, LPDISPATCH info);

Входной параметр:

entity	- указатель на интерфейс текущего элемента или объекта ksEntity,
info	- указатель на интерфейс ksRequestInfo3D.

Примечание:

Функция вызывается по нажатию кнопки мыши на подсвеченном объекте.

Получить доступ к массиву объектов, указываемых в процессе, можно вызовом метода GetEntityCollection у параметра info.

- Если входной параметр entity равен 0, то в процессе была указана точка и рекомендуется заполнить массив нулевыми указателями.
- Если entity не равен 0, то при необходимости можно добавить данный объект в массив.

CallBack-функция должна вернуть:

- 1, если содержимое массива выбранных элементов было изменено (в этом случае будет обновлена подсветка выбранных объектов в модели).
- 0, если массив не изменился (в модели никаких изменений не произойдет).

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksRequestInfo3D::SetCallBack.

SetCommandsString – Установить строку меню, если меню было задано строкой

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::Prompt.

Синтаксис COM:

BOOL SetCommandsString(LPOLESTR menu);

Входные параметры:

menu	- строка меню.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FALSE - в случае неудачи.

SetCursorId – Установить идентификатор курсора

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::CursorId.

Синтаксис COM:

BOOL SetCursorId(long cursorId);

Входные параметры:

cursorId - идентификатор курсора.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetCursorName – Установить имя курсора

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::CursorName.

Синтаксис COM:

BOOL SetCursorName(LPOLESTR cursor);

Входные параметры:

cursor - имя курсора.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetCursorText – Установить текст курсора

Интерфейс...

Синтаксис COM:

BOOL SetCursorText(BSTR Text);

Примечание

Функция работает при запущенном процессе.

Текст сразу выдается над курсором.

SetDynamicFiltering – Установить свойство режима динамической фильтрации

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::DynamicFiltering.

Синтаксис COM:

SetDynamicFiltering(BOOL f);

Входные параметры:

f - (BOOL) - флаг режима динамической фильтрации (TRUE - включить).

Возвращаемое значение:

TRUE - режим динамической фильтрации изменен.

Примечания:

Метод позволяет установить флаг режима динамической фильтрации (TRUE - включен).

Метод распространяется на работу процесса указания местоположения или объектов IDocument3D::UserGetPlacementAndEntity.

Перед запуском процесса определяются параметры запроса к системе - интерфейс IRequestInfo, в котором можно определить функцию фильтрации.

Если свойство DynamicFiltering, определяющее режим динамической фильтрации, равно FALSE (умолчательный режим), то функция фильтрации вызывается только когда курсор проходит над объектом, чтобы библиотека могла сообщить системе, подходит данный объект или нет. Если функция фильтрации возвращает TRUE, объект библиотеке подходит.

Если свойство DynamicFiltering равно TRUE, то функция фильтрации вызывается не только когда курсор проходит над объектом, но и когда объекта под курсором нет. В таком случае передается сигнал библиотеке, что объекта под курсором нет. Возвращаемое значение функции фильтрации игнорируется.

SetFilterCallBack – Изменить имя (в Automation) или адрес (в COM) функции обратной связи для фильтрации объектов

Интерфейс...

Синтаксис Automation:

BOOL SetFilterCallBack (BSTR methodName,

long hInst,

LPDISPATCH dispatchOCX);

Входные параметры:

methodName - имя функции,
hInst - HINSTANCE модуля, в котором находится функция,
dispatchOCX - интерфейс, в котором находится функция.

Синтаксис COM:

BOOL SetFilterCallBack (USERSELECTFILTERPROC callBack);

Входной параметр:

callBack - адрес функции.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Тип функции обратной связи для фильтрации объекта (функции-«фильтра»)

typedef BOOL (__stdcall * USERSELECTFILTERPROC)(LPENTITY);

Прототип FilterCallBack-функции:

Синтаксис Automation (пример для Visual C):

BOOL WINAPI SELECTFILTERPROC (LPDISPATCH _entity);

Синтаксис COM:

BOOL __stdcall SelectFilterProc (LPENTITY entity);

Входной параметр:

entity - указатель на интерфейс текущего элемента или объекта ksEntity или IEntity.

Возвращаемое значение:

TRUE - объект будет подсвечен.

В данной функции можно отобрать интересующие пользователя элементы или объекты. Функция вызывается при изменении положения курсора, если курсор находится рядом с элементом или объектом модели.

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64.

Рекомендуется использовать функцию ksRequestInfo3D::SetFilterCallBackEx.

SetFilterCallBackEx – Установить функцию обратной связи для фильтрации объектов

Интерфейс...

Синтаксис Automation:

BOOL SetFilterCallBackEx(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реали-зована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Аналог функции в COM, см. IRequestInfo3D::SetFilterCallBack.

Прототип FilterCallBack-функции:

Синтаксис Automation (пример для Visual C):

BOOL WINAPI SELECTFILTERPROC (LPDISPATCH _entity);

Входной параметр:

entity	- указатель на интерфейс текущего элемента или объекта ksEntity или IEntity.

Примечание:

В данной функции можно отобрать интересующие пользователя элементы или объекты. Функция вызывается при изменении положения курсора, если курсор находится рядом с элементом или объектом модели.

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksRequestInfo3D::SetFilterCallBack.

SetMenuId – Установить идентификатор меню

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::MenuId.

Синтаксис COM:

BOOL SetMenuId(long menuId);

Входные параметры:

menuId - идентификатор меню.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetObjectsFilter3D – Способ фильтрации 3D объектов в процессе

Интерфейс...

Синтаксис Automation:

BOOL SetObjectsFilter3D(long filterType, BOOL newVal);

Синтаксис COM :

BOOL SetObjectsFilter3D(long filterType, BOOL newVal);

Входные параметры:

filterType - тип объектов для фильтрации в процессе из перечисления ksProcessObjectsFilter3DEnum.
newVal - признак фильтрации

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetProcessingGroupObjectsCallBack – Установить имя (в Automation) или адрес (в COM) функции обратной связи для обработки объектов, пришедших при селектировании рамкой

Интерфейс...

Синтаксис Automation :

BOOL SetProcessingGroupObjectsCallBack(BSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName - имя функции,
hInst - HINSTANCE модуля, в котором находится функция,
dispatchOCX - интерфейс, в котором находится функция.

Синтаксис COM :

BOOL SetProcessingGroupObjectsCallBack(void* callBack);

Входной параметр:

callBack - адрес функции.

Возвращаемое значение

TRUE - в случае успешного завершения.

Прототип CallBack-функции:

Синтаксис Automation (пример для Visual C):

BOOL WINAPI USERPROCESSINGGROUPOBJECTS(VARIANT * Objects, long selectionType);

Синтаксис COM:

BOOL __stdcall USERPROCESSINGGROUPOBJECTS(VARIANT * Objects, long selectionType);

Входные параметры:

Objects	- список объектов массив SafeArray тип VT_ARRAY | VT_DISPATCH в автоматизации; VT_ARRAY | VT_UNKNOWN в COM.
selectionType	- Тип селектирования из перечисления.

Возвращаемое значение

- не используется.

Примечание

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

SetProcessParam – Установить указатель на интерфейс параметров процесса

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::processParam.

Синтаксис COM:

BOOL SetProcessParam(LPUNKNOWN param);

Входные параметры:

param - указатель на интерфейс IProcessParam.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Если в интерфейсе параметров запроса к системе указатель на интерфейс параметров процесса не NULL, то при выполнении процесса UserGetPlacementAndEntity элементы управления из IProcessParam попадут на Панель свойств процесса.

SetPrompt – Установить строку-подсказку процесса

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::Prompt.

Синтаксис COM:

BOOL SetPrompt(LPOLESTR prompt);

Входные параметры:

prompt - строка подсказки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetTitle – Установить заголовок

Интерфейс...

Аналог данного метода при использовании Automation - свойство ksRequestInfo3D::title

Синтаксис COM:

BOOL SetTitle(LPOLESTR title);

Входные параметры:

title - заголовок.

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Положение объекта (Интерфейсы ksPlacement, IPlacement)

Интерфейс локальной системы координат (положение объекта).

ksPlacement - интерфейс Automation
IPlacement - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов ksRequestInfo3D::GetPlacement, ksObject3DNotifyResult::GetPlacement, ksPlaneParam::GetPlacement, ksTorusParam::GetPlacement, ksEllipse3dParam::GetPlacement, ksCylinderParam::GetPlacement, ksArc3dParam::GetPlacement, ksConeParam::GetPlacement, ksDocument3D::DefaultPlacement, ksViewProjection::GetPlacement, ksCircle3dParam::GetPlacement, ksPart::GetPlacement, ksSphereParam::GetPlacement.

IPlacement – методы

GetAxis – Получить компоненты вектора направления оси OX, OY или OZ локальной системы координат

Интерфейс...

Синтаксис Automation:

BOOL GetAxis (double* x,

double* y,

double* z,

long type);

Синтаксис COM:

BOOL GetAxis (double *x,

double *y,

double *z,

long type);

Входной параметр:

type - тип оси:
0 – OX,
1– OY,
>1– OZ.

Выходные параметры:

x, y, z - компоненты вектора направления оси.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Положение начала локальной системы координат можно получить при помощи метода GetOrigin.

GetMatrix– Получить суммарную матрицу преобразования координат

Интерфейс...

Синтаксис Automation:

VARIANT GetMatrix();

Синтаксис COM:

HRESULT GetMatrix(VARIANT * Result);

Возвращаемое значение:

- массив SafeAttray типа (VT_ARRAY | VT_R8).

Примечание:

1. Элементы матрицы возвращаются в виде одномерного массива из 16 элементов.
2. Матрица имеет размер 4x4.

GetOrigin – Получить координаты начала локальной системы координат

Интерфейс...

Синтаксис Automation:

BOOL GetOrigin (double* x,
double* y,
double* z);

Синтаксис COM:

BOOL GetOrigin (double *x,
double *y,
double *z);

Выходные параметры:

x, y, z - координаты начала локальной системы координат в глобальной системе координат.

Возвращаемое значение:

TRUE - в случае успешного завершения.

Примечание:

Направление осей локальной системы координат можно получить при помощи метода GetAxis.

GetVector – Получить вектор для указанной оси

Интерфейс...

Синтаксис Automation:

BOOL GetVector(long type, double * X, double * Y, double * Z);

Синтаксис COM:

BOOL GetVector(long type, double * X, double * Y, double * Z);

Возвращаемое значение:

TRUE - в случае удачи.

Входные параметры:

type - тип оси:
0 – OX,
1 – OY,
>1 – OZ.

Выходные параметры:

X, Y, Z - компоненты вектора направления оси.

InitByMatrix3D – Установить систему координат по матрице

Интерфейс...

Синтаксис Automation:

BOOL InitByMatrix3D(VARIANT * mtr);

Синтаксис COM:

BOOL InitByMatrix3D(VARIANT * mtr);

Входные параметры:

mtr - массив SafeAttray типа (VT_ARRAY | VT_R8).

Примечание:

1. Элементы матрицы возвращаются в виде одномерного массива из 16 элементов.
2. Матрица имеет размер 4x4.

PointOn– Получить пространственную точку по точке на плоскости XY

Интерфейс...

Синтаксис Automation:

BOOL PointOn (double XIn, double YIn, double * XOut, double * YOut, double * ZOut);

Синтаксис COM:

BOOL PointOn (double XIn, double YIn, double * XOut, double * YOut, double * ZOut);

Входные параметры:

XIn, YIn - координаты исходной точки на плоскости XY.

Выходные параметры:

XOut, YOut, ZOut - координаты пространственной точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

PointProjection – Получить проекцию точки на плоскость XY

Интерфейс...

Синтаксис Automation:

BOOL PointProjection (double XIn, double YIn, double ZIn, double * XOut, double * YOut);

Синтаксис COM:

BOOL PointProjection (double XIn, double YIn, double ZIn, double * XOut, double * YOut);

Входные параметры:

XIn, YIn, ZIn - координаты исходной пространственной точки.

Выходные параметры:

XOut, YOut - координаты точки на плоскости XY.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Rotate – Получить последний объект в дереве

Интерфейс...

Синтаксис Automation:

BOOL Rotate(double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle);

Синтаксис COM:

BOOL Rotate(double X0, double Y0, double Z0, double AxisZX, double AxisZXY, double AxisZZ, double Angle);

Возвращаемое значение:

TRUE	- в случае успешного завершения,

SetAxes – Установить оси X и Y

Пример...

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetAxes (double Xx,
double Xy,
double Xz,
double Yx,
double Yy,
double Yz);

Синтаксис COM:

BOOL SetAxes (double Xx,
double Xy,
double Xz,
double Yx,
double Yy,
double Yz)

Входные параметры:

Xx, Xy, Xz	- координаты точки на оси X,

Yx, Yy, Yz - координаты точки на оси Y.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetAxis – Установить компоненты вектора направления оси OX, OY или OZ локальной системы координат

Пример...

Интерфейс...

Синтаксис Automation:

BOOL SetAxis (double x,

double y,

double z,

long type);

Синтаксис COM:

BOOL SetAxis (double x,

double y,

double z,

int type);

Входные параметры:

x, y, z - компоненты вектора направления оси,
type - тип оси:
0 – OX,
>0 – OY.

Возвращаемое значение:

TRUE - в случае успешного завершения.

SetOrigin – Изменить координаты начала локальной системы координат

Пример...

Интерфейс...

Синтаксис Automation:

BOOL SetOrigin (double x,

double y,

double z);

Синтаксис COM:

BOOL SetOrigin (double x,

double y,

double z);

Входные параметры:

x, y, z	- координаты начала локальной системы координат в глобальной системе координат.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetPlacement – Изменить локальную систему координат

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement (LPDISPATCH placement);

Синтаксис COM:

BOOL SetPlacement (LPPLACEMENT placement);

Входной параметр:

placement	- указатель на интерфейс ksPlacement или IPlacement.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

SetVector – Задать вектор для указанной оси

Интерфейс...

Синтаксис Automation:

BOOL SetVector(long type, double * X, double * Y, double * Z);

Синтаксис COM:

BOOL SetVector(long type, double * X, double * Y, double * Z);

Возвращаемое значение:

TRUE	- в случае удачи.

Входные параметры:

type - тип оси:

0 – OX,
1 – OY,
>1 – OZ.

Выходные параметры:

X, Y, Z - компоненты вектора направления оси.

Массив объектов модели

Интерфейсы ksEntityCollection и IEntityCollection

Объект модели

Интерфейсы ksEntity и IEntity

Интерфейсы пространственных кривых

Интерфейсы поверхностей

Интерфейсы копирования

Интерфейсы копирования компонентов сборки

Интерфейсы формообразующих операций

Интерфейсы дополнительных элементов

Интерфейсы вспомогательной геометрии

Массив сопряжений (Интерфейсы ksMateConstraintCollection, IMateConstraintCollection)

Справка системы КОМПАС...

kompas.chm::/1024_112_1_Obshchije_sved_sopr.htm

Интерфейс параметров массива сопряжений компонента сборки.

ksMateConstraintCollection - интерфейс Automation
IMateConstraintCollection - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksRequestInfo3D::GetMateConstraintCollection.

IMateConstraintCollection – методы

AddMateConstraint – Добавить сопряжение в массив

Интерфейс...

Синтаксис Automation:

BOOL AddMateConstraint (LPDISPATCH mate);

Синтаксис COM:

BOOL AddMateConstraint (LPMATECONSTRAINT mate);

Входные параметры:

mate	- указатель на интерфейс ksMateConstraint или IMateConstraint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Clear– Очистить массив сопряжений

Интерфейс...

Синтаксис Automation:

BOOL Clear();

Синтаксис COM:

BOOL Clear();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Объекты удаляются только из массива, в модель изменения не передаются.

GetByIndex – Получить указатель на интерфейс объекта с указанным индексом в массиве сопряжений

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPMATECONSTRAINT GetByIndex (long index);

Входной параметр:

index - индекс сопряжения в массиве.

Возвращаемое значение:

- указатель на интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.

GetCount – Получить количество элементов в массиве сопряжений

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

unsigned long GetCount();

Возвращаемое значение:

- количество элементов в массиве сопряжений.

GetSafeArrayByObj – Сформировать массив SAFEARRAY параметров сопряжений по объекту

Интерфейс...

Синтаксис Automation:

BOOL GetSafeArrayByObj(LPDISPATCH obj, VARIANT * pArray);

Синтаксис COM:

BOOL GetSafeArrayByObj(LPUNKNOWN obj, VARIANT * pArray);

Входные параметры:

obj - указатель на объект.

Выходные параметры:

pArray - указатель на массив параметров сопряжений.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Тип выходного массива VT_ARRAY | VT_UNKNOWN. Элемент, лежащий в массиве, IMateConstraint.
2. Входной объект либо компонент IPart, тогда будет сформирован массив всех сопряжений для этого компонента, либо объект модели, который может фигурировать в сопряжениях.

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt (LPDISPATCH entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Синтаксис COM:

unsigned long FindIt (LPMATECONSTRAINT entity);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
SYS_MAX_UINT	- если элемент не найден.

Входные параметры:

entity	- указатель на интерфейс ksMateConstraint или IMateConstraint.

First – Получить указатель на интерфейс первого объекта в массиве сопряжений

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPMATECONSTRAINT First();

Возвращаемое значение:

- указатель на интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.

Last – Получить указатель на интерфейс последнего объекта в массиве сопряжений

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPMATECONSTRAINT Last();

Возвращаемое значение:

- указатель на интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.

Next – Получить указатель на интерфейс следующего объекта в массиве сопряжений

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPMATECONSTRAINT Next();

Возвращаемое значение:

- указатель на интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.

Prev – Получить указатель на интерфейс предыдущего объекта в массиве сопряжений

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPMATECONSTRAINT Prev();

Возвращаемое значение:

- указатель на интерфейс параметров сопряжения ksMateConstraint или IMateConstraint.

Refresh – Обновить массив сопряжений

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод не работает для временного массива (полученного от ksRequestInfo3D::GetMateConstraintCollection).

RemoveMateConstraint – Удалить сопряжение из массива

Интерфейс...

Синтаксис Automation:

BOOL RemoveMateConstraint (LPDISPATCH mate);

Синтаксис COM:

BOOL RemoveMateConstraint (LPMATECONSTRAINT mate);

Входные параметры:

mate	- указатель на интерфейс ksMateConstraint или IMateConstraint.

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Интерфейсы структуры параметров сопряжения (Интерфейсы ksMateConstraint, IMateConstraint)

Справка системы КОМПАС...

`kompas.chm::/1024_112_1_Obshchije_sved_sopr.htm`

Интерфейс параметров сопряжения.

ksMateConstraint	- интерфейс Automation
IMateConstraint	- интерфейс COM

Примечание:

Данный интерфейс может быть получен следующими способами:

- с использованием метода интерфейса ksDocument3D::GetMateConstraint,

▼ с использованием методов интерфейса ksMateConstraintCollection.

IMateConstraint – свойства

constraintType – Тип сопряжения

Интерфейс...

Тип данных: short

Синтаксис Automation:

constraintType = mateConstraint.constraintType	=	Получить свойство(*)
mateConstraint.constraintType = constraintType	=	Установить свойство (*)
constraintType = mateConstraint.GetConstraintType()	=	Получить свойство (**)
mateConstraint.SetConstraintType(constraintType)		Установить свойство (**)

Синтаксис COM:

constraintType = mateConstraint.GetConstraintType()	=	Получить свойство
mateConstraint.SetConstraintType(constraintType)		Установить свойство

Типы сопряжений...

direction – Направление

Интерфейс...

Тип данных: short

Синтаксис Automation:

direction = mateConstraint.direction	=	Получить свойство(*)
mateConstraint.direction = direction	=	Установить свойство (*)
direction = mateConstraint.GetDirection()	=	Получить свойство (**)
mateConstraint.SetDirection(direction)		Установить свойство (**)

Синтаксис COM:

direction = mateConstraint.GetDirection()	Получить свойство
mateConstraint.SetDirection(direction)	Установить свойство

Направления сопряжений...

distance – Расстояние между объектами

Интерфейс...

Тип данных: double

Синтаксис Automation:

distance = mateConstraint.distance	Получить свойство(*)
mateConstraint.distance = distance	Установить свойство (*)
distance = mateConstraint.GetDistance( )	Получить свойство (**)
mateConstraint.SetDistance(distance)	Установить свойство (**)

Синтаксис COM:

distance = mateConstraint.GetDistance()	Получить свойство
mateConstraint.SetDistance(distance)	Установить свойство

fixed – Фиксация

Интерфейс...

Тип данных: short

Синтаксис Automation:

fixed = mateConstraint.fixed	Получить свойство(*)
mateConstraint.fixed = fixed	Установить свойство (*)
fixed = mateConstraint.GetFixed()	Получить свойство (**)
mateConstraint.SetFixed(fixed)	Установить свойство (**)

Синтаксис COM:

fixed	=	Получить свой-
mateConstraint.GetFixed()		ство
mateConstraint.SetFixed(fixed		Установить
)		свойство

Фиксации сопряжений...

Примечание:

В настоящее время не используется.

IMateConstraint – методы

Create – Создать временное сопряжение

Интерфейс...

Синтаксис Automation:

BOOL Create();

Синтаксис COM:

BOOL Create();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Сопряжения бывают постоянными и временными. Временные сопряжения, в отличие от постоянных, существуют только в течении процесса указания местоположения или объектов. Процесс запускается методом ksDocument3D::UserGetPlacementAndEntity. Постоянные сопряжения создаются с помощью метода ksDocument3D::AddMateConstraint.
2. Данный метод необходимо вызвать для создания временного сопряжения в модели после создания интерфейса с помощью метода ksDocument3D::GetMateConstraint и изменения параметров создаваемого сопряжения. Если до вызова данного метода параметры не изменены, сопряжение создается с параметрами, принятыми по умолчанию.
3. Пример использования методов создания временных и постоянных сопряжений приведен в демонстрационной библиотеке построения модели стандартного изделия (шпильки), которая сохранена в файле STUDS3D.rtw.

GetBaseObj – Получить указатель на интерфейс объекта для сопряжения по номеру

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetBaseObj (short number);

Синтаксис COM:

LPENTITY GetBaseObj (short number);

Входные параметры:

number - номер объекта для сопряжения (1 или 2).

Возвращаемое значение:

- указатель на интерфейс объекта для сопряжения ksEntity или IEntity.

GetEntityParams – Получить параметры математических обектов, участвующих в сопряжении

Интерфейс...

Синтаксис Automation:

long GetEntityParams (long index, VARIANT * params);

Синтаксис COM:

int GetEntityParams (int index, VARIANT * params);

Входные параметры:

index - индекс объекта.

Выходные параметры Automation:

params - параметры объекта массив SafeArray вещественных чисел.

Выходные параметры COM:

params - параметры объекта.

Возвращаемое значение:

тип математического объекта, участвующего в сопряжении из ksMateType - если параметры получены,
ksMateUnknown - в случае ошибки.

Примечание:

1. Параметры возвращаются в системе координат сборки. При необходимости их можно перевести в систему координат нужного компонента с помощью функции IPart::TransformPoint.
2. Геометрический объект для сопряжения описывается набором из 8-ми вещественных чисел, сгруппированных следующим образом:

- три координаты точки Pc, Pc = { pointX, pointY, pointZ },
- три координаты вектора V, V = { vectorI, vectorJ, vectorK },

▼ две координаты, соответствующие радиусам, V = { vectorI, vectorJ, vectorK }.

Представление математических объектов, участвующих в сопряжении...

GetFeature – Получить указатель на интерфейс параметров объекта дерева построения, связанного с данным объектом

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFeature();

Синтаксис COM:

LPFEATURE GetFeature();

Возвращаемое значение:

- указатель на интерфейс ksFeature или IFeature.

SetBaseObj – Установить указатель на интерфейс объекта для сопряжения по номеру

Интерфейс...

Синтаксис Automation:

BOOL SetBaseObj (short number, LPDISPATCH obj);

Синтаксис COM:

BOOL SetBaseObj (short number, LPENTITY obj);

Входные параметры:

number	- номер объекта для сопряжения (1 или 2),
obj	- указатель на интерфейс объекта для сопряжения ksEntity или IEntity.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Менеджер выбора (подсветки) объектов (Интерфейсы ksChooseMng, IChooseMng)

Справка системы КОМПАС...

kompas.chm::/98_8_5_Vydelenie_obwektov.htm

Интерфейс менеджера выбора (подсветки) объектов.

ksChooseMng - интерфейс Automation
IChooseMng - интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksDocument3D::GetChooseMng.

IChooseMng– методы

Choose – Выбрать (подсветить) объект

Интерфейс...

Синтаксис Automation:

BOOL Choose (LPDISPATCH obj)

Синтаксис COM:

BOOL Choose (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

First – Получить указатель на интерфейс первого выбранного (подсвеченного) объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPUNKNOWN First();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

GetCount – Получить количество выбранных (подсвеченных) объектов

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

int GetCount();

Возвращаемое значение:

- количество выбранных (подсвеченных) объектов.

GetManagerIndex – Получить индекс менеджера по указателю на выбранный объект

Интерфейс...

Синтаксис Automation:

long GetManagerIndex(LPDISPATCH obj);

Синтаксис COM:

long GetManagerIndex(LPUNKNOWN obj);

Входной параметр:

obj - указатель на объект.

Возвращаемое значение:

Индекс менеджера из перечисления ksChooseManagerTypeEnum

GetObjectByIndex – Получить указатель на интерфейс выделенного (подсвеченного) объекта по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObjectByIndex (long index);

Синтаксис COM:

LPDISPATCH GetObjectByIndex (long index);

Входной параметр:

index - индекс выделенного (подсвеченного) объекта.

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

GetObjectType – Получить тип выделенного (подсвеченного) объекта по индексу

Интерфейс...

Синтаксис Automation:

long GetObjectType (long index);

Синтаксис COM:

int GetObjectType (int index);

Входной параметр:

index - индекс выделенного (подсвеченного) объекта.

Возвращаемое значение:

- тип выделенного (подсвеченного) объекта Obj3dType.

IsChoosen – Получить признак подсвеченности (выбора) объекта

Интерфейс...

Синтаксис Automation:

BOOL IsChoosen (LPDISPATCH obj);

Синтаксис COM:

BOOL IsChoosen (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown.

Возвращаемое значение:

TRUE - объект подсвечен,
FALSE - объект не подсвечен.

Last – Получить указатель на интерфейс последнего выбранного (подсвеченного) объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPUNKNOWN Last();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

Next – Получить указатель на интерфейс следующего выбранного (подсвеченного) объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPUNKNOWN Next();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

Prev – Получить указатель на интерфейс предыдущего выбранного (подсвеченного) объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPUNKNOWN Prev();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

UnChoose – Снять выбор (подсветку) с объекта

Интерфейс...

Синтаксис Automation:

BOOL UnChoose (LPDISPATCH obj);

Синтаксис COM:

BOOL UnChoose (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

UnChooseAll – Снять выбор (подсветку) со всех подсвеченных объектов

Интерфейс...

Синтаксис Automation:

BOOL UnChooseAll();

Синтаксис COM:

BOOL UnChooseAll();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Менеджер выделенных объектов (Интерфейсы ksSelectionMng, ISelectionMng)

Интерфейс событий...

Справка системы КОМПАС...

`kompas.chm::/98_8_5_Vydelenie_obwektov.htm`

Интерфейс менеджера выделенных объектов.

ksSelectionMng - интерфейс Automation
ISelectionMng - интерфейс COM

Примечание:

Указатель на интерфейс может быть получен от интерфейса ksDocument3D, IDocument3D с помощью метода ksDocument3D::GetSelectionMng, IDocument3D::GetSelectionMng.

ISelectionMng – методы

First – Получить указатель на интерфейс первого выделенного объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPUNKNOWN First();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

GetCount – Получить количество выделенных объектов

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

int GetCount();

Возвращаемое значение:

- количество выделенных объектов.

GetObjectByIndex – Получить указатель на интерфейс выделенного (подсвеченного) объекта по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObjectByIndex (long index);

Синтаксис COM:

LPDISPATCH GetObjectByIndex (long index);

Входной параметр:

index - индекс выделенного (подсвеченного) объекта.

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

GetObjectType – Получить тип выделенного (подсвеченного) объекта по индексу

Интерфейс...

Синтаксис Automation:

long GetObjectType (long index);

Синтаксис COM:

int GetObjectType (int index);

Входной параметр:

index	- индекс выделенного (подсвеченного) объекта.

Возвращаемое значение:

- тип выделенного (подсвеченного) объекта Obj3dType.

IsSelected – Определить, выделен ли объект

Интерфейс...

Синтаксис Automation:

BOOL IsSelected (LPDISPATCH obj);

Синтаксис COM:

BOOL IsSelected (LPUNKNOWN obj);

Входной параметр:

obj	- указатель на интерфейс IDispatch или IUnknown выделенного объекта.

Возвращаемое значение:

TRUE	- объект выделен,
FALSE	- объект не выделен.

Last – Получить указатель на интерфейс последнего выделенного объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPUNKNOWN Last();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

Next – Получить указатель на интерфейс следующего выделенного объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPUNKNOWN Next();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

Prev – Получить указатель на интерфейс предыдущего выделенного объекта

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPUNKNOWN Prev();

Возвращаемое значение:

- указатель на интерфейс IDispatch или IUnknown.

Select – Выделить объект

Интерфейс...

Синтаксис Automation:

BOOL Select (LPDISPATCH obj);

Синтаксис COM:

BOOL Select (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Unselect – Снять выделение с объекта

Интерфейс...

Синтаксис Automation:

BOOL Unselect (LPDISPATCH obj);

Синтаксис COM:

BOOL Unselect (LPUNKNOWN obj);

Входной параметр:

obj - указатель на интерфейс IDispatch или IUnknown.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

UnselectAll – Снять выделение со всех объектов

Интерфейс...

Синтаксис Automation:

BOOL UnselectAll();

Синтаксис COM:

BOOL UnselectAll();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Параметры сохранения растра (Интерфейсы ksRasterFormatParam, IRasterFormatParam)

Справка системы КОМПАС...

`KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm`

Интерфейс параметров для записи документа в растровом формате.

ksRasterFormatParam - интерфейс Automation
IRasterFormatParam - интерфейс COM

Аналог данных параметров при использовании API экспортных функций - RasterFormatParam.

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейсов ksSpcDocument::RasterFormatParam, ksDocument3D::RasterFormatParam, ksDocumentTxt::RasterFormatParam, ksDocument2D::RasterFormatParam.

IRasterFormatParam – свойства

colorBPP – Цветность растра

Интерфейс...

Тип данных: short

Значения свойства...

Синтаксис Automation:

colorBPP = iRasterFormatParam.colorBPP		Получить свойство(*)
iRasterFormatParam.colorBPP = colorBPP		Установить свойство (*)
colorBPP	=	Получить
iRasterFormatParam.GetColorBPP()		свойство (**)
iRasterFormatParam.SetColorBPP(colorBPP)		Установить свойство (**)

colorType – Цвет вывода объектов

Интерфейс...

Тип данных: short

Значения свойства...

Синтаксис Automation:

colorType = iRasterFormatParam.colorType		Получить свойство(*)
iRasterFormatParam.colorType = colorType		Установить свойство (*)
colorType	=	Получить
iRasterFormatParam.GetColorType()		свойство (**)
iRasterFormatParam.SetColorType(colorType)		Установить свойство (**)

extResolution – Разрешение растра

Интерфейс...

Тип данных: long

Синтаксис Automation:

extResolution = iRasterFormatParam.extResolution	Получить свойство(*)
iRasterFormatParam.extResolution = extResolution	Установить свойство (*)

extResolution = iRasterFormatParam.GetExtResolution()	Получить свойство (**)
iRasterFormatParam.SetExtResolution(extResolution)	Установить свойство (**)

Примечание:

Значение 0 соответствует текущему экранному разрешению.

extScale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

extScale = iRasterFormatParam.extScale		Получить свойство(*)
iRasterFormatParam.extScale = extScale		Установить свойство (*)
extScale	=	Получить
iRasterFormatParam.GetExtScale()		свойство (**)
iRasterFormatParam.SetExtScale(extScale)		Установить свойство (**)

format – Формат растра

Интерфейс...

Тип данных: short

Значения свойства...

Синтаксис Automation:

format = iRasterFormatParam.format		Получить свойство(*)
iRasterFormatParam.format = format		Установить свойство (*)
format	=	Получить
iRasterFormatParam.GetFormat()		свойство (**)
iRasterFormatParam.SetFormat(format)		Установить свойство (**)

greyScale – Признак сохранения в оттенках серого

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- использовать оттенки серого,
FALSE	- сохранять цветной растр.

Синтаксис Automation:

greyScale = iRasterFormatParam.greyScale	Получить свойство(*)
iRasterFormatParam.greyScale = greyScale	Установить свойство (*)
greyScale = iRasterFormatParam.GetGreyScale()	Получить свойство (**)
iRasterFormatParam.SetGreyScale(greyScale)	Установить свойство (**)

multiPageOutput – Признак сохранения всех листов в одном файле

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- сохранять все листы в одном файле,
FALSE	- сохранять листы в отдельных файлах.

Синтаксис Automation:

multiPageOutput = iRasterFormatParam.multiPageOutput	Получить свойство(*)
iRasterFormatParam.multiPageOutput = multiPageOutput	Установить свойство (*)
multiPageOutput = iRasterFormatParam.GetMultiPageOutput()	Получить свойство (**)
iRasterFormatParam.SetMultiPageOutput(multiPageOutput)	Установить свойство (**)

Примечание:

Свойство используется только для формата TIFF.

onlyThinLine – Признак вывода всех линий тонкими

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- выводить все только тонкими линиями,
FALSE	- использовать установленные для объектов толщины линий.

Синтаксис Automation:

onlyThinLine = iRasterFormatParam.onlyThinLine	Получить свойство(*)
iRasterFormatParam.onlyThinLine = onlyThinLine	Установить свойство (*)
onlyThinLine = iRasterFormatParam.GetOnlyThinLine()	Получить свойство (**)
iRasterFormatParam.SetOnlyThinLine(onlyThinLine)	Установить свойство (**)

pages – Список диапазонов выводимых страниц

Интерфейс...

Тип данных: строка

Значения свойства:

Формат списка - "beg1-end1, beg2-end2, beg3-end3, ...",

где begN - начало N-го интервала, endN - конец N-го интервала.

Например, "1-18, 24-25".

Синтаксис Automation:

pages = iRasterFormatParam.pages		Получить свойство(*)
iRasterFormatParam.pages = pages		Установить свойство (*)
pages	=	Получить
iRasterFormatParam.GetPages()		свойство (**)
iRasterFormatParam.SetPages(pages)		Установить свойство (**)

Примечание:

1. Свойство используется для многолистовых документов (текстовых документов и спецификаций).
2. Если строка пуста, свойство не используется.

rangeIndex – Признак выбора стороны страниц

Интерфейс...

Тип данных: short

Значения свойства:

0	- все страницы,
1	- нечетные страницы,
2	- четные страницы.

Синтаксис Automation:

rangeIndex = iRasterFormatParam.rangeIndex	Получить свойство(*)
iRasterFormatParam.rangeIndex = rangeIndex	Установить свойство (*)
rangeIndex = iRasterFormatParam.GetRangeIndex()	Получить свойство (**)
iRasterFormatParam.SetRangeIndex(rangeIndex)	Установить свойство (**)

Примечание:

Свойство используется для многолистовых документов (текстовых документов и спецификаций).

saveWorkArea – Сохранить отображение рабочей области

Интерфейс...

Тип данных: BOOL

Синтаксис:

saveWorkArea = Object.saveWorkArea	Получить свойство(*)
Object.saveWorkArea = saveWorkArea	Установить свойство (*)
saveWorkArea = Object.GetSaveWorkArea()	Получить свойство (**)
Object.SetSaveWorkArea(saveWorkArea)	Установить свойство (**)

IRasterFormatParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Синтаксис COM:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры записи документа в растровом формате.

Конвертация в дополнительные форматы (Интерфейсы ksAdditionFormatParam, IAdditionFormatParam)

Справка системы КОМПАС...

kompas.chm::/990_Glava123_Obmen_informaciej_.htm

Интерфейс параметров для записи модели в форматы IGES, SAT, XT, X_B, STEP, STL, VRML.

ksAdditionFormatParam	- интерфейс Automation
IAdditionFormatParam	- интерфейс COM

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksDocument3D::AdditionFormatParam.

IAdditionFormatParam – свойства

angle – Максимально допустимое угловое отклонение касательных кривой или нормалей поверхности в соседних точках на расстоянии шага

Интерфейс...

Тип данных: double

Синтаксис Automation:

angle = Object.angle	Получить свойство(*)
Object.angle = angle	Установить свойство (*)
angle = Object.GetAngle()	Получить свойство (**)
Object.SetAngle(angle)	Установить свойство (**)

author – Автор

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

author = Object.author	Получить свойство(*)
Object.author = author	Установить свойство (*)
author = Object.GetAuthor()	Получить свойство (**)
Object.SetAuthor(author)	Установить свойство (**)

comment – Комментарий

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

comment = Object.comment	Получить свойство(*)
Object.comment = comment	Установить свойство (*)
comment = Object.GetComment()	Получить свойство (**)
Object.SetComment(comment)	Установить свойство (**)

createLocalComponents – TRUE – создавать вставки как локальные. FALSE – сохранять вставки в отдельных файлах

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

createLocalComponents = Object.createLocalComponents	=	Получить свойство(*)
Object.createLocalComponents = createLocalComponents	=	Установить свойство (*)

createLocalComponents	=	Получить
Object.GetCreateLocalComponents()		свойство (**)
Object.SetCreateLocalComponents(		Установить
createLocalComponents)		свойство (**)

format – Формат файла для записи модели

Интерфейс...

Тип данных: D3FormatConvType

Значения свойства...

Синтаксис Automation:

format = iAdditionFormatParam.format		Получить свойство(*)
iAdditionFormatParam.format = format		Установить свойство (*)
format	=	Получить
iAdditionFormatParam.GetFormat()		свойство (**)
iAdditionFormatParam.SetFormat(format)		Установить свойство (**)

formatBinary – Признак, определяющий тип файла (двоичный или текстовый)

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- двоичный формат,
FALSE	- текстовый формат.

Синтаксис Automation:

formatBinary = iAdditionFormatParam.formatBinary	Получить свойство(*)
iAdditionFormatParam.formatBinary = formatBinary	Установить свойство (*)
formatBinary = iAdditionFormatParam.GetFormatBinary()	Получить свойство (**)
iAdditionFormatParam.SetFormatBinary(formatBinary)	Установить свойство (**)

Примечание:

Свойство используется при записи в форматы XT, X_B, STL.

length – Максимально допустимое расстояние между соседними точками на расстоянии шага

Интерфейс...

Тип данных: double

Синтаксис Automation:

length = Object.length	Получить свойство(*)
Object.length = length	Установить свойство (*)
length = Object.GetLength()	Получить свойство (**)
Object.SetLength(length)	Установить свойство (**)

lengthUnits – Единицы измерения длины

Интерфейс...

Тип данных: из перечисления ksLengthUnitsEnum

Синтаксис Automation:

lengthUnits = Object.lengthUnits	Получить свойство(*)
Object.lengthUnits = lengthUnits	Установить свойство (*)
lengthUnits = Object.GetLengthUnits()	Получить свойство (**)
Object.SetLengthUnits(lengthUnits)	Установить свойство (**)

maxTeselationCellCount – Максимальное количество ячеек в строке и ряду триангуляционной сетки (если 0, то не задано)

Интерфейс...

Тип данных: long

Синтаксис Automation:

maxTeselationCellCount Object.maxTeselationCellCount	=	Получить свойство(*)

Object.maxTeselationCellCount	=	Установить
maxTeselationCellCount		свойство (*)
maxTeselationCellCount	=	Получить
Object.GetMaxTeselationCellCount()		свойство (**)
Object.SetMaxTeselationCellCount(		Установить
maxTeselationCellCount)		свойство (**)

needCreateComponentsFiles – Создавать файлы компонентов

Интерфейс...

Тип данных: BOOL

Синтаксис:

needCreateComponentsFiles	=	Получить
Object.needCreateComponentsFiles		свойство(*)
Object.needCreateComponentsFiles	=	Установить
needCreateComponentsFiles		свойство (*)
needCreateComponentsFiles	=	Получить
Object.GetNeedCreateComponentsFiles		свойство (**)
()		
Object.SetNeedCreateComponentsFiles		Установить
(needCreateComponentsFiles)		свойство (**)

Примечание

По умолчанию используется значение TRUE - файлы компонентов создаются на диске рядом с файлом сборки,
FALSE - создаются локальные вставки.

organization – Организация

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

organization = Object.organization	Получить свойство(*)
Object.organization = organization	Установить свойство (*)
organization = Object.GetOrganization()	Получить свойство (**)
Object.SetOrganization(organization)	Установить свойство (**)

password – Пароль для загрузки упрощенных вставок перед экспортом

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

password = Object.password	Получить свойство(*)
Object.password = password	Установить свойство (*)
password = Object.GetPassword()	Получить свойство (**)
Object.SetPassword(password)	Установить свойство (**)

saveResultDocument – Сохранить полученный документ

Интерфейс...

Тип данных: BOOL

Синтаксис:

saveResultDocument Object.saveResultDocument	=	Получить свойство(*)
Object.saveResultDocument saveResultDocument	=	Установить свойство (*)
saveResultDocument Object.GetSaveResultDocument()	=	Получить свойство (**)
Object.SetSaveResultDocument(saveResultDocument)		Установить свойство (**)

stepType – Способ вычисления приращения параметра при движении по объекту

Интерфейс...

Тип данных: из перечисления ksStepTypeEnum

Синтаксис Automation:

stepType = Object.stepType	Получить свойство(*)

Object.stepType = stepType	Установить свойство (*)
stepType = Object.GetStepType()	Получить свойство (**)
Object.SetStepType(stepType)	Установить свойство (**)

Примечание:

Тип шага может определяться комбинацией значений из перечисления ksStepTypeEnum.

stitchSurfaces – Флаг необходимости сшивки поверхностей при импорте

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

stitchSurfaces = Object.stitchSurfaces	Получить свойство(*)
Object.stitchSurfaces = stitchSurfaces	Установить свойство (*)
stitchSurfaces = Object.GetStitchSurfaces()	Получить свойство (**)
Object.SetStitchSurfaces(stitchSurfaces)	Установить свойство (**)

stitchPrecision – Точность сшивки поверхностей

Интерфейс...

Тип данных: double

Синтаксис Automation:

stitchPrecision = Object.stitchPrecision	Получить свойство(*)
Object.stitchPrecision = stitchPrecision	Установить свойство (*)
stitchPrecision = Object.GetStitchPrecision()	Получить свойство (**)
Object.SetStitchPrecision(stitchPrecision)	Установить свойство (**)

textExportForm – Признак, чтения\записи текстов

Интерфейс...

Тип данных: из перечисления ksTextExportFormEnum

Синтаксис Automation:

textExportForm = Object.textExportForm	=	Получить свойство(*)
Object.textExportForm = textExportForm	=	Установить свойство (*)
textExportForm = Object.GetTextExportForm()	=	Получить свойство (**)
Object.SetTextExportForm(textExportForm)		Установить свойство (**)

topolgyIncluded – Признак, определяющий, включать ли топологию модели при экспорте

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- включить топологию,
FALSE	- не включать топологию.

Синтаксис Automation:

topolgyIncluded = iAdditionFormatParam.topolgyIncluded	Получить свойство(*)
iAdditionFormatParam.topolgyIncluded = topolgyIncluded	Установить свойство (*)
topolgyIncluded = iAdditionFormatParam.GetTopolgyIncluded()	Получить свойство (**)
iAdditionFormatParam.SetTopolgyIncluded(topolgyIncluded)	Установить свойство (**)

Примечание:

Свойство используется при записи в формат IGES.

IAdditionFormatParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Синтаксис COM:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры записи модели в дополнительные форматы.

GetObjectsOptions – Получить признак чтения\записи объектов модели

Интерфейс...

Синтаксис Automation:

BOOL GetObjectsOptions(long option);

Синтаксис COM:

BOOL GetObjectsOptions(long option);

Входные параметры:

option	- константа из перечисления ksD3ConverterOptionsEnum.

GetPlacement – Получить систему координат позиционирования модели

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Возвращаемое значение :

- указатель на интерфейс ksPlacement системы координат отображения модели в окне.

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение :

- указатель на интерфейс IPlacement системы координат отображения модели в окне.

SetPlacement – Установить систему координат позиционирования модели

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement(LPDISPATCH place);

Возвращаемое значение:

TRUE	- в случае успеха.

Входные параметры :

	- указатель на интерфейс ksPlacement системы координат отображения модели в окне.

Синтаксис COM:

BOOL SetPlacement(LPPLACEMENT place);

Возвращаемое значение :

TRUE	- в случае успеха.

Входные параметры :

	- указатель на интерфейс IPlacement системы координат отображения модели в окне.

SetObjectsOptions – Установить признак чтения\записи объектов модели

Интерфейс...

Синтаксис Automation:

BOOL SetObjectsOptions(long option, BOOL set);

Синтаксис COM:

BOOL SetObjectsOptions(long option, BOOL set);

Входные параметры:

option	- константа из перечисления ksD3ConverterOptionsEnum,
set	- TRUE включить признак, FALSE - выключить признак.

Интерфейсы спецификации ksSpecification, ISpecification

Интерфейс спецификации ksSpecification

События...

Справка системы КОМПАС...

kompas.chm::/1168_Glava135_Rabota_s_dokument.htm

Интерфейс для работы со спецификацией.

Примечания:

1. Для Автоматизации используется интерфейс ksSpecification.
2. Для COM используется интерфейс ISpecification3D и группа экспортных функций. Указатель на интерфейс ISpecification3D можно получить из интерфейса IDocument3D через стандартную функцию QueryInterface.
3. Указатель на интерфейс можно получить при помощи методов ksDocument2D::GetSpecification, ksSpcDocument::GetSpecification, ksDocument3D::GetSpecification.
4. События интерфейса позволяют контролировать редактирование описаний спецификации. В Automation источником событий для подписки на данный интерфейс является объект ksSpecification. В COM источником для подписки является документ (графический, документ-модель, спецификация).

Смотрите также:

- ksDocument2D,
- ksDocument3D,
- ksSpcDocument.

ksSpecification – методы

D3GetSpcObjForGeomWithLimit – Получить указатель на объект спецификации, подключенный к трехмерному компоненту (детали или подсборке) с ограничениями по номеру раздела и типу атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1154_132_2_Obwekty_specifikacii.htm

Аналог данного метода при использовании API экспортных функций - D3GetSpcObjForGeomWithLimit.

Синтаксис Automation:

long D3GetSpcObjForGeomWithLimit (LPCTSTR nameLib,

long numb,

LPDISPATCH part,

short first,

short section,

double attrTypeNumb);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
part	- указатель на интерфейс добавляемого компонента ksPart,
first	- 1 -первый объект, 0 - следующий объект,
section	- номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb	- номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

Синтаксис COM:

ISpecification3D::GetSpcObjForGeomWithLimit

D3GetSpcObjGeometry – Получить деталь или подсборку, подключенную к объекту спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksD3GetSpcObjGeometry.

Синтаксис Automation:

LPDISPATCH D3GetSpcObjGeometry (long spcObj);

Входные параметры:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

Указатель на интерфейс ksPart	- в случае успешного завершения,
0	- если деталь не подключена или в случае ошибки.

D3SpcIncludePart – Добавить или изменить трехмерный компонент (деталь или подсборку) в объекте спецификации

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_SPC_SELECTOBJECT.htm

Аналог данного метода при использовании API экспортных функций - D3SpcIncludePart.

Синтаксис Automation:

BOOL D3SpcIncludePart (LPDISPATCH part, VARIANT_BOOL fillTexts);

Входные параметры:

part	- указатель на интерфейс компонента ksPart,
fillTexts	- признак автоматического заполнения: наименование, обозначение, масса из свойств ksPart: TRUE - автозаполнение включено, FALSE - выключено.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Синтаксис COM:

ISpecification3D::SpcIncludePart

Примечание:

Если при подключении компонента присылается флаг fillTexts = true, в объекте спецификации взводится флаг ISpecificationBaseObject::SyncronizeWithProperties.

GetSpcObjectColumnTextEx – Получить текст объекта спецификации для определенного типа колонки и исполнения в виде динамического массива TEXT_LINE_ARR

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectColumnTextEx.

Синтаксис Automation:

LPDISPATCH ksGetSpcObjectColumnTextEx (long spcObj, long columnType, long ispoln, long block);

Входные параметры:

spcObj	- указатель на объект спецификации,
columnType	- тип колонки, SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер колонки данного типа начиная с 1,
block	- номер блока, если block = 0, то количество исполнений меньше количества исполнений в бланке спецификации (для групповой спецификации 2.113-75 количество исполнений меньше 10).

Возвращаемое значение:

указатель на интерфейс ksDynamicArray	- в случае успеха,
0	- в случае неудачи.

GetSpcObjectNotify – Получить источник событий для объекта спецификации

Интерфейс...

Синтаксис Automation:

SpcObjectNotify* GetSpcObjectNotify();

Возвращаемое значение:

- указатель на интерфейс источника событий SpcObjectNotify.

ksAddSpcDescription – Добавить описание спецификации в указанный графический документ

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данного метода при использовании API экспортных функций - ksAddSpcDescription.

Синтаксис Automation:

long ksAddSpcDescription (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс параметров описания спецификации ksSpcDescrParam,

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksDeleteSpcDescription – Удалить параметры описания спецификации в указанном графическом документе

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данного метода при использовании API экспортных функций - ksDeleteSpcDescription.

Синтаксис Automation:

long ksDeleteSpcDescription (long index);

Входной параметр:

index - индекс (номер) описания спецификации в документе.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksEditWindowSpcObject – Запустить окно редактирования для объекта спецификации, существующего в графическом документе

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEditWindowSpcObject.

Синтаксис Automation:

long ksEditWindowSpcObject (long obj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

1 - в случае удачного завершения.

Примечание:

Метод запускает процесс. Так как система может работать только с одним процессом, то нужно завершить другие "процессные" функции: ksDocument2D::ksCursor, ksDocument2D::ksPlacement, ksDocument2D::ksCommandWindow, ksDocument2D::ksEditViewObject, ksDocument2D::ksCreateViewObject.

ksGetCurrentSpcObject – Получить указатель текущего (выделенного или редактируемого) объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurrentSpcObject.

Синтаксис Automation:

long ksGetCurrentSpcObject();

Возвращаемое значение:

указатель на текущий объект	- в случае успеха,
0	- в случае неудачи.

Примечание:

Метод работает для видимых таблиц спецификации.

ksGetSpcPropertyFill – Получить флаг Синхронизировать со свойствами компонента

Интерфейс...

Синтаксис Automation:

BOOL ksGetSpcPropertyFill(long spcObj);

Входные параметры:

spcObj	- указатель на объект спецификации.

Возвращаемое значение:

Значение флага	- синхронизировать со свойствами компонента.

ksSetSpcPropertyFill – Установить флаг Синхронизировать со свойствами компонента

Интерфейс...

Синтаксис Automation:

BOOL ksGetSpcPropertyFill(long spcObj, long val);

Входные параметры:

spcObj	- указатель на объект спецификации,
val	- 1 включить синхронизацию, 0 - выключить синхронизацию.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

ksGetSpcColumnNumb – Получить номер колонки для данного объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcColumnNumb.

Синтаксис Automation:

long ksGetSpcColumnNumb (long spcObj,

long columnType,

long ispoIn,

long block);

Входные параметры:

spcObj	- указатель на объект спецификации,
columnType	- тип колонки,
ispoIn	- номер колонки данного типа, начиная с 1,
block	- номер блока исполнений.

Возвращаемое значение:

- номер колонки спецификации.

Типы колонок спецификации...

ksGetSpcColumnType – Получить параметры колонки по ее номеру для данного объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcColumnType.

Синтаксис Automation:

long ksGetSpcColumnType (long spcObj,

long colNumb,

LPDISPATCH par);

Входные параметры:

spcObj	- указатель на объект спецификации,
colNumb	- номер колонки, начиная с 1.

Выходной параметр:

par	- указатель на интерфейс параметров колонки ksSpcColumnParam.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Типы колонок спецификации...

Примечание:

Если количество исполнений больше числа колонок *Количество* в бланке спецификации, то недостающие колонки исполнений "пристыкованы" справа от видимых в таблице колонок, затем "пристыкованы" дополнительные колонки

ksGetSpcDescription – Получить указатель на интерфейс параметров описания спецификации для указанного документа

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SPC_DESCRIBE.htm`

Аналог данного метода при использовании API экспортных функций - ksGetSpcDescription.

Синтаксис Automation:

long ksGetSpcDescription (long index,

LPDISPATCH param,

BOOL* state);

Входной параметр:

index	- индекс (номер) описания спецификации в документе (-1 - текущее описание).

Выходные параметры:

param	- указатель на интерфейс ksSpcDescrParam параметров описания спецификации,
state	- состояние описания спецификации: TRUE - текущее, FALSE - не текущее.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Для документа-спецификации параметр index не используется.

ksGetSpcObjForGeom – Получить указатель на объект спецификации, подключенный к данному графическому объекту

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjForGeom.

Синтаксис Automation:

long ksGetSpcObjForGeom (BSTR nameLib,

long numb,

long obj,

short equal,

short first);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
obj	- указатель на графический объект или группу объектов (0 - макроэлемент редактирования),
equal	- признак вхождения геометрии в состав объекта: 1 - геометрия идентична геометрии объекта спецификации, 0 - геометрия входит в объект спецификации,
first	- 1 -первый объект, 0 - следующий объект.

Возвращаемое значение:

- указатель на объект спецификации.

Примечания:

1. Данный метод - только для графических документов.
2. В настоящий момент метод реализован только для equal = 1.

ksGetSpcObjForGeomWithLimit – Получить указатель на объект спецификации, подключенный к данному графическому объекту с ограничениями по номеру раздела и типу атрибута

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjForGeomWithLimit.

Синтаксис Automation:

long ksGetSpcObjForGeomWithLimit (BSTR nameLib,

long numb,

long obj,

short equal,

short first,

long section,

double attrTypeNumb);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
obj	- указатель на графический объект или группу объектов (0 - макроэлемент редактирования),
equal	- признак вхождения геометрии в состав объекта: 1 - геометрия идентична геометрии объекта спецификации, 0 - геометрия входит в объект спецификации,
first	- 1 - первый объект, 0 - следующий объект,
section	- номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb	- номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

Примечания:

1. Данный метод - только для графических документов.
2. В настоящий момент метод реализован только для equal = 1.

ksGetSpcObjGeometry – Получить деталь или подсборку, подключенную к объекту спецификации

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1183_135_5_3_Prosmotr_geometrii.htm

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjGeometry.

Синтаксис Automation:

long ksGetSpcObjGeometry (long spcObj);

Входные параметры:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

указатель на группу	- в случае успешного завершения,
0	- в случае, если геометрия не подключена или в случае ошибки.

ksGetSpcObjGeometryEx – Получить деталь или подсборку, подключенную к объекту спецификации

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1183_135_5_3_Prosmotr_geometrii.htm

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjGeometryEx.

Синтаксис Automation:

long ksGetSpcObjGeometryEx (long spcObj, long geomMode);

Входные параметры:

spcObj	- указатель на объект спецификации,
geomMode	0 - только геометрия;
	1 - только линии выноски;
	2 - геометрия и линии выноски.

Возвращаемое значение:

указатель на группу	- в случае успешного завершения,
0	- в случае, если геометрия не подключена или в случае ошибки.

ksGetSpcObject – Получить указатель на объект спецификации по номеру

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObject.

Синтаксис Automation:

long ksGetSpcObject (double objNumb);

Входной параметр:

objNumb	- уникальный номер объекта спецификации.

Возвращаемое значение:

- указатель на объект спецификации.

ksGetSpcObjectAttributeNumber – Получить номер атрибута объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectAttributeNumber.

Синтаксис Automation:

double ksGetSpcObjectAttributeNumber (long spcObj);

Входные параметры:

spcObj - объект спецификации.

Возвращаемое значение:

- номер атрибута.

Примечания:

Если spcObj = 0, то возвращается номер редактируемого в текущий момент объекта спецификации, т.е. внутри блока:

ksSpcObjectEdit(...);

ksGetSpcObjectAttributeNumber(...);

ksSpcObjectEnd();

ksGetSpcObjectColumnText – Получить строку с текстом из определенной колонки объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectColumnText.

Синтаксис Automation:

BSTR ksGetSpcObjectColumnText (long spcObj,

long columnType,

long ispoln,

long block);

Входные параметры:

spcObj - указатель на объект спецификации,

columnType - тип колонки,
ispoln - номер колонки данного типа, начиная с 1,
block - номер блока исполнений, начиная с 0 (если количество исполнений больше числа колонок типа *Количество*).

Типы колонок спецификации...

Возвращаемое значение:

- строка с текстом из колонки.

ksGetSpcObjectColumnTextAlign – Получить выравнивание для текста колонки объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectColumnTextAlign.

Синтаксис Automation:

long ksGetSpcObjectColumnTextAlign(long spcObj,
long columnNumber,
long lineIndex);

Входной параметр:

spcObj - указатель на объект спецификации,
columnNumber - номер колонки, начиная с 1,
lineIndex - индекс строки текста.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

Значение переменной align может принимать значения:

- 0 - Выравнивание влево;
- 1 - Выравнивание по центру;
- 2 - Выравнивание вправо;
- 3 - Выравнивание по ширине.

ksGetSpcObjectNumber – Получить уникальный номер объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectNumber.

Синтаксис Automation:

double ksGetSpcObjectNumber (long spcObj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

- уникальный номер объекта спецификации.

ksGetSpcObjectSummaryCount – Получить суммарное количество для одинаковых объектов

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1218_139_2_Podschet_summy_znach.htm

Аналог данного метода при использовании API экспортных функций - ksGetSpcObjectSummaryCount.

Синтаксис Automation:

double ksGetSpcObjectSummaryCount(long spcObj,
long ispoln,
long blockNumber);

Входные параметры:

spcObj - объект спецификации,
ispoln - номер колонки типа "количество", начиная с 1 (актуально для СП типа Б),
blockNumber - номер блока, начиная с 0 (актуально для СП типа А).

Возвращаемое значение:

- суммарное количество.

ksGetSpcPerformanceName – Получить отображаемое имя исполнения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcPerformanceName.

Синтаксис Automation:

BSTR ksGetSpcPerformanceName(long index,
long ispoln,
long block);

Входные параметры:

index	- индекс описания СП в документе,
ispoln	- номер колонки типа "количество", начиная с 1 (актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А).

Возвращаемое значение:

имя исполнения	- в случае успешного завершения,
FALSE	- в случае, если геометрия не подключена или в случае ошибки.

ksGetSpcSectionName – Получить название раздела спецификации по указателю на объект спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcSectionName.

Синтаксис Automation:

BSTR ksGetSpcSectionName (long spcObj);

Входной параметр:

spcObj	- указатель на объект спецификации.

Возвращаемое значение:

- строка с названием раздела спецификации.

ksGetSpcStyleParam – Получить указатель на интерфейс параметров ksSpcStyleParam или ksSpcTuningStyleParam для указанного стиля спецификации

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_LIB_STYLE_DIALOG.htm

Аналог данного метода при использовании API экспортных функций - ksGetSpcStyleParam.

Синтаксис Automation:

long ksGetSpcStyleParam (BSTR nameLib,

long numb,

LPDISPATCH par,

long tPar);

Входные параметры:

nameLib	- имя библиотеки стилей спецификаций (NULL - параметры берутся у текущего документа),
numb	- номер стиля спецификации в библиотеке,
tPar	- тип возвращаемых параметров.

Выходной параметр:

par	- указатель на интерфейс параметров ksSpcStyleParam или ksSpcTuningStyleParam.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

tPar = ALLPARAM - интерфейс ksSpcStyleParam,

tPar = SPC_TUNING_PARAM - интерфейс ksSpcTuningStyleParam.

ksGetSpcTableColumn – Получить количество колонок для стиля спецификации в текущем документе

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcTableColumn.

Синтаксис Automation:

long ksGetSpcTableColumn (BSTR nameLib,

long numb,

short additioanalCol);

Входные параметры:

nameLib	- имя библиотеки стилей спецификаций,
numb	- номер стиля в библиотеке,

additioanalCol	- признак колонок: 0 - видимые колонки таблицы спецификации, 1 - дополнительные колонки.

Возвращаемое значение:

	- количество колонок для стиля спецификации в текущем документе.

Примечания:

1. Если документ - спецификация, то параметры nameLib и numb не используются.
2. Если количество исполнений больше числа колонок "количество" в бланке спецификации, то количество колонок выдается с учетом всех исполнений.

ksGetTuningSpcStyleParam – Получить параметры настройки спецификации документа

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm`

Аналог данного метода при использовании API экспортных функций - ksGetTuningSpcStyleParam.

Синтаксис Automation:

long ksGetTuningSpcStyleParam (long index, LPDISPATCH par);

Входные параметры:

index	- индекс описания спецификации в документе,
par	- указатель на интерфейс ksSpcTuningStyleParam структуры параметров настроек спецификации.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если index = -1, то для текущего описания спецификации.
2. Для документа спецификации настройки можно изменить всегда.
3. Для графического документа в случае, если документ подключен к спецификации или включен режим "Спецификация на листе".
4. Для документа трехмерного объекта в случае, если документ подключен к спецификации.

ksGetWidthColumnSpc – Получить ширину колонки или текста в колонке

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetWidthColumnSpc.

Синтаксис Automation:

double ksGetWidthColumnSpc (long numColumn, BOOL cellOrText);

Входные параметры:

numColumn	- номер колонки,
cellOrText	- признак объекта, ширину которого требуется узнать: TRUE - колонка, FALSE - текст в колонке.

Возвращаемое значение:

- ширина колонки или текста в колонке.

ksSetCurrentSpcObject – Установить текущий объект спецификации либо по указателю на него, либо по его индексу

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetCurrentSpcObject.

Синтаксис Automation:

long ksSetCurrentSpcObject (long spcObj,

long index);

Входные параметры:

spcObj	- указатель на объект,
index	- индекс объекта.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод работает для видимых таблиц спецификации.

ksSetSpcDescription – Установить указатель на интерфейс параметров описания спецификации для указанного документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данного метода при использовании API экспортных функций - ksSetSpcDescription.

Синтаксис Automation:

long ksSetSpcDescription (long index,

LPDISPATCH param,

short state);

Входные параметры:

index	- индекс (номер) описания спецификации в документе (-1 - текущее описание),
param	- указатель на интерфейс параметров описания спецификации ksSpcDescrParam,
state	- состояние описания спецификации: TRUE - текущее, FALSE - не текущее.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечания:

1. Метод изменяет параметры описания спецификации и/или делает его активным, если state = TRUE.
2. Если param = NULL - меняется только состояние.
3. Для документа-спецификации параметр index не используется.

ksSetSpcObjectAttributeNumber – Установить номер атрибута объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcObjectAttributeNumber.

Синтаксис Automation:

BOOL ksSetSpcObjectAttributeNumber (long spcObj, double attrNumber);

Входные параметры:

spcObj	- объект спецификации,
attrNumber	- номер атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.- в случае неудачи.

Примечания:

Если spcObj = 0, то возвращается номер редактируемого в текущий момент объекта спецификации, т.е. внутри блока:

ksSpcObjectEdit(...); .

ksGetSpcObjectAttributeNumber(...);

ksSpcObjectEnd();

ksSetSpcObjectColumnText – Установить текст в колонке определенного типа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcObjectColumnText.

Синтаксис Automation:

long ksSetSpcObjectColumnText (long columnType,

long ispoln,

long block,

BSTR str);

Входные параметры:

colum nType	- тип колонки,
ispoln	- номер колонки данного типа,
block	- номер блока исполнений, начиная с 0 (если количество исполнений больше числа колонок типа *Количество*),
str	- строка, из которой нужно взять текст.

Возвращаемое значение:

1	- в случае удачного завершения.

ksSetSpcObjectColumnTextEx – Задать текст объекта спецификации для определенного типа колонки и исполнения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcObjectColumnTextEx.

Синтаксис Automation:

long ksSetSpcObjectColumnTextEx (long columnType, long ispoln, long block, LPDISPATCH arr);

Входные параметры:

columnType	- тип колонки, SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер колонки данного типа, начиная с 1,
block	- номер блока. если block = 0, то количество исполнений меньше количества исполнений в бланке спецификации (для групповой спецификации 2.113-75 количество исполнений меньше 10),
arr	- указатель на интерфейс ksDynamicArray массива строк, тип массива - TEXT_LINE_ARR.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Функция работает в режиме создания нового или редактирования существующего объекта спецификации ksSpcObjectCreate(...); или ksSpcObjectEdit(...);. ksSetSpcObjectColumnTextEx(...);...reference spsObj = ksSpcObjectEnd();

ksSetSpcObjectColumnTextAlign – Установить выравнивание для текста колонки объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcObjectColumnTextAlign.

Синтаксис Automation:

BOOL ksSetSpcObjectColumnTextAlign (long spcObj,
long columnNumber,
long lineIndex,
long align);

Входной параметр:

spcObj	- указатель на объект спецификации.

columnNumber	- номер колонки, начиная с 1,
lineIndex	- индекс строки текста,
align	- выравнивание.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

Значение переменной align может принимать значения:

- 0 - Выравнивание влево;
- 1 - Выравнивание по центру;
- 2 - Выравнивание вправо;
- 3 - Выравнивание по ширине.

ksSetSpcPerformanceName – Установить отображаемое имя исполнения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcPerformanceName.

Синтаксис Automation:

BOOL ksSetSpcPerformanceName(long index,
long ispoln,
long block,
BSTR name);

Входные параметры:

Index	- индекс описания СП в документе,
Ispoln	- номер колонки типа *Количество*, начиная с 1(актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А),
Name	- имя исполнения.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае, если геометрия не подключена или в случае ошибки.

ksSetTuningSpcStyleParam – Установить параметры настроек спецификации документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Аналог данного метода при использовании API экспортных функций - ksSetTuningSpcStyleParam.

Синтаксис Automation:

long ksSetTuningSpcStyleParam(long index, LPDISPATCH par);

Входные параметры:

index	- индекс описания спецификации в документе,
par	- указатель на интерфейс ksSpcTuningStyleParam структуры параметров настроек спецификации.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если index = -1, то для текущего описания спецификации.
2. Для документа спецификации настройки можно изменить всегда.
3. Для графического документа в случае, если документ подключен к спецификации или включен режим "Спецификация на листе".
4. Для документа трехмерного объекта в случае, если документ подключен к спецификации.

ksSpcChangeValue – Изменить значение компоненты с указанным номером в указанной колонке

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcChangeValue.

Синтаксис Automation:

long ksSpcChangeValue (long colNumb,

long itemNumb,

LPDISPATCH userPars,

short typeVal);

Входные параметры:

colNumb	- номер колонки, начиная с единицы,
itemNumb	- номер компоненты, начиная с единицы,
userPars	- указатель на интерфейс ksUserParam,
typeVal	- тип данных.

Типы данных..

Возвращаемое значение:

1	- в случае удачного завершения.

Примечания:

1. Метод работает в режиме создания нового или редактирования существующего объекта спецификации.
2. Если typeVal - запись (RECORD_ATTR_TYPE), в колонке может быть несколько компонент и itemNumb может быть не равен 1.

ksSpcCount – Установить количество деталей для определенного исполнения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcCount.

Синтаксис Automation:

long ksSpcCount (short ispoln,

BSTR sCount);

Входные параметры:

ispoln	- номер исполнения, начиная с 1,
sCount	- количество деталей.

Возвращаемое значение:

1	- в случае удачного завершения (если в объекте существует колонка с типом *Количество*).

ksSpcDocLinksClear – Создать объект спецификации в графическом документе

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcObjectCreate.

Синтаксис Automation:

BOOL ksSpcDocLinksClear(long doc)

Входные параметры:

doc	- указатель на документ-источник объекта спецификации (графический, 3D, спецификация).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

Функция удаляет связки документа-источника объекта спецификации (графический, 3D, спецификация) с документом-владельцем объекта спецификации;

Если doc = 0, то документом-источником объекта спецификации считается текущий документ.

ksSpcIncludeReference – Добавить или изменить геометрию графического объекта или линию-выноску в объекте спецификации

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/1182_135_5_1_Vkljuchenie_geomet.htm

Аналог данного метода при использовании API экспортных функций - ksSpcIncludeReference.

Синтаксис Automation:

long ksSpcIncludeReference (long obj,

short clear);

Входные параметры:

obj	- группа объектов или объект вида,
clear	- признак обработки существующей геометрии объекта спецификации: 1 - удалить геометрию, 0 - добавить новые объекты к существующей геометрии.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksSpcMassa – Заполнить дополнительную колонку масса для объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcMassa.

Синтаксис Automation:

long ksSpcMassa (BSTR sMassa);

Входной параметр:

sMassa	- строка с массой детали.

Возвращаемое значение:

1	- в случае удачного завершения (если в объекте существует колонка с типом *Масса*),
0	- в случае неудачи.

ksSpcObjectCreate – Создать объект спецификации в графическом документе

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/1195_136_1_Sozdanie_obwektov_sp.htm`

Аналог данного метода при использовании API экспортных функций - ksSpcObjectCreate.

Синтаксис Automation:

long ksSpcObjectCreate (BSTR nameLib,

long styleNumb,

long secNumb,

long subSecNumb,

double numb,

short typeObj);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
styleNumb	- номер стиля спецификации в библиотеке,
secNumb	- номер раздела,
subSecNumb	- номер подраздела,
numb	- для базового объекта - тип атрибута, который задан по ключам, или 0, для вспомогательного объекта - номер базового объекта, к которому прикрепляется вспомогательный, либо 0,

typeObj - тип строки спецификации
(0- базовый объект,
1 – вспомогательный объект).

Возвращаемое значение:

1 - в случае удачного завершения.

Примечания:

1. Объект спецификации - составной объект. Его создание завершается методом ksSpecification::ksSpcObjectEnd.
2. Для атрибутов, описанных по ключам, нужно указать тип в numb, потому что типов с одинаковыми ключами может быть несколько.
3. Параметры nameLib и styleNumb используются только для объектов спецификации в графическом документе.

ksSpcObjectEdit – Запустить режим редактирования объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcObjectEdit.

Синтаксис Automation:

long ksSpcObjectEdit (long spcObj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

1. После вызова данного метода у объекта можно изменить атрибуты, геометрию, линии выноски, позицию, массу.
2. Редактирование объекта завершится вызовом метода ksSpecification::ksSpcObjectEnd.

ksSpcObjectEnd – Завершить создание или редактирование объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcObjectEnd.

Синтаксис Automation:

long ksSpcObjectEnd();

Возвращаемое значение:

- указатель на объект спецификации.

Примечание:

Предварительно объект спецификации должен быть открыт на редактирование, или должен быть запущен режим его создания методами ksSpecification::ksSpcObjectEdit или ksSpecification::ksSpcObjectCreate.

ksSpcPosition – Установить номер позиции

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcPosition.

Синтаксис Automation:

long ksSpcPosition (long pos);

Входной параметр:

pos - номер позиции.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка с типом *Позиция*),
0 - в случае неудачи.

ksSpcVisible – Установить признак видимости компонента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSpcVisible.

Синтаксис Automation:

long ksSpcVisible (long colNumb,

long itemNumb,

short flagOn);

Входные параметры:

colNumb - номер колонки, начиная с единицы,
itemNumb - номер компоненты, начиная с единицы,
flagOn - признак видимости компонента:
1 -включить,
0- выключить.

Возвращаемое значение:

1 - в случае удачного завершения,

0 - в случае неудачи.

Примечания:

1. Метод работает в режиме создания нового или редактирования существующего объекта спецификации.
2. В случае записи в колонке может быть несколько компонент и itemNumb может быть не равен 1.

Интерфейс спецификации ISpecification3D

Справка системы КОМПАС...

KOMPAS.chm::/1154_132_2_Obwekty_specifikacii.htm

Интерфейс для работы со спецификацией.

COM – интерфейс.

Примечания:

1. Для Автоматизации используется интерфейс ksSpecification.
2. Для COM используется интерфейс ISpecification3D и группа экспортных функций.
3. Указатель на интерфейс можно получить при помощи метода IDocument3D::GetSpecification.

   Смотрите также: ksDocument3D

ISpecification3D – методы

D3GetSpcObjGeometry – Получить деталь или подсборку, подключенную к объекту спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksD3GetSpcObjGeometry.

Синтаксис Automation:

LPPART D3GetSpcObjGeometry(long spcObj);

Входные параметры:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

Указатель на интерфейс IPart	- в случае успешного завершения,
0	- если деталь не подключена или в случае ошибки.

GetSpcObjForGeomWithLimit – Получить указатель на объект спецификации, подключенный к трехмерному компоненту (детали или подсборке) с ограничениями по номеру раздела и типу атрибута

Интерфейс...

Синтаксис COM:

long GetSpcObjForGeomWithLimit (LPOLESTR nameLib,

long numb,

LPPART part,

short first,

short section,

double attrTypeNumb);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
part	- указатель на интерфейс добавляемого компонента IPart,
first	- 1 -первый объект, 0 - следующий объект,
section	- номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb	- номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

Синтаксис Automation:

D3GetSpcObjForGeomWithLimit

SpcIncludePart – Подключить трехмерный компонент (деталь или подсборку) к объекту спецификации

Интерфейс...

Синтаксис COM:

BOOL SpcIncludePart (LPPART part,

BOOL fillTexts);

Входные параметры:

part	- указатель на интерфейс компонента IPart,

fillTexts - признак автоматического заполнения: наименование, обозначение, масса из свойств ksPart:
TRUE - автозаполнение включено,
FALSE - выключено.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Синтаксис Automation:

D3SpcIncludePart

Примечание:

Если при подключении компонента присылается флаг fillTexts = true, в объекте спецификации взводится флаг ISpecificationBaseObject::SyncronizeWithProperties.

Интерфейс управления положением компонентов в сборке (Интерфейсы ksComponentPositioner, IComponentPositioner)

Справка системы КОМПАС...

```
kompas.chm::/
peremescheniya_komponentov_obzor.htm#displace_general
```

Интерфейс управления положением компонентов в сборке.

ksComponentPositioner - интерфейс Automation
IComponentPositioner - интерфейс COM

Описание:

Сдвиг проводится в плоскости, которая определена относительно сборки. Для сдвига точка захвата должна быть определена в системе координат сдвигаемого компонента, а точка сдвига должна быть в системе координат сборки и преобразована в проекцию на плоскость сдвига. Перед началом сдвига нужно задать плоскость и точку захвата.

Вращение проводится относительно оси, которая задается относительно сборки. Перед началом вращения нужно задать ось вращения. Это может быть ось или прямолинейное ребро. Начало перемещения определяется вызовом метода Prepare. Окончание перемещения определяется вызовом метода Finish. Между ними может быть сколько угодно вызовов MoveComponent или RotateComponent в зависимости от указанного типа перемещения.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса ksDocument3D::ComponentPositioner.

IComponentPositioner – методы

Finish – Завершить перемещение компонента

Интерфейс...

Синтаксис Automation:

BOOL Finish();

Синтаксис COM:

BOOL Finish();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Начало перемещения определяется вызовом метода Prepare. Между ними может быть сколько угодно вызовов MoveComponent или RotateComponent в зависимости от указанного типа перемещения.

MoveComponent – Переместить компонент

Интерфейс...

Справка системы КОМПАС...

```
kompas.chm::/
peremescheniya_komponentov_obzor.htm#displace_general
```

Синтаксис Automation:

BOOL MoveComponent (double x, double y, double z);

Синтаксис COM:

BOOL MoveComponent (double x, double y, double z);

Входные параметры:

x, y, z	- координаты точки сдвига.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Перед началом сдвига нужно задать плоскость и точку захвата. Сдвиг проводится в плоскости, которая определена относительно сборки. Начало перемещения определяется вызовом метода Prepare с типом pnMove. Окончание перемещения определяется вызовом метода Finish. Между ними может быть сколько угодно вызовов MoveComponent.

Prepare – Подготовиться к перемещению компонента

Интерфейс...

Синтаксис Automation:

long Prepare(ksPart * part, Positioner_Type positionerType);

Синтаксис COM:

long Prepare(LPPART part, Positioner_Type positionerType);

Входные параметры:

part	- указатель на интерфейс компоненты ksPart или IPart,
positionerType	- тип перемещения.

Возвращаемое значение:

0	- в случае успешного завершения,
1	- в случае неудачи.

Примечание:

Окончание перемещения определяется вызовом метода Finish. Между ними может быть сколько угодно вызовов MoveComponent или RotateComponent в зависимости от указанного типа перемещения.

RotateComponent – Повернуть компонент

Интерфейс...

Справка системы КОМПАС...

```
kompas.chm::/
peremescheniya_komponentov_obzor.htm#displace_general
```

Синтаксис Automation:

BOOL RotateComponent (double angl);

Синтаксис COM:

BOOL RotateComponent (double angl);

Входные параметры:

angl	- угол поворота компоненты, рад.

Возвращаемое значение:

TRUE	- в случае успешного завершения,

FAL - в случае неудачи.
SE

Примечание:

Перед началом вращения нужно задать ось вращения. Это может быть ось или прямолинейное ребро. Начало перемещения определяется вызовом метода Prepare с типом pnRotate. Окончание перемещения определяется вызовом метода Finish. Между ними может быть сколько угодно вызовов RotateComponent.

SetAxis – Задать ось

Интерфейс...

Синтаксис Automation:

BOOL SetAxis (LPDISPATCH axis);

Синтаксис COM:

BOOL SetAxis (LPUNKNOWN axis);

Входные параметры:

axis - указатель на интерфейс конструктивной оси или прямолинейного ребра.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetAxisByPoints – Задать ось по точкам

Интерфейс...

Синтаксис Automation:

BOOL SetAxisByPoints (double x1,
double y1,
double z1,
double x2,
double y2,
double z2);

Синтаксис COM:

BOOL SetAxisByPoints (double x1,
double y1,
double z1,
double x2,
double y2,
double z2);

Входные параметры:

x1, y1, z1	- координаты первой точки оси,
x2, y2, z2	- координаты второй точки оси.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Координаты точек, задающих ось, должны быть определены относительно сборки.

SetDragPoint – Задать точку захвата (ручка)

Интерфейс...

Синтаксис Automation:

BOOL SetDragPoint (double x, double y, double z);

Синтаксис COM:

BOOL SetDragPoint (double x, double y, double z);

Входные параметры:

x, y, z	- координаты точки захвата.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Точка захвата должна быть определена в системе координат сдвигаемого компонента.

SetPlane – Задать плоскость

Интерфейс...

Синтаксис Automation:

BOOL SetPlane (LPDISPATCH plane);

Синтаксис COM:

BOOL SetPlane (LPUNKNOWN plane);

Входные параметры:

plane	- указатель на интерфейс конструктивной плоскости или плоской поверхности.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

SetPlaneByPlacement – Задать плоскость по системе координат

Интерфейс...

Синтаксис Automation:

BOOL SetPlaneByPlacement (ksPlacement * plane);

Синтаксис COM:

BOOL SetPlaneByPlacement (LPPLACEMENT plane);

Входные параметры:

plane	- указатель на интерфейс ksPlacement или IPlacement системы координат, задающей плоскость.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Система координат, задающая плоскость, должна быть определена относительно сборки.

SetPlaneByPoints – Задать плоскость по точкам

Интерфейс...

Синтаксис Automation:

BOOL SetPlaneByPoints (double x1,
double y1,
double z1,
double x2,
double y2,
double z2,
double x3,
double y3,
double z3);

Синтаксис COM:

BOOL SetPlaneByPoints (double x1,
double y1,
double z1,
double x2,

double y2,
double z2,
double x3,
double y3,
double z3);

Входные параметры:

x1, y1, z1	- координаты первой точки на плоскости,
x2, y2, z2	- координаты второй точки на плоскости,
x3, y3, z3	- координаты третьей точки на плоскости.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Координаты точек, задающих плоскость, должны быть определены относительно сборки.

Источник событий для документа модели

Document3DNotify

Параметры отображения точки, позволяющей определить место применения контрола (Интерфейс IMouseEnterLeaveParameters)

Примечание:

Интерфейс параметров используется в событии IProcess2DNotify::GetMouseEnterLeavePoint

Версия: KOMPAS v19

IMouseEnterLeaveParameters – свойства

X – Координата X

Интерфейс...

Тип данных: double

Синтаксис Automation:

X = Object.X	Получить свойство (*)
Object.X = X	Установить свойство (*)

X = Object.GetX()	Получить свойство(**)
Object.SetX(X)	Установить свойство (**)

Y – Координата Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

Y = Object.Y	Получить свойство (*)
Object.Y = Y	Установить свойство (*)
Y = Object.GetY()	Получить свойство(**)
Object.SetY(Y)	Установить свойство (**)

Offset – Смещение символа относительно точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

Offset = Object.Offset	Получить свойство (*)
Object.Offset = Offset	Установить свойство (*)
Offset = Object.GetOffset()	Получить свойство(**)
Object.SetOffset(Offset)	Установить свойство (**)

OffsetAngle – Направление смещения символа относительно точки

Интерфейс...

Тип данных: double

Синтаксис Automation:

OffsetAngle = Object.OffsetAngle	=	Получить свойство (*)
Object.OffsetAngle = OffsetAngle	=	Установить свойство (*)
OffsetAngle = Object.GetOffsetAngle()	=	Получить свойство(**)
Object.SetOffsetAngle(OffsetAngle)		Установить свойство (**)

Symbol – Символ, отображаемый в поле документа, при наведении на контрол

Интерфейс...

Тип данных: long

Синтаксис Automation:

Symbol = Object.Symbol	Получить свойство (*)
Object.Symbol = Symbol	Установить свойство (*)
Symbol = Object.GetSymbol()	Получить свойство(**)
Object.SetSymbol(Symbol)	Установить свойство (**)

SymbolColor – Цвет символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: long

Синтаксис Automation:

SymbolColor = Object.SymbolColor	=	Получить свойство (*)
Object.SymbolColor = SymbolColor	=	Установить свойство (*)
SymbolColor = Object.GetSymbolColor()	=	Получить свойство(**)
Object.SetSymbolColor(SymbolColor)		Установить свойство (**)

SymbolFont – Шрифт символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

SymbolFont = Object.SymbolFont	=	Получить свойство (*)
Object.SymbolFont = SymbolFont	=	Установить свойство (*)
SymbolFont = Object.GetSymbolFont()	=	Получить свойство(**)
Object.SetSymbolFont(SymbolFont)		Установить свойство (**)

SymbolScale – Увеличение высоты символа, отображаемого в поле документа, при наведении на контрол

Интерфейс...

Тип данных: double

Синтаксис Automation:

SymbolScale = Object.SymbolScale	=	Получить свойство (*)
Object.SymbolScale = SymbolScale	=	Установить свойство (*)
SymbolScale = Object.GetSymbolScale()	=	Получить свойство(**)
Object.SetSymbolScale(SymbolScale)		Установить свойство (**)

Синтаксис COM:

Object.get_SymbolScale(&SymbolScale)	Получить свойство
Object.put_SymbolScale(SymbolScale)	Установить свойство

Интерфейс коллекции проекций отображения модели (Интерфейсы ksViewProjectionCollection, IViewProjectionCollection)

Справка системы КОМПАС...

kompas.chm::/674_82_4_Orientacija_modeli.htm

Интерфейс массива проекций отображения модели в окне.

ksViewProjectionCollection	- интерфейс Automation
IViewProjectionCollection	- интерфейс COM

Типы предопределенных проекций отображения...

Описание:

К предопределенным проекциям можно добавлять пользовательские, индексы которых могут изменяться.

Примечание:

Получить этот интерфейс можно методом ksDocument3D::GetViewProjectionCollection, при этом он будет автоматически заполнен проекциями, заданными в данном документе.

IViewProjectionCollection – свойства

viewProjectionScheme – Текущая схема ориентаций модели

Интерфейс...

Тип данных: из перечисления ksViewProjectionScheme

Синтаксис Automation:

viewProjectionScheme = Object.viewProjectionScheme	Получить свойство (*)
Object.viewProjectionScheme = viewProjectionScheme	Установить свойство (*)
viewProjectionScheme = Object.GetViewProjectionScheme()	Получить свойство(**)
Object.SetViewProjectionScheme(viewProjectionScheme)	Установить свойство (**)

IViewProjectionCollection – методы

Add – Добавить элемент в конец массива

Интерфейс...

Синтаксис Automation:

BOOL Add (LPDISPATCH projection);

Синтаксис COM:

BOOL Add (LPVIEWPROJECTION projection);

Входные параметры:

projection	- указатель на интерфейс ksViewProjection или IViewProjection.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Добавить в массив можно только проекцию с индексом vp_None. После добавления проекция становится текущей и ей присваивается новый индекс, равный ее индексу в массиве. После добавления проекции редактирование ее параметров невозможно.

AddUnfoldProjection – Добавить проекцию отображения – развертка

Интерфейс...

Синтаксис Automation:

LPDISPATCH AddUnfoldProjection(VARIANT place);

Возвращаемое значение:

- указатель на интерфейс проекции ksViewProjection.

Синтаксис COM:

LPVIEWPROJECTION AddUnfoldProjection(VARIANT * place);

Возвращаемое значение:

- указатель на интерфейс проекции IViewProjection.

DetachByBody – Отсоединить элемент по указателю на интерфейс

Интерфейс...

Синтаксис Automation:

BOOL DetachByBody (LPDISPATCH projection);

Синтаксис COM:

BOOL DetachByBody (LPVIEWPROJECTION projection);

Входные параметры:

projection	- указатель на интерфейс ksViewProjection или IViewProjection.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Отсоединить можно только пользовательскую проекцию, стандартные проекции не отсоединяются.

DetachByIndex – Отсоединить элемент по индексу

Интерфейс...

Синтаксис Automation:

BOOL DetachByIndex (long index);

Синтаксис COM:

BOOL DetachByIndex (long index);

Входной параметр:

index	- индекс проекции в массиве.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Отсоединить можно только пользовательскую проекцию, стандартные проекции не отсоединяются.

DetachByName – Отсоединить элемент по имени проекции

Интерфейс...

Синтаксис Automation:

BOOL DetachByName (BSTR name);

Синтаксис COM:

BOOL DetachByName (LPOLESTR name);

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Отсоединить можно только пользовательскую проекцию, стандартные проекции не отсоединяются.

FindIt – Получить индекс элемента в массиве

Интерфейс...

Синтаксис Automation:

long FindIt (LPDISPATCH projection);

Синтаксис COM:

long FindIt (LPVIEWPROJECTION projection);

Возвращаемое значение:

индекс элемента	- если элемент найден в массиве,
-1	- если элемент не найден.

Входные параметры:

projection	- указатель на интерфейс ksViewProjection или IViewProjection.

First – Получить указатель на первый элемент массива

Интерфейс...

Синтаксис Automation:

LPDISPATCH First();

Синтаксис COM:

LPVIEWPROJECTION First();

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

GetByIndex – Получить указатель на элемент массива по индексу

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByIndex (long index);

Синтаксис COM:

LPVIEWPROJECTION GetByIndex (long index);

Входной параметр:

index - индекс проекции в массиве.

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

GetByName – Получить указатель на элемент массива по имени

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetByName (BSTR name, BOOL testFullName, BOOL testIgnoreCase);

Синтаксис COM:

LPVIEWPROJECTION GetByName (LPOLESTR name, BOOL testFullName, BOOL testIgnoreCase);

Входные параметры:

name - имя проекции,
testFullName - признак полного имени:
TRUE - name - полное имя,
FALSE - имя name может быть частью полного имени,
testIgnoreCase - признак игнорирования регистра символов:
TRUE - игнорировать регистр,
FALSE - учитывать регистр.

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

GetBaseUserOrientation – Получить текущую пользовательскую базовую ориентацию модели

Интерфейс...

Синтаксис Automation:

VARIANT GetBaseUserOrientation();

Синтаксис COM:

VARIANT GetBaseUserOrientation();

Возвращаемое значение:

- массив SafeAttray типа (VT_ARRAY | VT_R8).

Примечание:

1. Элементы матрицы возвращаются в виде одномерного массива из 16 элементов. Матрица имеет размер 4x4.
2. Метод позволяет получить матрицу, которая может использоваться для отображения вида спереди.
3. По умолчанию используется единичная матрица без смещения.

GetCount – Получить количество проекций в массиве

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество проекций в массиве.

Last – Получить указатель на последний элемент массива

Интерфейс...

Синтаксис Automation:

LPDISPATCH Last();

Синтаксис COM:

LPVIEWPROJECTION Last();

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

NewViewProjection – Добавить новую проекцию

Интерфейс...

Синтаксис Automation:

LPDISPATCH NewViewProjection();

Синтаксис COM:

LPVIEWPROJECTION NewViewProjection();

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

Примечание:

Проекции присваивается индекс vp_None. Созданная проекция в массив не добавляется, для добавления в массив нужно использовать метод Add.

Next – Получить указатель на следующий элемент массива

Интерфейс...

Синтаксис Automation:

LPDISPATCH Next();

Синтаксис COM:

LPVIEWPROJECTION Next();

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

Prev – Получить указатель на предыдущий элемент массива

Интерфейс...

Синтаксис Automation:

LPDISPATCH Prev();

Синтаксис COM:

LPVIEWPROJECTION Prev();

Возвращаемое значение:

- указатель на интерфейс ksViewProjection или IViewProjection.

Refresh – Обновить массив

Интерфейс...

Синтаксис Automation:

BOOL Refresh();

Синтаксис COM:

BOOL Refresh();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод перезаполняет массив проекциями, заданными в документе.

SetBaseUserOrientation – Установить пользовательскую базовую ориентацию модели

Интерфейс...

Синтаксис Automation:

BOOL SetBaseUserOrientation(VARIANT place);

Синтаксис COM:

BOOL SetBaseUserOrientation(VARIANT * place);

Входные параметры:

place	- массив SafeAttray типа (VT_ARRAY | VT_R8).

Примечание:

1. Элементы матрицы возвращаются в виде одномерного массива из 16 элементов. Матрица имеет размер 4x4.
2. Метод позволяет получить матрицу, которая может использоваться для отображения вида спереди.
3. По умолчанию используется единичная матрица без смещения.

Проекции отображения модели в окне (Интерфейсы ksViewProjection, IViewProjection)

Справка системы КОМПАС...

`kompas.chm::/674_82_4_Orientacija_modeli.htm`

Интерфейс проекции отображения модели в окне.

ksViewProjection	- интерфейс Automation
IViewProjection	- интерфейс COM

Типы предопределенных проекций отображения...

Описание:

К этим проекциям можно добавлять пользовательские, индексы которых могут изменяться. Свойства уже заданных в документе проекций (с индексом больше vp_None) изменить нельзя. Эти свойства доступны только для чтения.

Примечание:

Данный интерфейс может быть получен с использованием методов интерфейса ksViewProjectionCollection.

IViewProjection – свойства

UserProjectionIndex– Индекс проекции отображения модели в окне

Интерфейс...

Тип данных: long

Синтаксис Automation:

index = iViewProjection.index	Получить свойство (*)
index = iViewProjection.GetIndex()	Получить свойство (**)

Примечание:

1. Свойство доступно только для чтения.
2. Свойство работает только для пользовательских проекций. Для получения типа проекции нужно использовать метод ksViewProjection::GetViewProjectonType.

name – Имя проекции

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

name = iViewProjection.name	Получить свойство(*)
iViewProjection.name = name	Установить свойство (*)
name = iViewProjection.GetName()	Получить свойство (**)
iViewProjection.SetName(name)	Установить свойство (**)

scale – Масштаб отображения модели в окне

Интерфейс...

Тип данных: double

Синтаксис Automation:

scale = iViewProjection.scale	Получить свойство(*)
iViewProjection.scale = scale	Установить свойство (*)
scale = iViewProjection.GetScale()	Получить свойство (**)
iViewProjection.SetScale(scale)	Установить свойство (**)

Примечание:

Масштаб можно установить для любой проекции. Если значение масштаба равно -1, то масштаб отображения будет задан таким образом, чтобы в окне был показан документ целиком - см.

Справка системы КОМПАС ...

`KOMPAS.chm::/670_82_3_1_Masshtabirovanie_i_s.htm`

Если проекция является текущей, SetScale устанавливает нужный масштаб сразу.

Получить масштаб отображения можно для пользовательской проекции, либо для текущей.

IViewProjection – методы

GetPlacement – Получить интерфейс системы координат отображения модели в окне

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlacement();

Синтаксис COM:

LPPLACEMENT GetPlacement();

Возвращаемое значение:

- указатель на интерфейс ksPlacement или IPlacement.

Примечание:

Изменение параметров системы координат возможно только для проекции с индексом vp_None.

GetViewProjectonType – Получить тип проекции

Интерфейс...

Синтаксис Automation:

long GetViewProjectonType();

Синтаксис COM:

long GetViewProjectonType();

Возвращаемое значение:

- Тип проекции из перечисления ksViewProjectionType.

IsCurrent – Определить, является ли данная проекция текущей

Интерфейс...

Синтаксис Automation:

BOOL IsCurrent();

Синтаксис COM:

BOOL IsCurrent();

Возвращаемое значение:

TRUE	- данная проекция является текущей,
FALSE	- данная проекция не является текущей.

Примечание:

Для проекции с индексом vp_None возвращаемое значение всегда FALSE.

SetCurrent – Установить данную проекцию отображения модели в окне текущей

Интерфейс...

Синтаксис Automation:

BOOL SetCurrent();

Синтаксис COM:

BOOL SetCurrent();

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Для проекции с индексом vp_NormalTo необходимо, чтобы в модели был выбран хотя бы один плоский объект.
2. Для проекции с индексом vp_None модель разворачивается и масштабируется в соответствии с заданными параметрами проекции. Сама проекция в массив не добавляется.

SetMatrix3D – Установить систему координат отображения модели в окне по матрице

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL SetMatrix3D (VARIANT Matrix3D);

Синтаксис COM:

BOOL SetMatrix3D (VARIANT Matrix3D);

Входные параметры:

Динамический массив SAFEARRAY double (VT_ARRAY | VT_R8).

В массиве должны лежать 16 элементов, которые представляют собой матрицу размера 4x4.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

1. Для отображения данной проекции в модели надо после установки системы координат вызвать метод SetCurrent.
2. Матрицу можно получить у ассоциативного вида методом ksDocument2D::ksAssociationViewMatrix3D.

SetPlacement – Установить систему координат отображения модели в окне

Интерфейс...

Синтаксис Automation:

BOOL SetPlacement (LPDISPATCH place);

Синтаксис COM:

BOOL SetPlacement (LPPLACEMENT place);

Входные параметры:

- указатель на интерфейс ksPlacement или IPlacement.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечание:

Изменение параметров системы координат возможно только для проекции с индексом vp_None.

Интерфейс коллекции атрибутов объектов модели (Интерфейс Attribute3DCollection)

Интерфейс коллекции трехмерных координат (Интерфейсы ksCoordinate3dCollection, ICoordinate3dCollection)

Интерфейс коллекции трехмерных координат.

ksCoordinate3dCollection	- интерфейс Automation
ICoordinate3dCollection	- интерфейс COM

Примечания.

1. Коллекция содержит список трехмерных координат точек. Используется при получении результатов расчета пересечений кривой с деталью, телом или поверхностью.
2. Получить интерфейс коллекции можно с помощью метода ksDocument3D::GetInterface с параметром o3dType = o3d_coordinate3dCollection.

ICoordinate3dCollection – методы

GetByIndex – Получить координаты точки по индексу

Интерфейс...

Синтаксис Automation:

BOOL GetByIndex (long index, double* x, double* y, double* z);

Синтаксис COM:

BOOL GetByIndex (long index, double* x, double* y, double* z);

Входные параметры:

index	- индекс точки.

Выходные параметры:

x, y, z	- координаты точки.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetCount – Получить количество точек

Интерфейс...

Синтаксис Automation:

long GetCount();

Синтаксис COM:

long GetCount();

Возвращаемое значение:

- количество точек в коллекции.

GetSafeArray – Сформировать массив SAFEARRAY координат точек

Интерфейс...

Синтаксис Automation:

BOOL GetSafeArray (VARIANT* pArray);

Синтаксис COM:

BOOL GetSafeArray (VARIANT* pArray);

Выходные параметры:

pArray	- одномерный массив вещественных чисел VT_ARRAR | VT_R8.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

Примечания:

Координаты точек в полученном массиве лежат в следующей последовательности: x0, y0, z0, x1, y1, z1, ...xi, yi, zi

Интерфейс сопряжения (Интерфейсы ksMateConstraint, IMateConstraint)

МЦХ тела вращения или выдавливания (Интерфейсы ksMassInertiaParam, IMassInertiaParam)

Интерфейс области применения для тел в операции (Интерфейсы ksChooseBodies, IChooseBodies)

Интерфейс области применения для тел в операции.

ksChooseBodies	- интерфейс Automation
IChooseBodies	- интерфейс COM

Примечание.

Интерфейс позволяет выбирать тела, участвующие в операции.

IChooseBodies – свойства

ChooseBodiesType – Тип действия над телами

Интерфейс...

Тип данных: из перечисления

Синтаксис Automation:

chooseBodiesType = chooseBodies.ChooseBodiesType	Получить свойство(*)
chooseBodies.ChooseBodiesType = chooseBodiesType	Установить свойство (*)

Синтаксис COM

chooseBodiesType = chooseBodies.GetChooseBodiesType()	Получить свойство (**)
chooseBodies.SetChooseBodiesType(chooseBodiesType)	Установить свойство (**)

Примечание:

При ручном указании и автоматическом выборе тел в операции участвуют тела, которые находятся в коллекции тел области применения. Коллекцию можно получить с помощью методов ksChooseBodies::BodyCollection и IChooseBodies::BodyCollection.

IChooseBodies – методы

BodyCollection – Получить указатель на интерфейс массива трехмерных тел

Интерфейс...

Синтаксис Automation:

LPDISPATCH BodyCollection();

Синтаксис COM:

LPBODYCOLLECTION BodyCollection();

Возвращаемое значение:

- указатель на интерфейс массива тел ksBodyCollection или IBodyCollection.

Интерфейсы ksChooseParts, IChooseParts

Справка системы КОМПАС...

kompas.chm::/906_107_2_Oblastq_primenenija_o.htm

Интерфейс области применения для компонентов сборки в сборочной операции.

ksChooseParts	- интерфейс Automation
IChooseParts	- интерфейс COM

Примечание.

Интерфейс позволяет выбирать компоненты сборки, участвующие в операции.

IChooseParts – свойства

ChoosePartsType – Способ определения области применения для компонентов в сборочной операции

Интерфейс...

Тип данных: ksChoosePartsType

Синтаксис Automation:

ChoosePartsType = ChooseParts.ChoosePartsType	Получить свойство(*)
ChooseParts.ChoosePartsType = ChoosePartsType	Установить свойство (*)
ChoosePartsType = ChooseParts.GetChoosePartsType()	Получить свойство (**)
ChooseParts.SetChoosePartsType(ChoosePartsType)	Установить свойство (**)

IChooseParts – методы

PartCollection – Получить указатель на интерфейс массива списка компонентов

Интерфейс...

Синтаксис Automation:

LPDISPATCH PartCollection();

Синтаксис COM:

LPPARTCOLLECTION PartCollection();

Возвращаемое значение:

- указатель на интерфейс массива тел ksPartCollection или IPartCollection.

Интерфейс булевой операции над твердыми телами(Интерфейсы ksAggregateDefinition, IAggregateDefinition)

Справка системы КОМПАС...

`kompas.chm::/730_87_5_Buleva_operacija.htm`

Интерфейс булевой операции над твердыми телами.

ksAggregateDefinition	- интерфейс Automation
IAggregateDefinition	- интерфейс COM

IAggregateDefinition– свойства

BooleanType – Тип операции над телами

Интерфейс...

Тип данных: long

Синтаксис Automation:

BooleanType = iAggregateDefinition.BooleanType	Получить свойство(*)
iAggregateDefinition.BooleanType = BooleanType	Установить свойство (*)
BooleanType = iAggregateDefinition.GetBooleanType()	Получить свойство (**)
iAggregateDefinition.SetBooleanType(BooleanType)	Установить свойство (**)

Типы булевых операций над твердыми телами...

Примечание:

Коллекцию тел можно получить с помощью методов ksAggregateDefinition::BodyCollection и IAggregateDefinition::BodyCollection.

IAggregateDefinition – методы

BodyCollection – Получить указатель на интерфейс массива тел

Интерфейс...

Синтаксис Automation:

LPDISPATCH BodyCollection();

Синтаксис COM:

LPBODYCOLLECTION BodyCollection();

Возвращаемое значение:

- указатель на интерфейс массива тел ksBodyCollection или IBodyCollection.

Графический документ (Интерфейсы – ksDocument2D и IDocument2D)

Интерфейс графического документа системы КОМПАС

Интерфейс событий графического документа; работа с файлами...

Примечание:

Данный интерфейс в автоматизации может быть получен от интерфейса API Kompas KompasObject с помощью методов KompasObject::Document2D, KompasObject::ActiveDocument2D.

ksDocument2D – свойства

OrthoMode – Режим ортогонального черчения

Интерфейс...

Справка системы КОМПАС...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

orthoMode = iDocument2D.orthoMode	Получить свойство (*)
iDocument2D.lookStyle = orthoMode	Установить свойство (*)
orthoMode = iDocument2D.GetOrthoMode()	Получить свойство (**)
iDocument2D.SetOrthoMode(orthoMode)	Установить свойство (**)

Примечание:

Используется только для визуальных документов.

Reference – Указатель на графический документ системы КОМПАС

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iDocument2D.reference	Получить свойство (*)
iDocument2D.reference = ref	Установить свойство (*)
ref = iDocument2D.GetReference()	Получить свойство (**)
iDocument2D.SetReference(ref)	Установить свойство (**)

ksDocument2D – методы

GetDocument2DNotify – Получить источник событий для графического документа

Интерфейс...

Синтаксис Automation:

Document2DNotify* GetDocument2DNotify();

Возвращаемое значение:

- указатель на интерфейс источника событий Document2DNotify.

ksSetLightObjType – Установить тип подсветки объекта (light=1 – красный) или (light=0 – зеленый)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetLightObjType.

Синтаксис Automation:

long ksSetLightObjType(reference ref, long light);

Входные параметры:

ref	- указатель на объект,
light	- light=1 - красный или light=0 - зеленый.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksGetObjectNameByType – Вернуть имя объекта по его типу

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetObjectNameByType.

Синтаксис Automation:

BSTR ksGetObjectNameByType(long type);

Входные параметры:

type	- тип объекта из перечисления DrawingObjectTypeEnum.

Возвращаемое значение:

- Строковое имя типа объекта - отрезок, дуга и т.д..

ksGetObjectNameByTypeW – Вернуть имя объекта по его типу. Множественное число

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetObjectsNameByType.

Синтаксис Automation:

BSTR ksGetObjectsNameByType(long type);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum.

Возвращаемое значение:

- Строковое имя типа объекта - отрезок, дуга и т.д..

GetFragment – Получить указатель на интерфейс фрагмента

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFragment();

Возвращаемое значение:

- указатель на интерфейс фрагмента ksFragment.

GetObject2DNotify – Получить интерфейс источника событий для 2D документов

Интерфейс...

Синтаксис Automation:

ksObject2DNotify * GetObject2DNotify (long objType);

Входные параметры:

objType - объекты, для которых предназначен данный интерфейс событий.

Возвращаемое значение:

- указатель на интерфейс источника событий Object2DNotify.

Примечание:

Значением objType является либо тип объекта (ALL_OBJ...AXISLINE_OBJ, VIEW_OBJ), либо указатель на объект.

- Если задан тип объекта, события генерируются для всех объектов этого типа.
- Если задан указатель на объект, события генерируются только для этого объекта.
- Если задан тип объектов ALL_OBJ, события генерируются для всех объектов.

GetObject2DNotifyResult – Получить интерфейс результатов редактирования объекта графического документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetObject2DNotifyResult.

Синтаксис Automation:

ksObject2DNotifyResult * GetObject2DNotifyResult();

Возвращаемое значение:

- указатель на интерфейс результатов редактирования объекта ksObject2DNotifyResult.

GetSelectionMngNotify – Получить интерфейс источника событий для графических документов – менеджер выделенных объектов

Интерфейс...

Синтаксис Automation:

ksSelectionMngNotify * GetSelectionMngNotify();

Возвращаемое значение:

- указатель на интерфейс источника событий SelectionMngNotify.

GetSpecification – Получить указатель на интерфейс для работы с объектами спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSpecification();

Возвращаемое значение:

- указатель на интерфейс для работы с объектами спецификации ksSpecification.

GetStamp – Получить интерфейс ksStamp основной надписи документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetStamp();

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

GetStampEx – Получить указатель на интерфейс основной надписи чертежа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetStampEx (long sheetNumb);

Входные параметры:

sheetNumb - номер листа, начиная с 1.

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

ksAssociationViewMatrix3D – Создать матрицу ассоциативного вида

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksAssociationViewMatrix3D.

Синтаксис Automation:

VARIANT ksAssociationViewMatrix3D (long view);

Входные параметры:

view - указатель на ассоциативный вид.

Возвращаемое значение:

SAFEARRAY double (VT_ARRAY | VT_R8)

В массиве будут лежать 16 элементов, которые представляют собой матрицу размера 4x4.

Примечание:

Если указатель на вид не задан (view = 0), то метод будет выполняться для текущего вида.

ksCalcRasterScale – Рассчитать масштаб для вставки растра в прямоугольник заданных габаритов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCalcRasterScale.

Синтаксис Automation:

double ksCalcRasterScale (BSTR fileName, double w, double h);

Входные параметры:

fileName	- полный путь к файлу растра,
w	- ширина габаритного прямоугольника,
h	- высота габаритного прямоугольника.

Возвращаемое значение:

масштаб для вставки растра в прямоугольник	- в случае успеха,
0	- в случае неудачи.

ksChangeLeader – Создать линию выноску для обозначения изменения

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

long ksChangeLeader(LPDISPATCH leaderParam);

Входные параметры:

leaderParamr	- указатель на интерфейс параметров обозначения изменения ksChangeLeaderParam.

Возвращаемое значение:

указатель reference обозначения изменения	- в случае успешного завершения.

ksChangeObjectInLibRequest – Изменить фантом или компоненты команд

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksChangeObjectInLibRequest.

Синтаксис Automation:

long ksChangeObjectInLibRequest (LPDISPATCH info,

LPDISPATCH phantom);

Входные параметры:

info	- указатель на интерфейс ksRequestInfo,
phantom	- указатель на интерфейс ksPhantom.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Во время работы методов ksDocument2D::ksCursor и ksDocument2D::ksPlacement может возникнуть необходимость изменить фантом или компоненты команд. Функция ksChangeObjectInLibRequest позволяет передать изменения в цикл ksCursor и ksPlacement. Это может понадобиться для отработки команд пользовательских инструментальных панелей.

Предварительно нужно убедиться, что вызов ksCursor и требование к ksCursor передаются в одном документе. Если какой-либо указатель равен 0, соответствующий параметр не используется.

ksChangeObjectsOrder – Изменить порядок отрисовки объектов чертежа

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksChangeObjectsOrder.

Синтаксис Automation:

long ksChangeObjectsOrder (long group, long obj, int orderType);

Входные параметры:

group - указатель на группу объектов, для которой требуется изменить последовательность отрисовки,
obj - указатель на объект, относительно которого изменяется отрисовка группы,
orderType - Тип изменения порядка объектов.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksColouringEx – Создать фоновую заливку цветом

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksColouringEx.

Синтаксис Automation:

long ksColouringEx (long color, reference group);

Входные параметры:

color - цвет заливки,
group - группа, образующая границу заливки.

Возвращаемое значение:

указатель на заливку - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если color = -1 или 0xFFFFFFFF,то создается заливка цветом фона документа.

ksCommandWindow – Запрос к системе на создание окна с деревом команд

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CommandWindow.

Синтаксис Automation:

long ksCommandWindow (LPDISPATCH info);

Входной параметр:

info - указатель на интерфейс ksRequestInfo,

Возвращаемое значение:

- идентификатор выбранной команды, определенный в файле ресурсов, - порядковый номер в строке команд.	- в случае успешного завершения,
-1	- если команда не выбрана.

Примечание:

Функция создает окно с деревом команд, определяемых строкой команд в интерфейсе ksRequestInfo, или идентификатором меню из файла ресурсов. Команды в строке разделены пробелом или восклицательным знаком. Функции обратной связи передается управление после выбора команды или отказа.

В случае, если callBack равен NULL, управление из ksCommandWindow возвращается немедленно после выбора команды. В противном случае управление вернется, если пользователь закроет окно, или функция обратной связи вернет FALSE.

ksCopyObjEx – Копировать объект

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCopyObjEx.

Синтаксис Automation:

long ksCopyObjEx (LPDISPATCH param);

Входные параметры:

param	- указатель на интерфейс параметров копирования объекта ksCopyObjectParam.

Возвращаемое значение:

указатель на объект или группу объектов	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция позволяет копировать объект (объект вида, вид, группу, слой) в новую точку с возможностью задания масштабирования копии и поворота ее вокруг базовой точки.
2. При копировании одиночного объекта новый объект создается на текущем слое текущего вида.
3. При копировании группы объектов слой сохраняется.
4. При копировании одиночного многослойного макрообъекта он перестает быть многослойным. Макро и входящие в него объекты переносятся на текущий слой. Для копирования многослойного макро с сохранением многослойности требуется добавить его в группу.

ksCreateViewObject – Создать объект заданного типа, используя визуальный процесс создания объекта

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksCreateViewObject.

Синтаксис Automation:

long ksCreateViewObject (long type);

Входной параметр:

type - тип объекта.

Типы объектов и интерфейсы...

Возвращаемое значение:

указатель на созданный объект - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Метод распространяется на все объекты вида, кроме слоя, контура, нетипизированного макроэлемента, вставленного фрагмента и дуги эллипса.

Так как система может работать только с одним процессом, то нужно завершить другие "процессные" методы: ksDocument2D::ksCursor, ksDocument2D::ksPlacement, ksDocument2D::ksCommandWindow.

ksCursor – Запрос к системе на получение точки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Cursor.

Синтаксис Automation:

long ksCursor (LPDISPATCH info,

double* x,

double* y,

LPDISPATCH phantom);

Входные параметры:

info - указатель на интерфейс ksRequestInfo,
phantom - указатель на интерфейс ksPhantom.

Выходные параметры:

x, y - возвращаемые координаты точки.

Возвращаемое значение:

-1 - если точка указана.
идентификатор выбранной команды,
определенный в файле ресурсов,
порядковый номер в строке команд.

Примечание:

Интерактивный ввод точки или определение варианта действия. Возможные варианты (команды) задаются в строке commands интерфейса ksRequestInfo и разделяются восклицательными знаками или пробелами.

Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений.

Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.

ksCursorEx – Запрос к системе на получение координат точки или определение варианта действия

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CursorEx.

Синтаксис Automation:

long ksCursorEx (LPDISPATCH info,
double* x,
double* y,
LPDISPATCH phantom,
LPDISPATCH processParam);

Входные параметры:

info - указатель на интерфейс параметров запроса к системе ksRequestInfo,
phantom - указатель на интерфейс фантома ksPhantom,
processParam - указатель на интерфейс параметров процесса IProcessParam.

Выходные параметры:

x, y - возвращаемые координаты точки.

Возвращаемое значение:

-1 - если указана точка,
0 - отказ (Esc), идентификатор выбранной команды из командной строки или меню, определенный в файле ресурсов.

Примечание:

Возможные варианты (команды) задаются в строке commands интерфейса ksRequestInfo и разделяются восклицательными знаками или пробелами. Если вместо

строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если функция обратной связи ksRequestInfo не определена, то действие метода прекращается после первого шага.

ksDestroyObjects – Разрушить присланные составные объекты

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksDestroyObjects.

Синтаксис Automation:

long ksDocument2D::ksDestroyObjects (long p)

Входной параметр:

p	- указатель на объект, вид, слой или группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Метод разрушает не только составные объекты (макроэлементы, вставки фрагментов, эквидистанты, прямоугольники, контуры), но и виды (если у них есть связь с моделью).
2. Метод работает только для документов, открытых в «видимом» режиме.

ksDrawKompasDocument – Отрисовать документ системы КОМПАС как слайд в присланном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawKompasDocument.

Синтаксис Automation:

long ksDrawKompasDocument (long HWindow,

BSTR docFileName);

Входные параметры:

Hwindow	- "несущее" окно,
docFileName	- полное имя файла документа.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Действие метода распространяется также на фрагменты и модели из библиотек. При этом имя файла должно иметь вид "c:\gr\lib1.l3d|детали|литье|фланец", где:
c:\gr\lib1.l3d - имя файла библиотеки,

|детали|литье| - разделы, подразделы внутри библиотеки,

фланец - имя фрагмента или модели.

ksDrawKompasGroup – Отрисовать группу как слайд в присланном окне

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDrawKompasGroup.

Синтаксис Automation:

long ksDrawKompasGroup (long HWindow, long gr);

Входные параметры:

Hwindow	- "несущее" окно,
gr	- указатель на группу.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksFindObj – Найти ближайший к заданной точке объект вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - FindObj.

Синтаксис Automation:

long ksFindObj (double x, double y, double limit);

Входные параметры:

x, y	- координаты точки,
limit	- размер стороны квадрата-"ловушки" с центром в указанной точке.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Зона поиска - квадрат с указанной стороной и центром.
2. Если значение limit равно MAXDOUBLE, выполняется поиск ближайшего к указанной точке объекта без ограничения зоны поиска.
3. Метод ksFindObj можно использовать для поиска объектов внутри макроэлемента.
4. Для этого макроэлемент должен быть открыт на редактирование функцией ksDocument2D::ksOpenMacro.

Пример:

```
doc2D.ksOpenMacro(obj);
long obj_i=doc2D.ksFindObj(xn, xn, AXIS_FIND_SENS);
if( doc2D.ksExistObj(obj_i) )
doc2D.ksLightObj(obj_i, 1);
doc2D.ksEndObj();
```

При этом, если для макроэлемента задана система координат (см. ksDocument2D::ksPlacement), то координаты в ksFindObj должны передаваться в этой системе координат.

Пересчитать координаты в систему координат макроэлемента можно через функцию ksDocument2D::ksPointIntoMtr. Для этого нужно функцией ksDocument2D::ksMtr создать матрицу преобразования координат.

Пример:

```
doc2D.ksMtr(xp, yp, ang,? 1, 1);
doc2D.ksPointIntoMtr(x, y, &xn, &yn);
doc2D.ksEndMtr();
```

ksGetCursorPosition – Получить координаты курсора

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCursorPosition.

Синтаксис Automation:

long ksGetCursorPosition (double* x, double* y, long type);

Входной параметр:

type - признак, какие координаты возвращать:
0 - без учета привязок,
1 - с учетом привязок.

Выходные параметры:

x, y - координаты курсора (в миллиметрах).

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetCursorLimit – Получить радиус окружности, вписанной в "ловушку" курсора

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetCursorLimit.

Синтаксис Automation:

double ksGetCursorLimit();

Возвращаемое значение:

- радиус окружности (мм), вписанной в "ловушку" курсора.

ksGetDocumentPagesCount – Получить количество листов документа

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetDocumentPagesCount.

Синтаксис Automation:

long ksGetDocumentPagesCount();

Возвращаемое значение:

- количество листов документа.

ksGetLeaderShelfLength – Получить длину полки линии-выноски и координаты ее конечной точки

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetLeaderShelfLength.

Синтаксис Automation:

double ksGetLeaderShelfLength (long leader, double *x, double *y);

Входные параметры:

leader - указатель на линию-выноску.

Выходные параметры:

x - координата конечной точки полки линии выноски по оси X,
y - координата конечной точки полки линии выноски по оси Y.

ksGetObjectStyle – Получить стиль для объекта 2D документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetObjectStyle.

Синтаксис Automation:

long ksGetObjectStyle(long obj);

Входные параметры:

obj - указатель reference объекта 2D.

Примечание:

Метод позволяет получить стиль для кривых и эквидистанты.

ksGetShelfPoint – Получить координаты начала и конца выносной полки и ножки

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetShelfPoint.

Синтаксис Automation:

BOOL ksGetShelfPoint(long p, long index, double *x, double *y, long paramType);

Входные параметры:

p - указатель на объект (размер или линию-выноску),
index - индекс точки.
Значения индекса:
Для линии с ножкой:

```
    1 __ 2
    /
  0 /
```

Для линии без ножки (с выносной полкой на продолжении выносной линии):

```
  0 __ 1
```

Точка 0 примыкает к выносной линии.
paramType - тип параметров, задает систему координат:
ALLPARAM - в системе координат владельца;
SHEET_ALLPARAM - в системе координат листа;
VIEW_ALLPARAM - в системе координат вида.

Выходные параметры:

x, y - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ksHatchByParam – Создать штриховку с заданными параметрами

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksHatch.

Синтаксис Automation:

long ksHatchByParam (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс ksHatchParam.

Возвращаемое значение:

указатель на штриховку - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Объекты границы штриховки лежат во временной группе ksHatchParam::boundaries. После выполнения метода ksDocument2D::ksHatchByParam временная группа прекращает свое существование.

ksInitFilePreviewFunc – Инициализировать адрес пользовательской функции просмотра пользовательского файла

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksInitFilePreviewFunc.

Синтаксис Automation:

long ksInitFilePreviewFunc (BSTR funcName, long hInst);

Входные параметры:

funcName	- имя функции,
hInst	- идентификатор приложения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksInitFilePreviewFuncW – Инициализировать адрес пользовательской функции просмотра пользовательского файла (Unicode)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksInitFilePreviewFuncW.

Синтаксис Automation:

long ksDocument2D::ksInitFilePreviewFuncW(BSTR funcName, long hInst, LPDISPATCH dispatchOCX);

Входные параметры:

funcName	- имя функции,
hInst	- идентификатор приложения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksIsActiveProcessRunnig – Проверить, запущен ли в текущем графическом документе процесс построения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsActiveProcessRunnig.

Синтаксис Automation:

long ksIsActiveProcessRunnig();

Возвращаемое значение:

1 - процесс запущен,
0 - процесс не запущен.

ksIsCursorOrPlacementDocument – Проверить, запущен ли в текущем графическом документе процесс ksDocument2D_ksCursor или ksDocument2D_ksPlacement

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsCursorOrPlacementDocument.

Синтаксис Automation:

long ksIsCursorOrPlacementDocument();

Возвращаемое значение:

1 - процесс запущен,
0 - процесс не запущен.

ksIsCurveClosed – Проверить, замкнута ли указанная кривая

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsCurveClosed.

Синтаксис Automation:

long ksIsCurveClosed(long p);

Входной параметр:

p - указатель на кривую.

Возвращаемое значение:

1 - кривая замкнута,
0 - кривая разомкнута,

-1 - в случае неудачи.

ksIsPointInsideContour – Проверить положение точки относительно кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIsPointInsideContour.

Синтаксис Automation:

long ksIsPointInsideContour (long Reference,

double x,

double y,

double precision);

Входные параметры:

Reference	- указатель на контур,
x, y	- координаты точки,
precision	- точность проверки (от 1 до 1E-6).

Возвращаемое значение:

0	- в случае неудачи,
1	- точка вне контура,
2	- точка на контуре,
3	- точка внутри контура.

ksIsSlaveSpcOpened – Проверить, открыто ли окно спецификации в Slave режиме

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksIsSlaveSpcOpened.

Синтаксис Automation:

long ksIsSlaveSpcOpened();

Синтаксис COM:

int ksIsSlaveSpcOpened();

Возвращаемое значение:

0	- окно спецификации в Slave режиме не открыто,
1	- окно спецификации в Slave режиме открыто,

2 - окно спецификации в Slave режиме открыто и активно.

ksLengthFromMtr – Пересчитать длину из локальной системы координат в СК вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksLengthFromMtr.

Синтаксис Automation:

long ksLengthFromMtr (double* len);

Входной и Выходной параметр:

len - длина.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр len является одновременно и входным, и выходным. На входе он представляет собой длину в локальной СК, а в результате работы функции преобразуется в длину в СК вида.

ksLengthIntoMtr – Пересчитать длину из системы координат вида в локальную СК

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksLengthIntoMtr.

Синтаксис Automation:

long ksLengthIntoMtr (double* len);

Входной и Выходной параметр:

len - длина.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр len является одновременно и входным, и выходным. На входе он представляет собой длину в СК вида, а в результате работы функции преобразуется в длину в локальной СК.

ksMakeEncloseContours – Получить указатель на группу объектов, охватывающих заданную точку

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMakeEncloseContours.

Синтаксис Automation:

long ksMakeEncloseContours (long gr,

double x,

double y);

Входные параметры:

gr	- временная группа или 0, если требуются контуры в текущем виде,
x, y	- координаты точки внутри охватывающих контуров.

Возвращаемое значение:

указатель на временную группу контуров	- в случае удачного завершения,
0	- если контуров не нашлось или в случае неудачи.

Примечание:

Возвращаемая группа принадлежит указанной временной группе или текущему виду в целом.

ksMakeEncloseContoursEx – Получить указатель на группу объектов, охватывающих заданную точку

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMakeEncloseContoursEx.

Синтаксис Automation:

long ksMakeEncloseContoursEx(long gr,
double x,
double y,
BOOL forHatch);

Входные параметры:

gr	- временная группа или 0, если требуются контуры в текущем виде,
x, y	- координаты точки внутри охватывающих контуров,
forHatch	- признак создания контура для штриховки.

Возвращаемое значение:

указатель на временную группу контуров	- в случае удачного завершения,
0	- если контуров не нашлось или в случае неудачи.

Примечание:

forHatch == 1 - при создании контуров для штриховки используются объекты со стилем основная и утолщенная, объекты других стилей не учитываются.

forHatch == 0 - при создании контура с использованием любых стилей линий.

ksMovePoint – Сдвинуть точку в указанном направлении на указанное расстояние

Интерфейс...

Аналог данного метода при использовании API экспортных функций - MovePoint.

Синтаксис Automation:

BOOL ksMovePoint (double* x,

double* y,

double ang,

double len);

Входные параметры:

x, y	- координаты точки,
ang	- угол вектора сдвига в градусах,
len	- длина вектора сдвига.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Параметры x и y являются одновременно и входными, и выходными. На входе они представляют собой начальные координаты точки, а в результате работы метода они преобразуются в координаты точки после сдвига.

ksParametrizeObjects – Параметризовать группу объектов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksParametrizeObjects.

Синтаксис Automation:

long ksParametrizeObjects (long group, LPDISPATCH par);

Входные параметры:

group	- указатель на группу объектов,
par	- параметры параметризации группы объектов ksParametrizationParam.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

ksPhantomShowHide – Сделать фантом видимым или невидимым

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPhantomShowHide.

Синтаксис Automation:

long ksPhantomShowHide (BSTR show);

Входной параметр:

show	- признак отрисовки фантома: 1 - включить, 0 - выключить.

Возвращаемое значение:

-1	- в случае удачного завершения,
0	- в случае неудачи.

ksPlacement – Запрос к системе на получение точки и угла

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Placement.

Синтаксис Automation:

long ksPlacement (LPDISPATCH info,

double* x,

double* y,

double* Angle,

LPDISPATCH phantom);

Входные параметры:

info	- указатель на интерфейс ksRequestInfo,
phantom	- указатель на интерфейс ksPhantom.

Выходные параметры:

x, y	- возвращаемые координаты точки,
angle	- возвращаемое значение угла.

Возвращаемое значение:

-1	- если точка указана.

идентификатор выбранной команды, определенный в файле ресурсов,
порядковый номер в строке команд.

Примечание:

Интерактивный ввод точки и угла или определение варианта действия. Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами.

Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений.

Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.

ksPlacementEx – Запрос к системе на получение координат точки и угла

Интерфейс...

Аналог данного метода при использовании API экспортных функций - PlacementEx.

Синтаксис Automation:

long ksPlacementEx (LPDISPATCH info,
double* x,
double* y,
double* angle,
LPDISPATCH phantom,
LPDISPATCH processParam);

Входные параметры:

info	- указатель на интерфейс параметров запроса к системе ksRequestInfo,
phantom	- указатель на интерфейс фантома ksPhantom,

processParam - указатель на интерфейс параметров процесса IProcessParam.

Выходные параметры:

x, y - возвращаемые координаты точки.
angle - возвращаемое значение угла.

Возвращаемое значение:

-1 - если указана точка,
0 - отказ (Esc), идентификатор выбранной команды из командной строки или меню, определенный в файле ресурсов.

Примечание:

Возможные варианты (команды) задаются в строке commands интерфейса ksRequestInfo и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если функция обратной связи ksRequestInfo не определена, то действие метода прекращается после первого шага.

ksPointFromMtr – Пересчитать координаты точки из локальной системы координат в СК вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPointFromMtr.

Синтаксис Automation:

long ksPointFromMtr (double x, double y, double* xn, double* yn);

Входные параметры:

x, y - координаты точки в локальной системе координат.

Выходные параметры:

xn, yn - координаты точки в СК вида.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksPointIntoMtr – Пересчитать координаты точки из системы координат вида в локальную СК

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPointIntoMtr.

Синтаксис Automation:

long ksPointIntoMtr (double x,

double y,

double* xn,

double* yn);

Входные параметры:

x, y - координаты точки в системе координат вида.

Выходные параметры:

xn, yn - координаты точки в локальной СК.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksPoint3DToAssociationView – Преобразовать координаты 3D точки в координаты ассоциативного вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPoint3DToAssociationView.

Синтаксис Automation:

BOOL ksPoint3DToAssociationView (long view,
double x3D, double y3D, double z3D,
double * x2D, double * y2D);

Входные параметры:

view - указатель на ассоциативный вид,
x3D, y3D, z3D - координаты 3D точки.

Выходные параметры:

x2D, y2D - координаты 3D точки в ассоциативном виде.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Если указатель на вид не задан (view = 0), то функция будет выполняться для текущего вида.

ksPolylineByParam – Создать ломаную линию

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPolyline.

Синтаксис Automation:

long ksPolylineByParam (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс ломаной линии ksPolylineParam.

Возвращаемое значение:

указатель на ломаную линию - в случае удачного завершения,
0 - в случае неудачи.

ksReDrawDocPart – Перерисовать часть графического документа

Интерфейс...

Синтаксис Automation:

int ksReDrawDocPart (IDispatch* rect,

long view);

Входные параметры:

rect - указатель на интерфейс ksRectParam параметров прямоугольника, ограничивающего перерисовываемую часть,
view - указатель на вид.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksRemoteElement – Создать объект "Выносной элемент"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksRemoteElement.

Синтаксис Automation:

long ksRemoteElement (LPDISPATCH par);

Входные параметры:

par - указатель на интерфейс ksRemoteElementParam параметров выносного элемента.

Возвращаемое значение:

указатель на объект "Выносной элемент" - в случае успешного завершения.

ksSaveToDXF – Сохранить документ в формате DXF

Интерфейс...

Синтаксис Automation:

BOOL ksSaveToDXF (LPCTSTR dxfFileName);

Входные параметры:

dxfFileName - имя файла для записи в формате DXF.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ksSetMaterialParam – Установить параметры материала в чертеже

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksSetMaterialParam.

Синтаксис Automation:

long ksSetMaterialParam (LPDISPATCH material, double density);

Входные параметры:

material - наименование материала, указатель на интерфейс динамического массива ksDynamicArray строк текста (TEXT_LINE_ARR),
density - плотность (г/куб.мм).

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

ksSetMixDlgMaterialParam – Установить параметры материала для диалога МЦХ

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksSetMixDlgMaterialParam.

Синтаксис:

BOOL ksSetMixDlgMaterialParam(BSTR material, double density);

Входные параметры

material	- обозначение материала,
density	- плотность.

Возвращаемое значение:

TRUE	- в случае успеха.

ksSetObjectStyle – Установить стиль для объекта 2D документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetObjectStyle.

Синтаксис Automation:

BOOL ksSetObjectStyle(long obj, long style);

Входные параметры:

obj	- указатель reference объекта 2D,
style	- номер стиля.

Примечание:

Метод позволяет установить стиль для кривых и эквидистанты.

ksSheetToView – Пересчитать точку из системы координат листа в СК текущего вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SheetToView.

Синтаксис Automation:

long ksSheetToView (double x,

double y,

double* outX,

double* outY);

Входные параметры:

x, y - координаты точки в системе координат листа.

Выходные параметры:

outX, outY - координаты точки в системе координат вида.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksSpecificationOnSheet – Установить признак размещения спецификации на листе

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSpecificationOnSheet.

Синтаксис Automation:

long ksSpecificationOnSheet (short onSheet);

Входной параметр:

onSheet - признак размещения спецификации на листе
1 - включено,
0 - выключено.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksTextEx – Создать многострочный текст

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTextEx.

Синтаксис Automation:

long ksTextEx (LPDISPATCH txtParam, long align);

Входные параметры:

txtParam - указатель на интерфейс ksTextParam параметров текста,

align	- выравнивание текста: 0 - слева, 1 - по центру, 2 - справа, 3 - по ширине, -1 установить выравнивание как у стиля текста.

Возвращаемое значение:

указатель на созданный текст	- в случае успеха,
0	- в случае неудачи.

ksViewToSheet – Пересчитать точку из СК текущего вида в СК листа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ViewToSheet.

Синтаксис Automation:

long ksViewToSheet (double x,

double y,

double* outX,

double* outY);

Входные параметры:

x, y	- координаты точки в системе координат вида.

Выходные параметры:

outX, outY	- координаты точки в системе координат листа.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

RasterFormatParam – Получить указатель на интерфейс, определяющий параметры записи в растровый формат

Интерфейс...Справка системы КОМПАС...

Синтаксис Automation:

LPDISPATCH RasterFormatParam();

Возвращаемое значение:

указатель на интерфейс ksRasterFormatParam	- в случае успешного завершения,
NULL	- в случае неудачи.

SaveAsToRasterFormat – Сохранить документ в растровом формате

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSaveAsToRasterFormat.

Синтаксис Automation:

BOOL SaveAsToRasterFormat (BSTR fileName,

LPDISPATCH rasterPar);

Входные параметры:

fileName	- полное имя файла документа,
rasterPar	- указатель на интерфейс ksRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечания:

Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

SaveAsToUncompressedRasterFormat – Сохранить документ в растровом формате без сжатия

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSaveAsToUncompressedRasterFormat.

Синтаксис Automation:

BOOL SaveAsToUncompressedRasterFormat (BSTR fileName, IDispatch* rasterPar);

Входные параметры:

fileName	- имя файла документа, должно быть задано полное имя файла,
rasterPar	- указатель на интерфейс ksRasterFormatParam, определяющий параметры конвертации в растровый формат.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Метод позволяет сохранить документ в растровом формате без сжатия, с заданными свойствами: цветом, форматом (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) и т.п. Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

   Подробнее см. описание ksRasterFormatParam.

Аннотационные объекты

ksAnnArcByPoint – Создать аннотационную дугу по точкам

Интерфейс...

Аналог данного метода при использовании API экспортных функций - AnnArcByPoint.

Синтаксис Automation:

long ksAnnArcByPoint (double xc,

double yc,

double rad,

double x1,

double y1,

double x2,

double y2,

short direction,

short term1,

short term2,

long style);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
x1, y1	- координаты начальной точки дуги,
x2, y2	- координаты конечной точки дуги,
direction	- направление отрисовки дуги: 1 - против часовой стрелки, -1 - по часовой стрелке,
term1, term2	типы значков на концах дуги,
style	стиль линии.

Системные типы значков...

Системные стили линий...

Возвращаемое значение:

указатель на аннотационную дугу	- в случае удачного завершения,
0	- в случае неудачи.

ksAnnCircle – Создать объект "Аннотационная окружность"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnCircle.

Синтаксис Automation:

long ksAnnCircle(double xc,
double yc,
double rad,
long style);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
style	- стиль линии.

Возвращаемое значение:

указатель на аннотационную окружность	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnEllipse – Создать объект "Аннотационный эллипс"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnEllipse.

Синтаксис Automation:

long ksAnnEllipse(LPDISPATCH par);

Входные параметры:

par - указатель на интерфейс параметров эллипса ksEllipseParam,

Возвращаемое значение:

указатель на аннотационный эллипс - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnEllipseArc – Создать объект "Аннотационная дуга эллипса"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnEllipseArc.

Синтаксис Automation:

long ksAnnEllipseArc(LPDISPATCH par,
short term1,
short term2);

Входные параметры:

par - указатель на интерфейс параметров параметров дуги эллипса ksEllipseArcParam,
term1, - типы значков на концах ломаной.
term2

Возвращаемое значение:

указатель на аннотационную дугу эллипса - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnLineSeg – Создать аннотационный отрезок

Интерфейс...

Аналог данного метода при использовании API экспортных функций - AnnLineSeg.

Синтаксис Automation:

long ksAnnLineSeg (double x1,

double y1,

double x2,

double y2,

short term1,

short term2,

long style);

Входные параметры:

x1, y1	- координаты первой точки отрезка,
x2, y2	- координаты второй точки отрезка,
term1, term2	- типы значков на концах линии,
style	- стиль линии.

Системные типы значков...

Системные стили линий...

Возвращаемое значение:

указатель на аннотационный отрезок	- в случае удачного завершения,
0	- в случае неудачи.

ksAnnParEllipseArc – Создать объект "Аннотационная дуга эллипса"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnParEllipseArc.

Синтаксис Automation:

long LIB_FUNC ksAnnParEllipseArc(LPDISPATCH par,
short term1,
short term2);

Входные параметры:

par	- указатель на интерфейс параметров параметров дуги эллипса ksEllipseArcParam1,
term1, term2	- типы значков на концах ломаной.

Возвращаемое значение:

указатель на аннотационную дугу эллипса	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnPoint – Создать объект "точка" с аннотационной точкой привязки

Интерфейс...

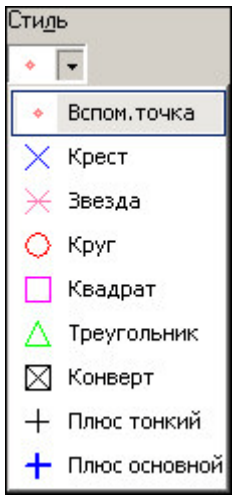
Аналог данного метода при использовании API экспортных функций - ksAnnPoint.

Синтаксис Automation:

long ksAnnPoint(double x,
double y,
long style);

Входные параметры:

x, y	- координаты точки,
style	- стиль отрисовки точки: 0 - вспом. точка, 1 - плюс тонкий, 2 - крест, 3 - квадрат, 4 - треугольник, 5 - круг, 6 - звезда, 7 - конверт, 8 - плюс основной.



Возвращаемое значение:

указатель на точку	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.
3. Координаты точки привязки аннотационные, не зависят от масштаба вида.

ksAnnPolyline – Создать аннотационную ломаную линию

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnPolyline.

Синтаксис Automation:

long ksAnnPolyline (long style,
short term1,
short term2);

Входные параметры:

style	- стиль линии,
term1, term2	- типы значков на концах ломаной.

Возвращаемое значение:

указатель на аннотационную ломаную линию	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. ksDocument2D::ksAnnPolyline - составной объект. Объекты ksDocument2D::ksPoint, вводимые между операторами ksDocument2D::ksAnnPolyline и ksDocument2D::ksEndObj, принадлежат аннотационной ломаной. Метод ksDocument2D::ksEndObj возвращает указатель на аннотационную ломаную линию. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро аннотационные объекты превращаются в геометрические.

ksAnnPolylineEx – Создать объект "Аннотационная ломаная линия" по интерфейсу параметров

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnPolylineEx

Синтаксис Automation:

long ksAnnPolylineEx(LPDISPATCH par,
short term1,
short term2)

Входные параметры:

par	- указатель на интерфейс параметров ломаной ksPolylineParam,
term1, term2	- типы на концах ломаной.

Возвращаемое значение:

указатель на аннотационную ломаную	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnTextEx – Создать объект "Текст с аннотационной точкой привязки"

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAnnTextEx.

Синтаксис Automation:

long ksAnnTextEx(LPDISPATCH txtParam,
long align);

Входные параметры:

txtParam	- указатель на интерфейс параметров текста,
align	- выравнивание текста: -1 - установить выравнивание как у стиля текста, 0 - слева, 1 - по центру, 2 - справа, 3 - на всю ширину.

Возвращаемое значение:

указатель на текст	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.
3. Координаты точки привязки аннотационные не зависят от масштаба вида.

Составные объекты

ksAddObjectToMacro – Добавить объект, слой, вид или группу объектов в макроэлемент

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksAddObjectToMacro.

Синтаксис Automation:

long ksAddObjectToMacro (long macro, long obj);

Входные параметры:

macro	- указатель на макроэлемент,
obj	- указатель на добавляемый объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksContour – Создать контур

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Contour.

Синтаксис Automation:

long ksContour (long style);

Входной параметр:

style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Контур - составной объект. Объекты, вводимые между методами ksDocument2D::ksContour и ksDocument2D::ksEndObj, принадлежат контуру.
2. В контуре могут лежать отрезки, дуги, окружности (они превращаются в дуги), сплайны.
3. Конечная точка предыдущего звена должна совпадать с начальной точкой последующего звена.
4. ksDocument2D::ksEndObj возвращает указатель на контур.

ksDuplicateBoundaries – Получить временную группу контуров, задающих границы штриховки или заливки

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksDuplicateBoundaries.

Синтаксис Automation:

long ksDuplicateBoundaries (long reference);

Входной параметр:

reference	- указатель на штриховку или заливку.

Возвращаемое значение:

указатель на временную группу контуров, задающих границы штриховки или заливки	- в случае удачного завершения,
0	- в случае неудачи.

ksEditMacroMode – Получить режим работы метода библиотеки (создание нового или редактирование существующего макроэлемента)

Интерфейс..

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - EditMacroMode.

Синтаксис Automation:

long ksEditMacroMode();

Возвращаемое значение:

указатель на редактируемый макроэлемент	- при редактировании макроэлемента,
0	- при создании макроэлемента.

ksEndObj – Завершить создание комплексного объекта

Интерфейс...Аналог данного метода при использовании API экспортных функций - EndObj.

Синтаксис Automation:

long ksEndObj();

Возвращаемое значение:

указатель на завершенный объект (контур, макроэлемент и т.д.)	- в случае удачного завершения,
0	- в случае неудачи.

ksGetMacroParamSize – Получить размер памяти параметров указанного макроэлемента

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetMacroParamSize.

Синтаксис Automation:

long ksGetMacroParamSize (long ref);

Входной параметр:

ref	- указатель на макроэлемент.

Возвращаемое значение:

размер памяти	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

При ref = 0 выдается размер памяти параметров макроэлемента, редактирование которого производится в данный момент (если таковой имеется).

ksGetMacroWaitDblClickEdit – Получить режим ожидания DblClick при редактировании макроэлемента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetMacroWaitDblClickEdit.

Синтаксис Automation:

long ksGetMacroWaitDblClickEdit (reference ref);

Входные параметры:

ref	- указатель на макроэлемент.

Возвращаемое значение:

1	- режим ожидания DblClick включен,
0	- режим ожидания DblClick выключен.

Примечание:

Макроэлемент, созданный библиотекой, может редактироваться двумя способами:

- по первому клику мыши - через характерные точки и\или интерфейс внешних воздействий;
- по двойному клику - через команду библиотеки.

Если время инициализации характерных точек сравнимо с двойным кликом, то редактирование через команду библиотеки становится невозможно. Во время редактирования макроэлемента, при включенном режиме ожидания DblClick, после первого щелчка по макроэлементу КОМПАС будет ожидать повторного нажатия в течении времени, установленного в настройках "Скорость выполнения двойного щелчка".

При выключенном режиме ожидания DblClick пауза отсутствует, и нажатие всегда интерпретируется как одинарное нажатие.

ksMacro – Создать новый макроэлемент

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Macro.

Синтаксис Automation:

long ksMacro (short type);

Входной параметр:

type	- тип макроэлемента: 0 - объединяет объекты текущего слоя, 1- включаемые объекты могут принадлежать разным слоям.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Макроэлемент - составной объект.

Объекты вида, вводимые между методами ksDocument2D::ksMacro и ksDocument2D::ksEndObj, принадлежат макроэлементу. ksDocument2D::ksEndObj возвращает указатель на макроэлемент.

ksOpenMacro – Открыть макроэлемент для редактирования

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksOpenMacro.

Синтаксис Automation:

long ksOpenMacro (long macro);

Входной параметр:

macro	- указатель на макроэлемент.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Когда макроэлемент открыт для редактирования; можно добавлять в его состав новые объекты вида, использовать метод ksDocument2D::ksFindObj.

Режим редактирования закрывается методом ksDocument2D::ksEndObj;

В режиме редактирования открыт доступ к внутренним объектам макроэлементов. Теперь при создании макроэлемента или при добавлении к макроэлементу новых объектов вида выдается указатель на внутренние объекты, как и на самостоятельные объекты вида.

Допускается вводить пустой макроэлемент. Пустой макроэлемент является вырожденным объектом. Он существует только в период редактирования документа. После записи документа, в котором находился пустой макроэлемент, этот макроэлемент будет удален. Макроэлемент считается нормальным, если в нем находится хотя бы один внутренний объект.

Все методы работы с указателями на объекты допускают работу с внутренними объектами макроэлемента. Это методы:

ksDocument2D::ksDeleteObj

ksDocument2D::ksMoveObj

ksDocument2D::ksLightObj

ksDocument2D::ksTransformObj

ksDocument2D::ksGetObjParam

ksDocument2D::ksSetObjParam

ksDocument2D::ksRotateObj

ksDocument2D::ksCopyObj

ksDocument2D::ksSymmetryObj

ksDocument2D::ksEditViewObject

ksDocument2D::ksGetObjGabaritRect

ksDocument2D::ksDecomposeObj

ksGetMacroParam – Получить параметры макроэлемента

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetMacroParam.

Синтаксис Automation:

long ksGetMacroParam (long ref, LPDISPATCH userPars);

Входные параметры:

ref	- указатель на макроэлемент,
userPars	- указатель на интерфейс ksUserParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Дополнительные параметры редактирования предварительно задаются функцией ksDocument2D::ksSetMacroParam.
2. При ref = 0 выдаются параметры макроэлемента, редактирование которого производится в данный момент (если таковой имеется).
3. Способ получения пользовательских данных (userPars) должен совпадать со способом их сохранения (void*, SAFEARRAY, или DynamicArray) через ksDocument2D::ksSetMacroParam

- Если пользовательские данные были сохранены через SAFEARRAY (userPars), то перед их получением нужно создать SAFEARRAY соответствующего размера (размер данных можно определить при помощи ksDocument2D::ksGetMacroParamSize).
- Если пользовательские данные были сохранены через ksUserParam::SetUserArray, то перед их получением нужно создать UserArray, аналогичный по структуре используемому при сохранении, и передать его в ksUserParam::SetUserArray.

ksGetMacroPlacement – Получить точку привязки и угол поворота – систему координат макроэлемента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetMacroPlacement.

Синтаксис Automation:

long ksGetMacroPlacement (long macro,

double* x,

double* y,

double* angl);

Входной параметр:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент).

Выходные параметры:

x, y	- координаты точки привязки,
angl	- угол поворота от оси OX против часовой стрелки (в градусах).

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи,
-1	- у макроэлемента нет СК (не вызывалась функция ksDocument2D::ksSetMacroPlacement).

Смотрите также:

Функция ksDocument2D::ksSetMacroPlacement

ksGetMacroPlacementEx– Получить точку привязки и угол поворота – систему координат макроэлемента

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksGetMacroPlacementEx.

Синтаксис Automation:

long ksGetMacroPlacementEx(long macro,
double* x,
double* y,
double* angl,
long sheetParam,
long * mirrorSymmetry);

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
sheetParam	- управление, в какой системе координат получить СК макроэлемента: 1 - в системе координат листа, 0 - в системе координат вида.

Выходные параметры:

x, y	- координаты точки начала координат макроэлемента,
angl	- угол поворота от оси OX против часовой стрелки (в градусах),
mirrorSymmetry	- флаг зеркальной симметрии объекта; 0 - нормальный исходный объект, 1 - макроэлемент получен операцией симметрии из исходного.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи (у макроэлемента нет СК).

Смотрите также:

1. Функция ksDocument2D::ksSetMacroPlacement
2. Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена относительно исходного ее построения...

Примечание:

Метод позволяет получить СК макроэлемента в СК вида или листа, угол поворота, флаг зеркальной симметрии объекта. Если СК макро не имеет, функция вернет 0.

Если macro = 0, функция позволяет получить СК редактируемого макроэлемента.

ksSetMacroParam – Записать параметры макроэлемента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetMacroParam

Синтаксис Automation:

long ksSetMacroParam (long ref,

LPDISPATCH userPars,

bool dblClickOff,

bool hotpoints,

bool externEdit);

Входные параметры:

ref	- указатель на макроэлемент,
userPars	- указатель на интерфейс ksUserParam,

dblClickOff	- признак редактирования по двойному щелчку: TRUE - выключено, FALSE - включено,
hotpoints	- признак наличия интерфейса характерных точек: TRUE - интерфейс характерных точек включен, FALSE - выключен,
externEdit	признак наличия интерфейса внешнего управления: TRUE - интерфейс внешнего управления включен, FALSE - выключен.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Метод записывает в указанный макроэлемент параметры редактирования: имя файла библиотеки, имя библиотеки и номер функции, предназначенной для редактирования данного макроэлемента. Получить значения параметров макроэлемента можно, используя метод ksDocument2D::ksGetMacroParam.
2. При необходимости в параметрах макроэлемента можно дополнительно сохранить пользовательские данные (допустимы все типы данных, кроме указателей), задав ksUserParam::userParams или ksUserParam::SetUserArray.
3. При значениях параметров ksUserParam::fileName = 0, ksUserParam::libName = 0, ksUserParam::number=-1 в макроэлементе сохраняется в качестве редактирующей та функция, которая вызвала метод ksSetMacroParam.
4. При обработке событий это правило не действует и в макро прописываются имя той библиотеки и ее файла, которая была использована непосредственно перед этим событием. Поэтому при создании макроэлемент по событию необходимо явно задавать имя библиотеки, имя файла и номер функции.

ksSetMacroPlacement – Установить точку привязки и угол поворота – систему координат макроэлемента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetMacroPlacement.

Синтаксис Automation:

long ksSetMacroPlacement (long macro,

double x,

double y,

double angl,

long relativ);

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
x, y	- координаты точки привязки,
angl	- угол поворота от оси OX против часовой стрелки (в градусах),
relativ	признак системы координат: 1 - x, y, angl показывают смещение относительно СК редактируемого макроэлемента, 0 - абсолютные значения x, y, angl в СК вида.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также:

5. Методы ksDocument2D::ksGetMacroPlacement,ksDocument2D::ksSetMacroPlacementExРекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена относительно исходного ее построения...

ksSetMacroPlacementEx – Установить точку привязки и угол поворота – систему координат макроэлемента

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksSetMacroPlacementEx.

Синтаксис Automation:

long ksSetMacroPlacementEx(long macro,
double x,
double y,
double angl,
long relativ,
long mirrorSymmetry);

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
x, y	- координаты точки начала координат макроэлемента,
angl	- угол поворота от оси OX против часовой стрелки (в градусах),
relativ	признак системы координат: 1 - x, y, angl показывают смещение относительно СК редактируемого макроэлемента, 0 - абсолютные значения x, y, angl в СК вида,
mirrorSymmetry	- флаг зеркальной симметрии объекта.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Метод позволяет установить СК макроэлемента либо в СК вида, либо смещением относительно СК макро.

Смотрите также**:**

1. Функция ksDocument2D::ksSetMacroPlacement
2. Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена относительно исходного ее построения...

ksSetMacroWaitDblClickEdit – Изменить режим ожидания DblClick при редактировании макроэлемента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetMacroWaitDblClickEdit.

Синтаксис Automation:

long ksSetMacroWaitDblClickEdit (reference ref, long waitDblClick);

Входные параметры:

ref	- указатель на макроэлемент,
waitDblClick	- режим ожидания DblClick (1 - включить, 0 - выключить).

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Макроэлемент, созданный библиотекой, может редактироваться двумя способами:

- по первому клику мыши - через характерные точки и\или интерфейс внешних воздействий;
- по двойному клику - через команду библиотеки.

Если время инициализации характерных точек сравнимо с двойным кликом, то редактирование через команду библиотеки становится невозможно. Во время редактирования макроэлемента, при включенном режиме ожидания DblClick, после первого щелчка по макроэлементу КОМПАС будет ожидать повторного нажатия в течении времени, установленного в настройках "Скорость выполнения двойного щелчка".

При выключенном режиме ожидания DblClick пауза отсутствует, и нажатие всегда интерпретируется как одинарное нажатие.

ksUpdateMacro – Очистить макроэлемент и положить в него группу

Интерфейс...Аналог данного метода при использовании API экспортных функций - ksUpdateMacro.

Синтаксис Automation:

long ksUpdateMacro (long macro, long gr);

Входные параметры:

macro	- указатель на макроэлемент,
gr	- указатель на группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод удаляет всю геометрию макроэлемента.

Стили

DeleteStyleFromDocument – Удалить стиль из текущего документа

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksDeleteStyleFromDocument.

Синтаксис Automation:

long ksDeleteStyleFromDocument (short type,

LPDISPATCH param,

short copy);

Входные параметры:

type	тип стиля,
param	указатель на интерфейс соответствующего типа,
copy	- признак используемой структуры стиля: 0 - создать вручную, 1- взять стиль из библиотеки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

В случае, когда стиль берется из библиотеки, param - указатель на интерфейс ksLibStyle.

ksAddStyle – Добавить в документ новый стиль

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - AddStyle.

Синтаксис Automation:

long ksAddStyle (short type, LPDISPATCH param, short copy);

Входные параметры:

type	тип стиля,
param	указатель на интерфейс соответствующего типа,
copy	- признак создания стиля: 0 - создать вручную, 1- взять стиль из библиотеки.

Типы стилей и интерфейсы...

Возвращаемое значение:

идентификатор стиля	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. В настоящее время метод реализован для стилей кривых, текстов и штриховок.
2. При копировании стиля из библиотеки param - указатель на интерфейс ksLibStyle.

ksGetStyleParam – Получить параметры стиля из документа

Интерфейс...Справка системы КОМПАС: стиль текста...

Справка системы КОМПАС:стили геометрических объектов...

Аналог данного метода при использовании API экспортных функций - GetStyleParam.

Синтаксис Automation:

long ksGetStyleParam (short type, short styleId, LPDISPATCH param);

Входные параметры:

type	- тип стиля,
styleId	- номер стиля,
param	- указатель на интерфейс соответствующего типа.

Типы стилей и интерфейсы...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

В случае неудачи структура param заполняется параметрами стиля по умолчанию.

ksIsStyleInDocument – Проверить, существует ли стиль в текущем документе

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksIsStyleInDocument.

Синтаксис Automation:

long ksIsStyleInDocument (short type,

LPDISPATCH param,

short copy);

Входные параметры:

type	- тип стиля,
param	- указатель на интерфейс соответствующего типа,
copy	- признак используемой структуры стиля: 0 - создать вручную, 1- взять стиль из библиотеки.

Типы стилей и интерфейсы...**Возвращаемое значение**:

1	- стиль в документе есть,
0	- стиля в документе нет.

Примечание:

В случае, когда стиль берется из библиотеки, param - указатель на интерфейс ksLibStyle.

Кривые Безье, NURBS, ломаные

ksAddPowerForm – Ввести параметр для построения NURBS кусочно-степенным способом

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAddPowerForm.

Синтаксис Automation:

long ksAddPowerForm (double x, double y);

Входные параметры:

x, y	- параметры степенной функции (см. Примечание).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Кусочно-степенная форма состоит из последовательности стыкующихся полиномов:

P0 [0, 1), P1 [1, 2), ... Pk [k, k1),

где Pi - полином, [i, i1) - интервал определения,

Pi(t).x = a0.x a1.x * (t - i) a2.x * (t - i) * (t - i) ... an.x * (t - i)^n

Pi(t).y = a0.y a1.y * (t - i) a2.y * (t - i) * (t - i) ... an.y * (t - i)^n

Функция ksAddPowerForm должна вызываться последовательно n1 раз для пар параметров (a0.x, a0.y), (a1.x, a1.y), ... (an.x, an.y).

Затем вызывается функция ksCreatePowerArc, которая создает из переданных параметров дугу NURBS степени n 1 и присоединяет ее к существующей кривой NURBS.

ksBezier – Создать кривую Безье

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Bezier.

Синтаксис Automation:

long ksBezier (short closed, long style);

Входные параметры:

closed	- признак замыкания сплайна: 0 - незамкнутый, 1 - замкнутый,
style	стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Кривая Безье - составной объект. Объекты ksDocument2D::ksBezierPoint, вводимые между методами ksDocument2D::ksBezier и ksDocument2D::ksEndObj, принадлежат кривой. ksDocument2D::ksEndObj возвращает указатель на кривую Безье.

ksBezierPoint – Создать точку для построения кривой Безье

Интерфейс...

Аналог данного метода при использовании API экспортных функций - BezierPoint.

Синтаксис Automation:

long ksBezierPoint (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс кривой Безье ksBezierPointParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksCreatePowerArc – Построить дугу NURBS кусочно–степенным способом и присоединить ее к существующей кривой NURBS

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCreatePowerArc.

Синтаксис Automation:

long ksCreatePowerArc();

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание :

Параметры для построения дуги определяются функцией ksAddPowerForm.

ksNurbs – Создать кривую NURBS

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Nurbs.

Синтаксис Automation:

long ksNurbs (short degree, BOOL close, long style);

Входные параметры:

degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
close	- признак замыкания сплайна: FALSE - незамкнутый, TRUE - замкнутый,
style	стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Узлы кривой определяются следующими далее методами ksDocument2D::ksPoint или ksDocument2D::ksNurbsPoint.

Описание кривой завершается методом ksDocument2D::ksEndObj, возвращающим указатель на созданную кривую.

ksNurbsForConicCurve – Создать кривую NURBS по характеристическим точкам конического сечения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - NurbsForConicCurve.

Синтаксис Automation:

long ksNurbsForConicCurve (LPDISPATCH xArr,

LPDISPATCH yArr,

short style);

Входные параметры:

xArr, yArr	- указатели на интерфейсы динамических массивов ksDynamicArray чисел типа DOUBLE_ARR,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на кривую NURBS	- в случае удачного завершения,
0	- в случае неудачи.

ksNurbsKnot – Создать узел NURBS–кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksNurbsKnot.

Синтаксис Automation:

long ksNurbsKnot (double knot);

Входной параметр:

knot - узел кривой NURBS.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Метод предназначен для считывания форматов обмена, в частности, DXF.

Для создания кривой NURBS массив узлов (узловой вектор) задавать не обязательно. Но если он задан, то должен подчиняться следующим правилам:

1. Узловой вектор не должен быть убывающим.
2. Количество узлов (knotCount) должно быть:

- для разомкнутого сплайна knotCount= degree + pointCount;
- для замкнутого сплайна knotCount= degree + pointCount + (degree -1);

где:

degree - степень сплайна,

pointCount - количество точек.

ksNurbsPoint – Создать узел NURBS

Интерфейс...

Аналог данного метода при использовании API экспортных функций - NurbsPoint.

Синтаксис Automation:

long ksNurbsPoint (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс ksNurbsPointParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Создать очередной узел кривой NURBS. Узлы кривой определяются методами ksDocument2D::ksPoint или ksDocument2D::ksNurbsPoint. Описание кривой завершается методом ksDocument2D::ksEndObj, возвращающим указатель на созданную кривую.

ksPolyline – Создать ломаную линию

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPolyline.

Синтаксис Automation:

long ksPolyline (long style);

Входной параметр:

style - стиль линии.

Системные стили линий...**Возвращаемое значение**:

указатель на ломаную линию - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Ломаная линия - составной объект. Объекты ksDocument2D::ksPoint, вводимые между методами ksDocument2D::ksPolyline и ksDocument2D::ksEndObj, принадлежат ломаной линии. ksDocument2D::ksEndObj возвращает указатель на ломаную линию.

Окна

ksGetZoomScale – Получить масштаб и центр изменения масштаба окна графического документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetZoomScale.

Синтаксис Automation:

long ksGetZoomScale (double* x,

double* y,

double* scale_);

Выходные параметры:

x, y - координаты (в СК вида) точки центра изменения масштаба,
scale_ - коэффициент изменения масштаба изображения.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksZoom – Задать масштаб изображения прямоугольной рамкой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksZoom.

Синтаксис Automation:

long ksZoom (double x1,

double y1,

double x2,

double y2);

Входные параметры:

x1, y1	- координаты (в СК вида) первой точки диагонали рамки,
x2, y2	- координаты (в СК вида) второй точки диагонали рамки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksZoomScale – Задать масштаб изображения по точке и коэффициенту масштабирования

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksZoomScale.

Синтаксис Automation:

long ksZoomScale (double x,

double y,

double scale_);

Входные параметры:

x1, y1	- координаты (в СК вида) точки центра изменения масштаба,
scale_	- коэффициент изменения масштаба изображения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksZoomPrevNextOrAll – Показать документ в предыдущем/последующем масштабе или документ целиком

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksZoomPrevNextOrAll.

Синтаксис Automation:

long ksZoomPrevNextOrAll (short type);

Входной параметр:

type	тип масштабирования: 0 - следующий масштаб, 1 - предыдущий масштаб, 2 - весь документ.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Параметры

ksGetObjParam – Получить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjParam.

Синтаксис Automation:

long ksGetObjParam (long reference,

LPDISPATCH param,

long paramType);

Входной параметр:

reference	- указатель на

Выходные параметры:

param	- указатель на интерфейс параметров,
paramType	- тип параметров объекта или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ).

Типы объектов и интерфейсы...

Типы параметров объектов...

Возвращаемое значение:

тип объекта	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Вызов метода с нулевыми значениями параметров param и parType позволяет получить тип объекта по его reference.

ksSetObjParam – Установить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetObjParam.

Синтаксис Automation:

long ksSetObjParam (long Reference,

LPDISPATCH param,

long paramType);

Входные параметры:

Reference	- указатель на объект,
param	- тип параметров объекта или индекс строки в массиве для объектов TECHNICALDEMAND_OBJ и TEXT_OBJ,
paramType	- тип параметров объекта или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ).

Типы объектов и интерфейсы...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Связи и ограничения

ksDestroyObjConstraint – Удалить параметрическую связь или ограничение, наложенные на указанный объект

Интерфейс...

Справка системы КОМПАС...

493_Glava58_Prosmotr_i_udalenie.htmАналог данного метода при использовании API экспортных функций - ksDestroyObjConstraint.

Синтаксис Automation:

long ksDestroyObjConstraint (long obj, LPDISPATCH par);

Входные параметры:

obj - указатель на объект,
par - указатель на интерфейс параметрических связей и ограничений ksConstraintParam.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetObjConstraints – Получить параметрические связи и ограничения, наложенные на указанный объект

Интерфейс...

Справка системы КОМПАС...

493_Glava58_Prosmotr_i_udalenie.htmАналог данного метода при использовании API экспортных функций - ksGetObjConstraints.

Синтаксис Automation:

LPDISPATCH ksGetObjConstraints (long obj);

Входной параметр:

obj - указатель на объект.

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray ограничений типа CONSTRAINT_ARR.

Смотрите также

ksConstraintParam

ksSetObjConstraint – Установить параметрическую связь или ограничение

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSetObjConstraint

Синтаксис Automation:

long ksSetObjConstraint (long obj, LPDISPATCH par);

Входные параметры:

obj - объект, на который накладывается связь или ограничение,
par - указатель на интерфейс параметрических связей и ограничений ksConstraintParam.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Навигация

ksGetObjGabaritRect – Получить габаритный прямоугольник объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjGabaritRect.

Синтаксис Automation:

long ksGetObjGabaritRect (long p,

LPDISPATCH param);

Входные параметры:

p - указатель на объект,
param - указатель на интерфейс параметров прямоугольника ksRectParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Габаритный прямоугольник возвращается в координатах листа.

ksGetViewObjCount – Получить количество объектов в виде

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetViewObjCount.

Синтаксис Automation:

long ksGetViewObjCount (long p);

Входной параметр:

p - указатель на вид.

Возвращаемое значение:

количество объектов в виде - в случае удачного завершения,
-1 - в случае неудачи.

Примечание:

В случае, если p=0, возвращается число объектов текущего вида или фрагмента.

ksKeepReference – Запомнить указатель на объект

Функция не поддерживается

Интерфейс...

Аналог данного метода при использовании API экспортных функций - KeepReference.

Синтаксис Automation:

long ksKeepReference (long r);

Входной параметр:

r - указатель на графический объект.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksReleaseReference – Освободить указатель объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ReleaseReference.

Синтаксис Automation:

long ksReleaseReference (long p);

Входной параметр:

p - указатель на объект.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Не временный объект не уничтожается. В результате таблица объектов сокращается, а операции, связанные с перебором объектов, ускоряются.

Параметрические переменные

ksGetDocVariableArray – Получить массив параметрических переменных графического документа или вставки фрагмента

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetDocVariableArray.

Синтаксис Automation:

LPDISPATCH ksGetDocVariableArray (long p);

Входной параметр:

p	- указатель на документ или вставку фрагмента.

Возвращаемое значение:

	- указатель на динамический массив ksDynamicArray параметрических переменных типа VARIABLE_ARR.

Смотрите также ksVariable

ksGetDimensionVariableName – Получить имя параметрической переменной, связанной с размером

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetDimensionVariableName.

Синтаксис Automation:

BSTR ksGetDimensionVariableName (long dimObj);

Входной параметр:

dimObj	- указатель на размер.

Возвращаемое значение:

строка с именем переменной	- если размеру поставлена в соответствие переменная,
пустая строка	- если размеру не поставлена в соответствие переменная или в случае неудачи.

ksSetDocVariableArray – Заменить значения и, если нужно, комментарии у параметрических переменных графического документа или вставки фрагмента

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSetDocVariableArray.

Синтаксис Automation:

long ksSetDocVariableArray (long obj,

LPDISPATCH arr,

BOOL setNote);

Входные параметры:

obj	- указатель на документ или вставку фрагмента,
arr	- указатель на динамический массив ksDynamicArray параметрических переменных типа VARIABLE_ARR,
setNote	признак редактирования комментариев: 0 - комментарии не менять, 1 - комментарии менять.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также ksVariable

Работа с документом

ksGetDocumentType – Получить тип документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetDocumentType.

Синтаксис Automation:

long ksGetDocumentType (long doc);

Входные параметры:

doc	- указатель на документ,

Возвращаемое значение:

тип документа	- в случае успешного завершения,
0	- в случае неудачи.

Типы документов

Примечание:

Если указатель doc = 0, возвращается тип активного документа.

ksRebuildDocument

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksRebuildDocument.

Синтаксис Automation:

BOOL ksRebuildDocument();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Позволяет перестроить все ассоциативные виды чертежа. Если в чертеже нет ни одного ассоциативного вида, перестраиваться ничего не будет. После вызова функции ассоциативные виды перерисовываются в соответствии с деталями, изображение которых в них содержится.

Оформление чертежа

ksCloseDocument – Закрыть документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Интерфейс...

Справка системы КОМПАС...

`Glava_13_Otkritie_zakritie_dok.htm`

Аналог данного метода при использовании API экспортных функций - CloseDocument.

Синтаксис Automation:

BOOL ksCloseDocument();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Если документ не сохранили, взводится флаг соответствующей ошибки.

ksCloseTechnicalDemand – Закрыть технические требования

Интерфейс...Справка системы КОМПАС...

CM_VIEW_TDEM_SIMPLE.htm

Аналог данного метода при использовании API экспортных функций - CloseTechnicalDemand.

Синтаксис Automation:

long ksCloseTechnicalDemand();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksCreateDocument – Создать документ (чертеж, фрагмент, текстовый документ, деталь, сборку)

Интерфейс...Справка системы КОМПАС...

Glava_12_Sozdanie_sohranenie_dok.htm

Аналог данного метода при использовании API экспортных функций - CreateDocument.

Синтаксис Automation:

BOOL ksCreateDocument (LPDISPATCH par);

Входной параметр:

par	- указатель на интерфейс параметров документа ksDocumentParam.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Созданный документ становится текущим.

ksCreateSheetView – Создать вид

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - CreateSheetView.

Синтаксис Automation:

long ksCreateSheetView (LPDISPATCH par, long* number);

Входные параметры:

par	- указатель на интерфейс параметров вида ksViewParam,
number	- номер вида.

Возвращаемое значение:

указатель на вид	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Созданный вид становится текущим.

Если вид с указанным номером уже существует, новый вид не создается (ошибка).

Если указать *number = 0, то создается вид с номером по возрастанию.

ksGetDocOptions – Получить настройки графического документа

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetDocOptions.

Синтаксис Automation:

long ksGetDocOptions (long optionsType, LPDISPATCH param);

Входной параметр:

optionsType	- тип настройки.

Выходной параметр:

param	- указатель на интерфейс параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

ksGetReferenceDocumentPart – Получить указатель на объект оформления чертежа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetReferenceDocumentPart.

Синтаксис Automation:

long ksGetReferenceDocumentPart (short t);

Входной параметр:

t - тип объекта оформления чертежа.

Возвращаемое значение:

указатель на объект выбранного типа - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Метод предназначен для использования в листе чертежа.

ksGetReferenceDocumentPartEx – Получить указатель на объект оформления чертежа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetReferenceDocumentPartEx.

Синтаксис Automation:

long ksGetReferenceDocumentPartEx (short t, long sheetNumb);

Входные параметры:

t - тип объекта оформления чертежа,

sheetNumb	- зависит от типа получаемого объекта оформления чертежа: для t=0 sheetNumb - номер листа, начиная с 1, для t=1 sheetNumb не используется, для t=2 sheetNumb не используется, для t=3 sheetNumb не используется, для t=4 sheetNumb - номер спецификации, начиная с 1 (0 - текущая спецификация), для t=5 sheetNumb не используется, для t=6 sheetNumb - номер листа, начиная с 1.

Типы объектов оформления чертежа...

Возвращаемое значение:

указатель на объект выбранного типа	- в случае удачного завершения,
0	- в случае неудачи.

ksGetViewNumber – Получить номер вида по указателю на вид

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetViewNumber.

Синтаксис Automation:

long ksGetViewNumber (long p);

Входной параметр:

p	- указатель на вид.

Возвращаемое значение:

номер вида	- в случае удачного завершения,
-1	- в случае неудачи.

Примечание:

Если p - указатель на вид или объект вида, возвращается номер вида этого объекта.

ksGetViewReference – Получить указатель на вид по номеру вида

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetViewReference

Синтаксис Automation:

long ksGetViewReference (long number);

Входной параметр:

number	- номер вида.

Возвращаемое значение:

указатель на вид	- в случае удачного завершения,
0	- в случае неудачи.

ksGetZona – Получить зону текущего чертежа по заданной точке

Интерфейс...Справка системы КОМПАС: определение зоны источника...

Справка системы КОМПАС:разбиение на зоны...

Аналог данного метода при использовании API экспортных функций - ksGetZona.

Синтаксис Automation:

BSTR ksGetZona (double x,

double y,

long* res);

Входные параметры:

x, y	- координаты точки в текущем документе.

Выходной параметр:

res	результат работы метода: 0 - ошибка (например, точка вне документа или документ - не чертеж), -1 - в текущем документе нет разбиения на зоны; 1 - успешное завершение.

Возвращаемое значение:

- строка с названием зоны.

ksNewViewNumber – Определить номер следующего вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - NewViewNumber.

Синтаксис Automation:

long ksNewViewNumber();

Возвращаемое значение:

номер следующего вида	- в случае успешного завершения,
0	- в случае неудачи.

ksOpenDocument – Открыть документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Интерфейс...Аналог данного метода при использовании API экспортных функций - OpenDocument.

Синтаксис Automation:

BOOL ksOpenDocument (BSTR nameDoc, BOOL regim);

Входные параметры:

nameDoc	- имя документа,
regim	- режим работы с документом: 0 - видимый, 1 - невидимый ("слепой")

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Невидимый режим применяется при пакетном (скрытом) формировании документа.

Открытый документ автоматически становится текущим.

ksOpenTechnicalDemand – Открыть технические требования

Интерфейс...Аналог данного метода при использовании API экспортных функций - OpenTechnicalDemand.

Синтаксис Automation:

long ksOpenTechnicalDemand (LPDISPATCH pGab,

long style);

Входные параметры:

pGab	- указатель на интерфейс динамического массива ksDynamicArray параметров листов технических требований типа RECT_ARR,
style	- стиль текста для технических требований.

Системные стили текстов...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Технические требования - составной объект оформления чертежа. Объекты ksDocument2D::ksTextLine, вводимые между ksOpenTechnicalDemand и ksDocument2D::ksCloseTechnicalDemand, принадлежат техническим требованиям.
2. Технические требования могут состоять из нескольких строк, строки могут состоять из нескольких компонент. Компоненты могут изменять параметры текущего шрифта.

   ksCloseTechnicalDemand возвращает указатель на технические требования.

   Если style = 0, то стиль текста - умолчательный.

ksOpenView – Сделать текущим существующий вид с указанным номером

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - OpenView.

Синтаксис Automation:

long ksOpenView (long number);

Входной параметр:

number	- номер вида.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSaveDocument – Сохранить документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Интерфейс...Аналог данного метода при использовании API экспортных функций - SaveDocument.

Синтаксис Automation:

BOOL ksSaveDocument (BSTR fileName);

Входной параметр:

fileName	- полное имя файла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

ksSaveDocumentEx – Сохранить документ в выбранной версии

Интерфейс...Аналог данного метода при использовании API экспортных функций - SaveDocumentEx.

Синтаксис Automation:

BOOL ksSaveDocumentEx (BSTR fileName, long version);

Входные параметры:

fileName	- полное имя файла,
version	- версия для сохранения:
	-1 - в предыдущую версию,
	0 - в текущую версию,
	1 - в версию 5.11.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Если fileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.

ksSetDocOptions – Заменить тип настройки графического документа

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - SetDocOptions.

Синтаксис Automation:

long ksSetDocOptions (long optionsType, LPDISPATCH param);

Входной параметр:

optionsType	- тип настройки.

Выходной параметр:

param	- указатель на интерфейс параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

ksSpecRough – Проставить знак неуказанной шероховатости в чертеже

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSpecRough.

Синтаксис Automation:

long ksSpecRough (LPDISPATCH par);

Входной параметр:

par	- указатель на интерфейс знака неуказанной шероховатости ksSpecRoughParam.

Возвращаемое значение:

указатель на знак неуказанной шероховатости	- в случае удачного завершения,
0	- в случае неудачи.

Группы объектов

ksAddObjGroup – Добавить объект в группу

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - AddObjGroup.

Синтаксис Automation:

long ksAddObjGroup (long group, long obj);

Входные параметры:

group	- указатель на группу,
obj	- указатель на добавляемый объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то добавление производится в группу выделения (т.е. происходит выделение объекта).

ksClearGroup – Очистить группу объектов

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksClearGroup.

Синтаксис Automation:

long ksClearGroup (long group, BOOL deleteTmp);

Входные параметры:

group	- указатель на очищаемую группу (0 - очищается группа селектирования),
deleteTmp	- признак удаления временных элементов: TRUE - временные элементы удаляются, FALSE - временные элементы сохраняются.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksEndGroup – Завершить создание группы

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - EndGroup.

Синтаксис Automation:

long ksEndGroup();

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Объекты вида, виды, слои, если они вводились между методами ksDocument2D::ksNewGroup и ksDocument2D::ksEndGroup, попадают в группу.

Если группа временная, объекты считаются временными.

ksExcludeObjGroup – Исключить объект из группы

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ExcludeObjGroup.

Синтаксис Automation:

long ksExcludeObjGroup (long group, long obj);

Входные параметры:

group	- указатель на группу,
obj	- указатель на исключаемый объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то исключение производится из группы выделения (т.е. снимается выделение объекта чертежа). Если объект создавался в режиме формирования временной группы, то при исключении он автоматически удаляется.

ksExistGroupObj – Проверить группу на наличие объектов

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ExistGroupObj.

Синтаксис Automation:

long ksExistGroupObj (long group);

Входной параметр:

group	- указатель на группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель группы равен нулю, то проверяется группа выделения (есть ли выделенные объекты документа).

ksGetGroup – Получить указатель на группу по ее имени

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - GetGroup.

Синтаксис Automation:

long ksGetGroup (BSTR name);

Входной параметр:

name	- имя группы.

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

ksGetGroupName – Получить имя группы по указателю на группу

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetGroupName**.**

Синтаксис Automation:

BSTR ksGetGroupName (long gr, long* group, long reserve);

Входные параметры:

gr	- указатель на группу,
reserve	- параметр, зарезервированный для дальнейшего использования (в настоящее время его значение не учитывается).

Выходной параметр:

group	- тип указанной группы: 0 - ошибка указания группы, 1 - именная группа (имя есть), 2 - рабочая группа (имени нет),

Возвращаемое значение:

- строка с именем группы.

ksNewGroup – Создать новую группу объектов

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - NewGroup.

Синтаксис Automation:

long ksNewGroup (short type);

Входной параметр:

type	- тип группы: 0 - модельная, 1 - временная.

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. В модельной группе лежат объекты, которые уже созданы в документе. Во временной группе могут лежать временные и существующие объекты.

   Если для временной группы не будет вызван метод ksDocument2D::ksStoreTmpGroup, то временные объекты группы будут уничтожены по окончании работы библиотеки.

2. Создаваемые в дальнейшем до вызова метода ksDocument2D::ksEndGroup объекты чертежа записываются в модель или во временный список объектов (например, для фантомной прорисовки при вводе).
3. Для дальнейшей обработки группы используются те же методы редактирования, что и для отдельных объектов, так как указатель на группу ничем не отличается от указателя на отдельный объект.
4. Группа может объединять объекты вида, виды и слои.
5. При создании новой группы необязательно закрывать предыдущую (т.е. поддерживается вложенность групп).

ksSaveGroup – Сохранить группу объектов в документе

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - SaveGroup.

Синтаксис Automation:

long ksSaveGroup (long group,

BSTR name);

Входные параметры:

group	- указатель на группу,
name	- имя сохраняемой группы.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Данный метод действителен только для модельной группы. После его выполнения группа автоматически сохраняется в чертеже при его записи. В противном случае группа действительна только в текущем сеансе работы. Если указатель группы равен нулю, то сохраняется группа селектирования (выделенные объекты документа).

ksSelectGroup – Автоматически сформировать группу объектов

Интерфейс..

Справка системы КОМПАС: группы объектов...

Справка системы КОМПАС:команды выделения объектов рамкой...

Аналог данного метода при использовании API экспортных функций - SelectGroup.

Синтаксис Automation:

long ksSelectGroup (long group,

short selectMode,

double xmin,

double ymin,

double xmax,

double ymax);

Входные параметры:

group	- указатель на группу,
selectMode	- тип выделения: 1 - объекты внутри прямоугольника-"ловушки", 2 - объекты снаружи прямоугольника-"ловушки", 3 - 3 - объекты, целиком или частично попавшие в прямоугольник-"ловушку".
xmin, ymin	- координаты левой нижней вершины прямоугольника-"ловушки",
xmax, ymax	- координаты правой верхней вершины прямоугольника-"ловушки".

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то добавление производится в группу выделения (т.е. происходит выделение объектов документа).

ksStoreTmpGroup – Вставить временную группу в документ (группа "рассыпается")

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - StoreTmpGroup.

Синтаксис Automation:

long ksStoreTmpGroup (long group);

Входной параметр:

group	- указатель на группу.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Временные объекты не записываются в документ и на экране не отображаются. Они сохраняются только до конца работы создающей их библиотечной команды. Временная группа чаще всего служит для отрисовки фантомного изображения и используется в качестве параметров в методах ksDocument2D и ksDocument2D::ksCursor. Метод ksDocument2D::ksStoreTmpGroup записывает все объекты временной группы в документ (т.е. фиксирует их). Сама же группа получает статус постоянной.

ksViewGetObjectArea – Получить группу графических объектов, определяющих область выделения, используя визуальный процесс

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksViewGetObjectArea.

Синтаксис Automation:

long ksViewGetObjectArea();

Возвращаемое значение:

указатель на временную группу - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Так как система может работать только с одним процессом, то нужно завершить другие процессные функции: ksDocument2D::ksCursor, ksDocument2D::ksPlacement, ksDocument2D::ksCommandWindow, ksDocument2D::ksEditViewObject.

Операции редактирования

ksApproximationCurve – Аппроксимировать кривую дугами и отрезками с определенной точностью

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksApproximationCurve.

Синтаксис Automation:

long ksApproximationCurve (long p,

double eps,

BOOL curentLayer,

double maxRad,

BOOL smooth);

Входные параметры:

p	- указатель на аппроксимируемую кривую,
eps	- точность аппроксимации (от 1e-7 до 1),
curentLayer	- тип размещения получившихся объектов на слоях:
	FALSE - на слой кривой,
	TRUE - в текущий слой,
maxRad	- максимально допустимый радиус дуг,
	(0 - ограничение не накладывается),
smooth	- признак сопряжения:
	1 - гладкое,
	0 - не гладкое.

Возвращаемое значение:

указатель на контур, дугу, отрезок	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Аппроксимируются кривые Безье, NURBS, эквидистанты, эллипсы, дуги эллипсов и контуры. Остальные кривые игнорируются.

ksClearRegion – Очистить указанную область внутри или снаружи границ, заданных группой объектов

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksClearRegion.

Синтаксис Automation:

long ksClearRegion (long grClear,

long grRegion,

BOOL inside);

Входные параметры:

grClear	- группа геометрии, которую нужно очистить
	(0 - просматривать все объекты текущего вида),
grRegion	- группа объектов, задающая область очистки,
inside	- признак расположения удаляемых объектов:
	0 - снаружи области, ограниченной grRegion,
	1 - внутри области.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksCopyObj – Скопировать объект

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCopyObj.

Синтаксис Automation:

long ksCopyObj (long ref,

double xOld,

double yOld,

double xNew,

double yNew,

double scale_,

double angle);

Входные параметры:

ref	- указатель на объект,
xOld, yOld	- координаты базовой точки объекта,
xNew, yNew	- координаты нового положения базовой точки,
scale	- масштаб,
angle	- угол поворота в градусах.

Возвращаемое значение:

указатель на получившийся объект или группу объектов	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если указатель на объект равен нулю, то копируется группа селектирования (выделенные объекты документа).
2. При копировании одиночного объекта новый объект создается на текущем слое текущего вида.
3. При копировании группы объектов слой сохраняется.
4. При копировании одиночного многослойного макрообъекта он перестает быть многослойным. Макро и входящие в него объекты переносятся на текущий слой. Для копирования многослойного макро с сохранением многослойности требуется добавить его в группу.

ksDecomposeObj – Разбить объект на составляющие части – отрезки, дуги, тексты

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - DecomposeObj.

Синтаксис Automation:

long ksDecomposeObj (long p,

short level,

double arrow,

BOOL type);

Входные параметры:

p	- указатель на разбиваемый объект,
level	- степень детализации разбиения: 0 - отрезки, дуги, тексты, точки, 1 - отрезки, тексты, точки, 2 - отрезки, дуги, тексты, 4 - отрезки, дуги, точки, тексты, 5 - отрезки, дуги, тексты, заливки стрелок и треугольников баз, 6 - разбиение объектов из ассоциативного чертежа на составляющие с учетом видимых и невидимых участков,
arrow	- размер "стрелки прогиба",
type	- признак выбранной системы координат: 0 - разбиение объекта в СК вида, 1 - разбиение объекта в СК листа.

Возвращаемое значение:

указатель на временную группу компонент сложного объекта	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Метод используется при разработке различных конверторов, преобразующих информацию из системы КОМПАС во внешние форматы.
2. Текущий документ должен быть графическим.
3. Графический документ разбивается по частям. Такими частями могут быть объекты вида, основная надпись, технические требования, спецификация на листе, знак неуказанной шероховатости.

4. Спецификация разбивается по листам (p - указатель на документ-спецификацию, type - номер листа спецификации, начиная с 1).
5. Сложные кривые заменяются набором отрезков и дуг (при level=1 - только набором отрезков).
6. Точность приближения к исходному объекту задается значением параметра arrow - максимальным расстоянием между исходным объектом и аппроксимирующим отрезком.
7. Если level=2, точки превращаются в графические объекты, служащие для отрисовки этих точек в КОМПАС-ГРАФИК (например, в два отрезка для точки типа "крест"). В остальных случаях точки (в том числе отрисованные в виде "крестов", "треугольников" и т.д.) превращаются в объект типа "точка".
8. Во всех случаях, кроме level=4, сложные тексты (например, тексты, написанные буквами разного начертания - прямого и курсивного) разбиваются на тексты с одинаковыми признаками. При level=4 тексты не изменяются.
9. Исходный объект после разбиения не изменяется.

   См. также:

   Метод ksDocument2D::ksConvertTextToCurve - Преобразовать указанный текст в кривые.

ksDeleteObj – Удалить объект

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteObj.

Синтаксис Automation:

long ksDeleteObj (long ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель ref равен нулю, то удаляются выделенные объекты документа.

ksEnableUndo – Включить/отключить отмену предыдущих операций

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEnableUndo.

Синтаксис Automation:

BOOL ksEnableUndo (BOOL enable);

Входные параметры:

enable	- признак отмены операций: FALSE - отключить, TRUE - включить.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Если enable=FALSE, количество шагов отмены операций устанавливается равным нулю.
2. Если enable=TRUE, восстанавливается старое значение количества шагов отмены операций, измененное вызовом этого метода с параметром enable=FALSE.
3. Если метод ранее не вызывался, количество шагов отмены операций будет равно умолчательному значению.

ksMoveObj – Сдвинуть объект

Интерфейс...

Аналог данного метода при использовании API экспортных функций - MoveObj.

Синтаксис Automation:

long ksMoveObj (long ref,

double dx,

double dy);

Входные параметры:

ref	- указатель на объект,
dx, dy	- вектор смещения объекта.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель ref равен нулю, то сдвигаются выделенные объекты документа.

ksReadGroupFromClip – Прочитать графические объекты из буфера обмена и разместить их во временной группе

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksReadGroupFromClip.

Синтаксис Automation:

long ksReadGroupFromClip();

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Графические объекты должны принадлежать одному виду. Параметрические связи и ограничения объектов при чтении из буфера теряются, атрибуты сохраняются.

ksRotateObj – Повернуть объект

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - RotateObj.

Синтаксис Automation:

long ksRotateObj (long ref,

double x,

double y,

double angle);

Входные параметры:

ref	- указатель на объект,
x, y	- координаты центра поворота,
angle	- угол поворота (в градусах).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель ref равен нулю, то поворачиваются выделенные объекты документа.

ksSymmetryObj – Отразить объект относительно оси

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSymmetryObj.

Синтаксис Automation:

long ksSymmetryObj (long ref,

double x1,

double y1,

double x2,

double y2,

BSTR copy);

Входные параметры:

ref	- указатель на объект,
x1, y1	- координаты первой точки на оси,
x2, y2	- координаты второй точки на оси,
copy	- режим копирования:
	0 - исходные объекты удаляются,
	1 - исходные объекты оставляются.

Возвращаемое значение:

указатель на получившийся объект или группу объектов	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если указатель ref равен нулю, то зеркально отображаются выделенные объекты документа.

ksTransformObj – Преобразовать объект по установленной матрице

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TransformObj.

Синтаксис Automation:

long ksTransformObj (long ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Для работы с матрицами служат методы ksDocument2D::ksMtr и ksDocument2D::ksDeleteMtr.

ksTrimNurbs – Усечь NURBS

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTrimNurbs.

Синтаксис Automation:

long ksTrimNurbs (long pObj,

double tMin,

double tMax);

Входные параметры:

pObj	- указатель на кривую NURBS или 0 для создаваемой кривой NURBS,
tMin, tMax	- границы интервала для усечения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksTrimmCurve – Усечь кривую, оставив часть между указанными точками

Интерфейс...

Справка системы КОМПАС...

CM_CURVEOPER_CUT.htmАналог данного метода при использовании API экспортных функций - ksTrimCurve.

Синтаксис Automation:

long ksTrimmCurve (long curve,

double x1, double y1,

double x2, double y2,

double x3, double y3,

bool deleteOldCurve);

Входные параметры:

curve	- указатель на усекаемую кривую,
x1, y1	- координаты начала оставляемого участка,
x2, y2	- координаты конца оставляемого участка,

x3, y3	- координаты точки, определяющей направление усечения для замкнутых кривых (эта точка принадлежит оставляемому участку),
deleteOldCurve	- признак удаления усекаемой кривой: TRUE - кривая будет удалена после усечения, FALSE - кривая не будет удалена после усечения.

Возвращаемое значение:

указатель на усеченную кривую	- в случае успеха,
0	- в случае неудачи.

Примечание:

Если указанные точки не лежат на кривой, то усечение производится по их проекциям на кривую.

ksWriteGroupToClip – Разместить группу в буфере обмена с удалением или оставлением геометрии в документе-источнике (скопировать или вырезать геометрию в буфер обмена)

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksWriteGroupToClip.

Синтаксис Automation:

long ksWriteGroupToClip (long group,

BOOL copy);

Входные параметры:

group	- указатель на группу,
copy	- признак копирования или вырезания: 1 - копирование, 0 - вырезание.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Редактирование графических объектов

ksEditViewObject – Запустить визуальный процесс редактирования объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEditViewObject.

Синтаксис Automation:

int ksEditViewObject (long ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- объект отредактирован,
0	- объект не отредактирован.

Примечание :

Так как система может работать только с одним процессом, требуется завершить другие процессные функции: ksDocument2D::ksCursor ksDocument2D::ksPlacement, ksDocument2D::ksCommandWindow, ksDocument2D::ksCreateViewObject.

ksExistObj – Проверить существование объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ExistObj.

Синтаксис Automation:

long ksExistObj(long ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- в случае наличия объекта,
0	- в случае отсутствия объекта.

ksLightObj – Выделить объект цветом ("подсветить" объект)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - LightObj.

Синтаксис Automation:

long ksLightObj (long ref,

short ligth);

Входные параметры:

ref	- указатель на объект,
ligth	- режим подсветки:
	1 - установить выделение,
	0 - снять выделение.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksUndoContainer – Включить/отключить объединение операций для Undo

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksUndoContainer.

Синтаксис Automation:

BOOL ksUndoContainer(BOOL add);

Синтаксис COM:

HRESULT ksUndoContainer(BOOL add, BOOL * result);

Входные параметры:

add	- false - закрывает текущий контейнер Undo для объединения операций, true - создает текущий контейнер Undo для объединения операций.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод позволяет включить/отключить объединение операций для Undo.

Создание видов

ksCreateSheetArbitraryView – Создать произвольный ассоциативный вид

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCreateSheetArbitraryView.

Синтаксис Automation:

long ksCreateSheetArbitraryView (LPDISPATCH par, long * number);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
number	- номер создаваемого вида.

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Если значение параметра number = 0, то создается вид с номером по возрастанию.
3. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.

ksCreateSheetArrowView – Создать ассоциативный вид по стрелке

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCreateSheetArrowView.

Синтаксис Automation:

long ksCreateSheetArrowView (LPDISPATCH par, long * number, long obj);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
number	- номер создаваемого вида,
obj	- указатель на объект "стрелка вида".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "стрелка вида" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.

ksCreateSheetProjectionView – Создать проекционный ассоциативный вид

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCreateSheetStandartViews.

Синтаксис Automation:

long ksCreateSheetProjectionView (LPDISPATCH par, long * number, long view);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
number	- номер создаваемого вида,
view	- указатель на базовый вид.

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Базовый вид должен быть ассоциативным.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.

ksCreateSheetRemoteView – Создать ассоциативный вид выносного элемента

Интерфейс...

Справка системы КОМПАС...

CM_REMOTE_ELEMENT.htm

Аналог данного метода при использовании API экспортных функций - ksCreateSheetRemoteView.

Синтаксис Automation:

long ksCreateSheetRemoteView (LPDISPATCH par, long * number, long obj);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
number	- номер создаваемого вида,
obj	- указатель на объект "Обозначение выносного элемента".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "Обозначение выносного элемента" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.

ksCreateSheetSectionView – Создать ассоциативный вид разреза/сечения

Интерфейс...

Справка системы КОМПАС...

CM_CREATE_SECTION_VIEW.htm

Аналог данного метода при использовании API экспортных функций - ksCreateSheetSectionView.

Синтаксис Automation:

long ksCreateSheetSectionView (LPDISPATCH par, long * number, long obj);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
number	- номер создаваемого вида,
obj	- указатель на объект "линия разреза".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "линия разреза" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.

ksCreateSheetStandartViews – Создать стандартные ассоциативные виды

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCreateSheetStandartViews.

Синтаксис Automation:

BOOL ksCreateSheetStandartViews (LPDISPATCH par, long bitVector, double dx, double dy);

Входные параметры:

par	- указатель на интерфейс ksAssociationViewParam параметров ассоциативного вида,
bitVector	- набор типов, которые нужно создать,

dx	- расстояние между видами по горизонтали,
dy	- расстояние между видами по вертикали.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Последний из созданных видов становится текущим.

Работа со слоями

ksChangeObjectLayer – Изменить слой одного объекта

Интерфейс...

Справка системы КОМПАС...

CM_SET_OBJECTS_LAYER_3D.htm

Аналог данного метода при использовании API экспортных функций - ChangeObjectLayer.

Синтаксис Automation:

long ksChangeObjectLayer (long Reference,

long number);

Входные параметры:

Reference	- указатель на объект,
number	- номер слоя, на который переносится объект.

Возвращаемое значение:

-1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Слой, на который переносится объект, должен существовать и быть доступным для редактирования (не фоновым и не выключенным).

ksGetLayerNumber – Получить номер слоя

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetLayerNumber.

Синтаксис Automation:

long ksGetLayerNumber (long Reference);

Входной параметр:

Reference	- указатель на слой.

Возвращаемое значение:

номер слоя	- в случае удачного завершения,
-1	- в случае неудачи.

Примечания:

1. Системный слой имеет номер 0.
2. Если Reference - объект слоя, возвращается номер слоя этого объекта.

ksGetLayerReference – Получить указатель на слой по номеру слоя текущего вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetLayerReference.

Синтаксис Automation:

long ksGetLayerReference (long number);

Входной параметр:

number	- номер слоя.

Возвращаемое значение:

указатель на слой	- в случае удачного завершения,
0	- в случае неудачи.

ksLayer – Сделать слой текущим

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Layer.

Синтаксис Automation:

long ksLayer (long number);

Входной параметр:

number	- номер слоя.

Возвращаемое значение:

указатель на слой	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если слоя с заданным номером нет, он создается.

Текстовые надписи

ksConvertTextToCurve – Преобразовать указанный текст в кривые

Интерфейс...

Справка системы КОМПАС...

CM_CONVERT_TO_NURBS.htm

Аналог данного метода при использовании API экспортных функций - ksConvertTextToCurve.

Синтаксис Automation:

long ksConvertTextToCurve (long text);

Входной параметр:

text	- указатель на текст.

Возвращаемое значение:

указатель на временную группу кривых	- в случае удачного завершения,
0	- в случае неудачи.

ksGetTextAlign – Получить тип привязки текста

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetTextAlign.

Синтаксис Automation:

long ksGetTextAlign (long pText);

Входной параметр:

pText	- указатель на объект "текст".

Возвращаемое значение:

тип привязки текста	- в случае успешного завершения,
-1	- в случае неудачи.

Типы привязки текста...

ksGetTextLength – Получить длину текста в миллиметрах

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetTextLength.

Синтаксис Automation:

double ksGetTextLength (BSTR text,

long style);

Входные параметры:

text	- указатель на строку текста,
style	- стиль текста.

Системные стили текстов...

Возвращаемое значение:

- вычисленное значение длины текста (в миллиметрах).

Примечание:

Функция может принимать строку в синтаксисе версии КОМПАС 4 (спецзнаки, дроби, отклонения).

ksGetTextLengthFromReference – Получить длину текста в миллиметрах

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetTextLengthFromReference.

Синтаксис Automation:

double ksGetTextLengthFromReference (long pText);

Входной параметр:

pText	- указатель на текст.

Возвращаемое значение:

- вычисленное значение длины текста (в миллиметрах).

Примечание:

Функция может принимать строку в синтаксисе версии КОМПАС 4 (спецзнаки, дроби, отклонения).

ksParagraph – Начать параграф

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Paragraph.

Синтаксис Automation:

long ksParagraph (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс ksParagraphParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Параграф - составной объект.
2. Объекты ksDocument2D::ksTextLine, вводимые между методами ksDocument2D::ksParagraph и ksDocument2D::ksEndObj, принадлежат параграфу.
3. Параграф может состоять из нескольких строк, строки могут состоять из нескольких компонент. Компоненты могут изменять параметры текущего шрифта.
4. ksDocument2D::ksEndObj возвращает указатель на параграф.

ksReadTableFromFile – Создать таблицу, используя информацию, хранящуюся в файле *.tbl

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksReadTableFromFile.

Синтаксис Automation:

long ksReadTableFromFile (BSTR tblFileName);

Входной параметр:

tblFileName	- полное имя файла таблицы.

Возвращаемое значение:

указатель на таблицу	- в случае удачного завершения,
0	- в случае неудачи.

ksSetTextAlign – Установить тип привязки текста

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetTextAlign.

Синтаксис Automation:

long ksSetTextAlign (long pText, long align);

Входные параметры:

pText	- указатель на объект "текст",
align	- тип привязки текста.

Типы привязки текста...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSetTextLineAlign – Установить выравнивание текста

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetTextLineAlign.

Синтаксис Automation:

int ksSetTextLineAlign (short align);

Входной параметр:

align	- признак выравнивания: 0 - по левому краю, 1 - по центру, 2 - по правому краю, 3 - по ширине.

Возвращаемое значение:

предыдущий признак выравнивания	- в случае успешного завершения,
-1	- в случае неудачи.

Примечание.

Функция ksDocument2D::ksSetTextLineAlign должна использоваться внутри блока, т.е. необходимо, чтобы был открытый на редактирование текст. Иначе функция не работает.

ksSetTableColumnText – Установить текст ячейки таблицы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetTableColumnText.

Синтаксис Automation:

long ksSetTableColumnText (long numb, LPDISPATCH param);

Входные параметры:

numb	- номер ячейки,
param	- указатель на интерфейс ksTextParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования таблицы.

ksTable – Создать таблицу в графическом документе

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Table

Синтаксис Automation:

long ksTable();

Возвращаемое значение:

указатель на таблицу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Таблица - составной объект. Объекты, вводимые между операторами ksDocument2D::ksTable и ksDocument2D::ksEndObj, принадлежат таблице. Допустимые объекты - отрезки, тексты. Отрезки должны быть вертикальными и горизонтальными.

ksDocument2D::ksEndObj возвращает указатель на таблицу.

Допустимые стили линии для отрезков таблицы:

1 - основная,

2 - тонкая,

7 - утолщенная,

0 - невидимая.

ksText – Создать текст в графическом документе

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Text.

Синтаксис Automation:

long ksText (double x,

double y,

double ang,

double hStr,

double ksuStr,

long bitVector,

BSTR s);

Входные параметры:

x, y	- координаты точки привязки текста,
ang	- угол наклона текста,
hStr	- высота символов,
ksuStr	- сужение текста,
bitVector	- битовый вектор, задающий признаки начертания текста,
s	- строка символов.

Признаки начертания текста...

Возвращаемое значение:

указатель на текст или 1 при использовании внутри таблицы	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Строка символов может включать спецсимвол. Например, чтобы задать 10 градусов, необходимо указать значение "10&01". Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
2. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ #, кроме случаев, когда эти символы используются в строке именно как управляющие."
3. Использование управляющих символов:

- отклонение: $ верхнее отклонение; нижнее отклонение $,
- дробь: $d числитель ; знаменатель $,
- спецсимвол: &nn номер спецсимвола 0...99.

4. bitVector формируется с помощью логической операции |. Поддерживаются определения:

- ITALIC_ON (включить наклон),
- BOLD_ON (включить утолщение),
- UNDERLINE_ON (включить подчеркивание). См. ltdefine.h.

ksTextLine – Создать компоненту строки текста или целую строку, состоящую из компонент

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TextLine.

Синтаксис Automation:

long ksTextLine (LPDISPATCH textItem);

Входной параметр:

textItem	- указатель на интерфейс ksTextItemParam для компоненты текста или ksTextLineParam для всей строки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Компонента строки должна иметь хотя бы одно свойство, отличное от текущего шрифта.
2. Строка символов может включать спецсимвол. Например, чтобы задать 10 градусов, необходимо указать значение "10&01". Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
3. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ #, кроме случаев, когда эти символы используются в строке именно как управляющие."
4. Использование управляющих символов:

- отклонение: $ верхнее отклонение ; нижнее отклонение $,
- дробь: $d числитель ; знаменатель $,
- спецсимвол: &nnn номер спецсимвола.

Работа с таблицей и с допуском формы

ksClearTableColumnText – Очистить ячейку таблицы или допуска формы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksClearTableColumnText.

Синтаксис Automation:

long ksClearTableColumnText (long numb);

Входной параметр:

numb - номер ячейки,

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если numb = 0, очищается вся таблица.

Нумерация ячеек начинается с 1.

Функция используется в режиме редактирования таблицы или допуска формы.

ksColumnNumber – задать номер текущей ячейки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ColumnNumber.

Синтаксис Automation:

long ksColumnNumber (long numb);

Входной параметр:

numb - номер ячейки.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Функция используется в режиме редактирования штампа, таблицы, допуска формы, в режиме создания допуска формы.

ksCombineTwoTableItems – Объединить две ячейки таблицы, если они имеют общую границу

Интерфейс...

Справка системы КОМПАС...

584_69_1_3_Obwedinenie_jacheek.htm

Аналог данного метода при использовании API экспортных функций - ksCombineTwoTableItems.

Синтаксис Automation:

long ksCombineTwoTableItems (long index1, long index2);

Входные параметры:

index1	- номер первой ячейки,
index2	- номер второй ячейки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Нумерация ячеек начинается с 1.

Функция используется в режиме редактирования таблицы или допуска формы.

ksDivideTableItem – Разделить ячейку таблицы

Интерфейс...

Справка системы КОМПАС...

584_69_1_4_Razdelenie_jacheek.htm

Аналог данного метода при использовании API экспортных функций - ksDivideTableItem.

Синтаксис Automation:

long ksDivideTableItem (long index,

BOOL vertical,

long style);

Входные параметры:

index	- номер ячейки,
vertical	- направление разделения ячейки: 1 - вертикально, 0 - горизонтально,
style	- стиль линии получившейся границы: 0 - невидимая, 1 - основная, 2 - тонкая, 7 - утолщенная.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Нумерация ячеек начинается с 1.

Функция используется в режиме редактирования таблицы или допуска формы.

ksGetPointOnToleranceTable – Получить координаты точки на таблице допуска формы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetPointOnToleranceTable.

Синтаксис Automation:

long ksGetPointOnToleranceTable (long tolerance,

short entry,

LPDISPATCH point);

Входные параметры:

tolerance - указатель на допуск формы,
entry - положение базовой точки.

Выходной параметр:

point - указатель на интерфейс параметров математической точки ksMathPointParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksGetTableBorderStyle – Получить стиль границы ячейки

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksGetTableBorderStyle.

Синтаксис Automation:

long ksGetTableBorderStyle (long index,

short typeBorder);

Входные параметры:

index	- номер ячейки,
typeBorder	- тип границы: 0 - левая, 1 - правая, 2 - верхняя, 3 - нижняя.

Возвращаемое значение:

- стиль линии границы: 0 - невидимая, 1 - основная, 2 - тонкая, 7 - утолщенная.	- в случае успешного завершения,
-1	- в случае неудачи.

Примечание:

Нумерация ячеек начинается с 1.

Функция используется в режиме редактирования таблицы или допуска формы.

ksGetTableColumnText – Получить текст ячейки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetTableColumnText.

Синтаксис Automation:

long ksGetTableColumnText (long* numb, LPDISPATCH param);

Выходные параметры:

numb	- номер ячейки,
param	- указатель на интерфейс ksTextParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования таблицы.

Нумерация ячеек начинается с единицы.

Если не определен номер ячейки с помощью функции ksColumnNumber, метод начинает работу с первой ячейки.

После выполнения метода происходит смещение на следующую ячейку.

Если param=0, все графы пройдены.

ksGetTableItemsCount – Получить количество ячеек в таблице (для виртуальной сетки)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetTableItemsCount.

Синтаксис Automation:

long ksGetTableItemsCount (short type);

Входной параметр:

type	- признак, какое число требуется получить: 0 - общее число ячеек, 1 - число ячеек в строке, 2 - число ячеек в колонке.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования таблицы или допуска формы.

ksGetToleranceColumnText – Получить текст ячейки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetToleranceColumnText.

Синтаксис Automation:

long ksGetToleranceColumnText (long* numb, LPDISPATCH par);

Выходные параметры:

numb	- номер ячейки,
par	- указатель на интерфейс параметров строки текста ksTextLineParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Функция используется в режиме редактирования допуска формы.

Нумерация ячеек начинается с единицы.

Если не определен номер ячейки с помощью функции ksDocument2D::ksColumnNumber , метод начинает работу с первой ячейки.

После выполнения метода происходит смещение на следующую ячейку.

Если par=NULL, все графы пройдены.

После использования массив par->pTextItem желательно удалить.

ksOpenTable – Открыть таблицу для редактирования

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksOpenTable.

Синтаксис Automation:

long ksOpenTable (long table);

Входной параметр:

table - указатель на таблицу.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Чтобы закрыть редактирование таблицы, нужно вызвать функцию ksDocument2D::ksEndObj.

ksOpenTolerance – Открыть допуск формы для редактирования

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksOpenTolerance.

Синтаксис Automation:

long ksOpenTolerance (long tolerance);

Входной параметр:

tolerance - указатель на допуск формы.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Чтобы закрыть редактирование допуска формы, нужно вызвать функцию ksDocument2D::ksEndObj.

ksRebuildTableVirtualGrid – Перестроить виртуальную сетку таблицы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksRebuildTableVirtualGrid.

Синтаксис Automation:

long ksRebuildTableVirtualGrid();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция используется в режиме редактирования таблицы или допуска формы.
2. Таблица работает с виртуальной сеткой - регулярной таблицей, наложенной на редактируемую таблицу и отображающей все ячейки. Если редактируемая таблица регулярная, то ее виртуальная сетка полностью совпадает с таблицей.
3. Нумерация ячеек начинается с левого верхнего угла и с единицы по строкам.
4. Перестраивать сетку нужно после объединения или разделения ячеек.

ksSetTableBorderStyle – Изменить стиль границы ячейки

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksSetTableBorderStyle.

Синтаксис Automation:

long ksSetTableBorderStyle (long index,

short typeBorder,

long style);

Входные параметры:

index	- номер ячейки,
typeBorder	- тип границы: 0 - левая, 1 - правая, 2 - верхняя, 3 - нижняя,
style	- стиль линии границы: 0 - невидимая, 1 - основная, 2 - тонкая, 7 - утолщенная.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Нумерация ячеек начинается с 1.

Функция используется в режиме редактирования таблицы или допуска формы.

ksSetToleranceColumnText – Установить текст ячейки допуска формы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetToleranceColumnText.

Синтаксис Automation:

long ksSetToleranceColumnText (long numb, LPDISPATCH par);

Входные параметры:

Numb	- номер ячейки,
Par	- указатель на интерфейс параметров строки текста ksTextLineParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования допуска формы.

Размеры и технологические обозначения

ksAngBreakDimension – проставить угловой размер с обрывом

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - AngBreakDimension.

Синтаксис Automation:

long ksAngBreakDimension (LPDISPATCH angPar);

Входной параметр:

angPar	- указатель на интерфейс углового размера с обрывом ksABreakDimParam.

Возвращаемое значение:

указатель на угловой размер с обрывом	- в случае удачного завершения,
0	- в случае неудачи.

ksAngDimension – Проставить угловой размер

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - AngDimension.

Синтаксис Automation:

long ksAngDimension (LPDISPATCH angPar);

Входной параметр:

angPar	- указатель на интерфейс углового размера ksADimParam.

Возвращаемое значение:

указатель на угловой размер	- в случае удачного завершения,
0	- в случае неудачи.

ksAxisLine – Создать объект "Осевая линия"

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksAxisLine.

Синтаксис Automation:

long ksAxisLine (LPDISPATCH param);

Входные параметры:

param	- указатель на интерфейс ksAxisLineParam параметров осевой линии.

Возвращаемое значение:

указатель на объект "Осевая линия"	- в случае успешного завершения,
0	- в случае неудачи.

ksBase – Проставить обозначения базы

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Base.

Синтаксис Automation:

long ksBase (LPDISPATCH par);

Входной параметр:

par	- указатель на интерфейс обозначения базы ksBaseParam.

Возвращаемое значение:

указатель на обозначение базы	- в случае удачного завершения,
0	- в случае неудачи.

ksBrandLeader – Создать линию-выноску для обозначения клеймения

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - BrandLeader.

Синтаксис Automation:

long ksBrandLeader (LPDISPATCH brandLeaderParam);

Входной параметр:

brandLeaderParam - указатель на интерфейс параметров линии-выноски для обозначения клеймения ksBrandLeaderParam.

Возвращаемое значение:

указатель на линию-выноску для обозначения клеймения - в случае удачного завершения,
0 - в случае неудачи.

ksCentreMarker – Создать обозначение центра

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksCentreMarker.

Синтаксис Automation:

long ksCentreMarker (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс обозначения центра ksCentreParam.

Возвращаемое значение:

указатель на обозначение центра - в случае удачного завершения,
0 - в случае неудачи.

ksCutLine – Создать линию разреза/сечения

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - CutLine.

Синтаксис Automation:

long ksCutLine (LPDISPATCH par);

Входной параметр:

par - указатель на интерфейс линии разреза/сечения ksCutLineParam.

Возвращаемое значение:

указатель на линию разреза/сечения	- в случае удачного завершения,
0	- в случае неудачи.

ksDiamDimension – Проставить диаметральный размер

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - DiamDimension.

Синтаксис Automation:

long ksDiamDimension (LPDISPATCH linPar);

Входной параметр:

linPar	- указатель на интерфейс диаметрального размера ksRDimParam.

Возвращаемое значение:

указатель на диаметральный размер	- в случае удачного завершения,
0	- в случае неудачи.

ksLeader – Создать линию-выноску

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Leader.

Синтаксис Automation:

long ksLeader (LPDISPATCH leaderPar);

Входной параметр:

leaderPar	- указатель на интерфейс параметров линии-выноски ksLeaderParam.

Возвращаемое значение:

указатель на линию-выноску	- в случае удачного завершения,
0	- в случае неудачи.

ksLinBreakDimension – Проставить линейный размер с обрывом

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - LinBreakDimension.

Синтаксис Automation:

long ksLinBreakDimension (LPDISPATCH linPar);

Входной параметр:

linPar	- указатель на интерфейс линейного размера с обрывом ksLBreakDimParam.

Возвращаемое значение:

указатель на линейный размер с обрывом	- в случае удачного завершения,
0	- в случае неудачи.

ksLinDimension – Проставить линейный размер

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - LinDimension.

Синтаксис Automation:

long ksLinDimension (LPDISPATCH linPar);

Входной параметр:

linPar	- указатель на интерфейс линейного размера ksLDimParam.

Возвращаемое значение:

указатель на линейный размер	- в случае удачного завершения,
0	- в случае неудачи.

ksMarkerLeader – Создать линию-выноску для обозначения маркировки

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - MarkerLeader.

Синтаксис Automation:

long ksMarkerLeader (LPDISPATCH markerLeaderParam);

Входной параметр:

markerLeaderParam	- указатель на интерфейс параметров линии-выноски для обозначения маркирования ksMarkerLeaderParam.

Возвращаемое значение:

указатель на линию-выноску для обозначения маркирования	- в случае удачного завершения,
0	- в случае неудачи.

ksOrdinatedDimension – Проставить размер высоты

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksOrdinatedDimension.

Синтаксис Automation:

long ksOrdinatedDimension (IDispatch* ordPar);

Входной параметр:

ordPar	- указатель на интерфейс параметров размера высоты ksOrdinatedDimParam.

Возвращаемое значение:

указатель на размер высоты	- в случае удачного завершения,
0	- в случае неудачи.

ksPositionLeader – Создать позиционную линию-выноску

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - PositionLeader.

Синтаксис Automation:

long ksPositionLeader (LPDISPATCH posLeaderParam);

Входной параметр:

posLeaderParam - указатель на интерфейс параметров позиционной линии-выноски ksPosLeaderParam.

Возвращаемое значение:

указатель на позиционную линию-выноску - в случае удачного завершения,
0 - в случае неудачи.

ksRadBreakDimension – Проставить радиальный размер с изломом

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - RadBreakDimension.

Синтаксис Automation:

long ksRadBreakDimension (LPDISPATCH par);

Входной параметр:

par - указатель на интерфейс радиального размера с изломом ksRBreakDimParam.

Возвращаемое значение:

указатель на радиальный размер с изломом - в случае удачного завершения,
0 - в случае неудачи.

ksRadDimension – Проставить радиальный размер

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - RadDimension.

Синтаксис Automation:

long ksRadDimension (LPDISPATCH par);

Входной параметр:

par - указатель на интерфейс радиального размера ksRDimParam.

Возвращаемое значение:

указатель на радиальный размер - в случае удачного завершения,
0 - в случае неудачи.

ksRough – Проставить обозначение шероховатости

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Rough.

Синтаксис Automation:

long ksRough (LPDISPATCH roughPar);

Входной параметр:

roughPar	- указатель на интерфейс обозначения шероховатости ksRoughParam.

Возвращаемое значение:

указатель на обозначение шероховатости	- в случае удачного завершения,
0	- в случае неудачи.

ksTolerance – Проставить допуск формы

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Tolerance.

Синтаксис Automation:

long ksTolerance (LPDISPATCH par);

Входной параметр:

par	- указатель на интерфейс допуска формы ksToleranceParam.

Возвращаемое значение:

указатель на допуск формы	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Операторы ksDocument2D::ksColumnNumber и ksDocument2D::ksTextLine, вводимые между операторами ksDocument2D::ksTolerance и ksDocument2D::ksEndObj, принадлежат допуску формы.

ksColumnNumber определяет номер ячейки, куда помещать текст. Вначале считается, что допуск формы имеет таблицу из 10 колонок и 10 строк. Каждая ячейка таблицы имеет свой номер. Нумерация идет слева направо и сверху вниз.

По завершении создания объекта таблица форматируется, и лишние ячейки удаляются. ksEndObj возвращает указатель на допуск формы.

ksViewPointer – Создать стрелку направления взгляда

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ViewPointer.

Синтаксис Automation:

long ksViewPointer (LPDISPATCH par);

Входной параметр:

par	- указатель на интерфейс стрелки направления взгляда ksViewPointerParam.

Возвращаемое значение:

указатель на стрелку взгляда	- в случае удачного завершения,
0	- в случае неудачи.

Матрицы преобразования

ksDeleteMtr – Удалить матрицу трансформации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteMtr.

Синтаксис Automation:

long ksDeleteMtr();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод отменяет режим преобразования координат, линейных и угловых параметров, введенный методом ksDocument2D::ksMtr.

ksMtr – Создать матрицу преобразования координат

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMtr.

Синтаксис Automation:

long ksMtr (double x,

double y,

double Angle,

double scaleX,

double scaleY);

Входные параметры:

x, y	- координаты начала локальной системы координат,
angle	- угол наклона системы координат в градусах,
scaleX	- масштаб локальной системы координат по оси X,
scaleY	- масштаб локальной системы координат по оси Y.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается вложение матриц трансформации. В результате вложения действует суммарная матрица, полученная произведением накопленных матриц. Объекты вида, вводимые между методами ksDocument2D::ksMtr и ksDocument2D::ksDeleteMtr, подвергаются преобразованию по суммарной матрице.

Графические примитивы

ksArcByAngle – Создать дугу по двум точкам и углу раствора

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ArcByAngle.

Синтаксис Automation:

long ksArcByAngle (double xc,

double yc,

double rad,

double f1,

double f2,

short direction,

long style);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
f1, f2	- начальный и конечный угол дуги в градусах,
direction	- направление отрисовки дуги: 1 - против часовой стрелки, -1 - по часовой стрелке,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

ksArcByPoint – Создать дугу по центру и конечным точкам

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ArcByPoint.

Синтаксис Automation:

long ksArcByPoint (double xc,

double yc,

double rad,

double x1,

double y1,

double x2,

double y2,

short direction,

long style);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
x1, y1	- координаты начальной точки дуги,
x2, y2	- координаты конечной точки дуги,
direction	- направление отрисовки дуги: 1 - против часовой стрелки, -1 - по часовой стрелке,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

ksArcBy3Points – Создать дугу по трем точкам

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ArcBy3Points.

Синтаксис Automation:

long ksArcBy3Points (double x1,

double y1,

double x2,

double y2,

double x3,

double y3,

long style);

Входные параметры:

x1, y1	- координаты начальной точки на дуге,
x2, y2	- координаты средней точки на дуге,
x3, y3	- координаты конечной точки на дуге,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

ksCircle – Создать окружность

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Circle.

Синтаксис Automation:

long ksCircle (double xc,

double yc,

double rad,

long style);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на окружность	- в случае удачного завершения,
0	- в случае неудачи.

ksColouring – Создать фоновую заливку цветом

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksColouring.

Синтаксис Automation:

long ksColouring (long color);

Входной параметр:

color	- цвет заливки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Заливка - составной объект. Объекты вида, вводимые между методами ksDocument2D::ksColouring и ksDocument2D::ksEndObj, принадлежат заливке и образуют ее границу.

ksDocument2D::ksEndObj возвращает указатель на заливку.

ksConicArc – Построить коническое сечение

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksConicArc.

Синтаксис Automation:

long ksConicArc (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс конического сечения ksConicArcParam.

Возвращаемое значение:

указатель на коническое сечение - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Коническим сечением может быть дуга окружности, дуга эллипса или кривая NURBS.

ksEllipse – Создать эллипс

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksEllipse.

Синтаксис Automation:

long ksEllipse (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс ksEllipseParam.

Возвращаемое значение:

указатель на эллипс - в случае удачного завершения,
0 - в случае неудачи.

ksEllipseArc – Создать дугу эллипса

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEllipseArc.

Синтаксис Automation:

long ksEllipseArc (LPDISPATCH param);

Входной параметр:

param - указатель на интерфейс ksEllipseArcParam.

Возвращаемое значение:

указатель на дугу эллипса - в случае удачного завершения,
0 - в случае неудачи.

ksEquidistant – Построить эквидистанту

Интерфейс..

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Equidistant.

Синтаксис Automation:

long ksEquidistant (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс ksEquidistantParam.

Возвращаемое значение:

указатель на эквидистанту	- в случае удачного завершения,
0	- в случае неудачи.

ksHatch – Создать штриховку

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - HatchEx.

Синтаксис Automation:

long ksHatch (long style,

double Angle,

double step,

double width,

double x0,

double y0);

Входные параметры:

style	- стиль штриховки,
angle	- угол штриховки в градусах,
step	- шаг штриховки,
width	- ширина полосы штрихования вдоль границы штриховки,
x0, y0	- координаты начальной точки штриховки.

Системные стили штриховок...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Все следующие далее до вызова метода ksDocument2D::ksEndObj геометрические примитивы определяют границы штриховки (внешние и внутренние). Порядок определения

элементов границы является произвольным. Метод ksDocument2D::ksEndObj возвращает указатель на штриховку.

ksInsertRaster – Вставить растровый объект

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksInsertRaster.

Синтаксис Automation:

long ksInsertRaster (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс растрового объекта ksRasterParam.

Возвращаемое значение:

указатель на растровый объект	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Создается новый объект "Выносной элемент" в текущем виде и в текущем документе.

ksLine – Создать прямую

Интерфейс...

Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Line.

Синтаксис Automation:

long ksLine (double x,

double y,

double Angle);

Входные параметры:

x, y	- координаты точки на прямой,
Angle	- угол наклона прямой относительно оси OX (в градусах).

Возвращаемое значение:

указатель на прямую	- в случае удачного завершения,

0 - в случае неудачи.

ksLineSeg – Создать отрезок

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - LineSeg.

Синтаксис Automation:

long ksLineSeg (double x1,

double y1,

double x2,

double y2,

long style);

Входные параметры:

x1, y1	- координаты первой точки отрезка,
x2, y2	- координаты второй точки отрезка,
style	- стиль линии.

Системные стили линий...**Возвращаемое значение**:

указатель на отрезок	- в случае удачного завершения,
0	- в случае неудачи.

ksParEllipseArc – Создать дугу эллипса

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksParEllipseArc.

Синтаксис Automation:

long ksParEllipseArc (LPDISPATCH param);

Входной параметр:

param	- указатель на интерфейс ksEllipseArcParam1.

Возвращаемое значение:

указатель на дугу эллипса	- в случае удачного завершения,
0	- в случае неудачи.

ksPoint – Создать точку

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - Point.

Синтаксис Automation:

long ksPoint (double x,

double y,

long style);

Входные параметры:

x, y	- координаты точки,
style	- стиль отрисовки точки.

Системные стили отрисовки точек...

Возвращаемое значение:

указатель на точку	- в случае удачного завершения,
0	- в случае неудачи.

ksPointArraw – Создать значок

Интерфейс...

Справка системы КОМПАС: стрелка линии-выноски...

Справка системы КОМПАС:тип значка...

Аналог данного метода при использовании API экспортных функций - PointArraw.

Синтаксис Automation:

long ksPointArraw (double x,
double y,
double ang,
short term);

Входные параметры:

x, y	- координаты точки привязки значка,
ang	- угол отрисовки значка,
term	- тип значка.

Системные типы значков...

Возвращаемое значение:

указатель на значок	- в случае удачного завершения,
0	- в случае неудачи.

ksRectangle – Создать прямоугольник

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksRectangle.

Синтаксис Automation:

long ksRectangle (LPDISPATCH param,

short centre);

Входные параметры:

param	- указатель на интерфейс параметров прямоугольника ksRectangleParam,
centre	- признак построения обозначения центра: 0 - нет осей, 1 - значок осей (маленький "крестик"), 2 - горизонтальная ось, 3 - обе оси.

Возвращаемое значение:

указатель на прямоугольник	- в случае удачного завершения,
0	- в случае неудачи.

ksRegularPolygon – Создать правильный многоугольник

Интерфейс...Справка системы КОМПАС...

Аналог данного метода при использовании API экспортных функций - ksRegularPolygon.

Синтаксис Automation:

long ksRegularPolygon (LPDISPATCH param,

short centre);

Входные параметры:

param	- указатель на интерфейс параметров многоугольника ksRegularPolygonParam,

centre	- признак построения обозначения центра: 0 - нет осей, 1 - значок осей (маленький "крестик"), 2 - горизонтальная ось, 3 - обе оси.

Возвращаемое значение:

указатель на правильный многоугольник	- в случае удачного завершения,
0	- в случае неудачи.

Интерфейсы фантомов

Интерфейс фантома ksPhantom

Интерфейс фантома.

Аналог данных параметров при использовании API экспортных функций - Phantom.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также:KompasObject

ksPhantom – свойства

phantom – Тип фантома

Интерфейс...

Тип данных:short.

Синтаксис Automation:

phantom = iPhantom.phantom	Получить свойство(*)
iPhantom.phantom = phantom	Установить свойство (*)
phantom = iPhantom.GetPhantom()	Получить свойство (**)
iPhantom.SetPhantom(phantom)	Установить свойство (**)

Типы фантомов...

ksPhantom – методы

GetPhantomParam – Получить указатель на интерфейс параметров фантома

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPhantomParam();

Возвращаемое значение:

- указатель на интерфейс параметров фантома ksType1, ksType2, ksType3, ksType5, ksType6.

Примечание

Метод возвращает указатель на интерфейс параметров фантома указанного типа (смотрите свойство phantom).

Типы фантомов...

Init – Инициализировать параметры

Интерфейс...

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Фантом для сдвига группы (Интерфейс ksType1)

Интерфейс параметров фантома для сдвига группы.

Аналог данных параметров при использовании API экспортных функций - Type1.

Примечание:

Указатель на интерфейс можно получить при помощи метода ksPhantom::GetPhantomParam.

Смотрите также: ksPhantom

ksType1– свойства

angle – Угол поворота группы

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iType1.angle	Получить свойство(*)
iType1.angle = angle	Установить свойство (*)
angle = iType1.GetAngle()	Получить свойство (**)
iType1.SetAngle (angle)	Установить свойство (**)

gr – Указатель на группу

Интерфейс...

Тип данных:long.

Синтаксис Automation:

gr = iType1.gr	Получить свойство(*)
iType1.gr = gr	Установить свойство (*)
gr = iType1.GetGr()	Получить свойство (**)
iType1.SetGr (gr)	Установить свойство (**)

scale_ – Масштаб

Интерфейс...

Тип данных:double.

Синтаксис Automation:

scale_ = iType1.scale_	Получить свойство(*)
iType1.scale_ = scale_	Установить свойство (*)
scale_ = iType1.GetScale_()	Получить свойство (**)
iType1.SetScale_(scale)	Установить свойство (**)

xBase, yBase – Координаты базовой точки группы

Интерфейс...Тип данных:double.

Синтаксис Automation:

xBase = iType1.xBase	Получить свойство(*)
iType1.xBase = xBase	Установить свойство (*)
xBase = iType1.GetXBase()	Получить свойство (**)
iType1.SetXBase (xBase)	Установить свойство (**)

ksType1 – методы

Init – Инициализировать параметры

Интерфейс...

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Фантом – отрезок, фантом – окружность (Интерфейс ksType2)

Интерфейс параметров фантома-отрезка или фантома-окружности.

Аналог данных параметров при использовании API экспортных функций - Type2.

Примечание:

Указатель на интерфейс можно получить при помощи метода ksPhantom::GetPhantomParam.

Смотрите также: ksPhantom

ksType2- свойства

xBase, yBase – Координаты базовой точки

Интерфейс...Тип данных:double.

Синтаксис Automation:

xBase = iType2.xBase	Получить свойство(*)
iType2.xBase = xBase	Установить свойство (*)
xBase = iType2.GetXBase()	Получить свойство (**)
iType2.SetXBase (xBase)	Установить свойство (**)

ksType2 – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Фантом прямоугольник, фантом отрезок под углом (Интерфейс ksType3)

Интерфейс параметров фантома-прямоугольника или фантома-отрезка с заданным углом.

Аналог данных параметров при использовании API экспортных функций - Type3.

Примечание:

Указатель на интерфейс можно получить при помощи метода ksPhantom::GetPhantomParam.

Смотрите также: ksPhantom

ksType3 – свойства

angle – Угол поворота

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
angle = iType3.angle                Получить свойство(* )
iType3.angle = angle                Установить свойство (* )
angle = iType3.GetAngle()           Получить свойство (**)
iType3.SetAngle (angle)             Установить свойство (**)
```

xBase, yBase – Координаты базовой точки

Интерфейс...Тип данных:double.

Синтаксис Automation:

```
xBase = iType3.xBase                Получить свойство(* )
iType3.xBase = xBase                Установить свойство (* )
xBase = iType3.GetXBase()           Получить свойство (**)
iType3.SetXBase (xBase)             Установить свойство (**)
```

ksType3 – методы

Init – Инициализировать параметры

Интерфейс...

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Фантом – половина прямоугольника (Интерфейс ksType5)

Интерфейс параметров фантома-половины прямоугольника с заданным углом.

Аналог данных параметров при использовании API экспортных функций - Type5.

Примечание:

Указатель на интерфейс можно получить при помощи метода ksPhantom::GetPhantomParam.

Смотрите также ksPhantom

ksType5 – свойства

angle – Угол поворота

Интерфейс..

Тип данных:double.

Синтаксис Automation:

angle = iType5.angle	Получить свойство(*)
iType5.angle = angle	Установить свойство (*)
angle = iType5.GetAngle()	Получить свойство (**)
iType5.SetAngle (angle)	Установить свойство (**)

horizon – Признак направления подхода к курсору

Интерфейс..

Тип данных:BOOL.

Значения свойства:

TRUE	- подходим к курсору по горизонтали,
FALSE	- подходим к курсору по вертикали.

Синтаксис Automation:

horizon = iType5.horizon	Получить свойство(*)
iType5.horizon = horizon	Установить свойство (*)
horizon = iType5.GetHorizon()	Получить свойство (**)
iType5.SetHorizon (horizon)	Установить свойство (**)

xBase, yBase – Координаты базовой точки

Интерфейс...Тип данных:double.

Синтаксис Automation:

xBase = iType5.xBase	Получить свойство(*)
iType5.xBase = xBase	Установить свойство (*)
xBase = iType5.GetXBase()	Получить свойство (**)
iType5.SetXBase (xBase)	Установить свойство (**)

ksType5 – методы

Init – Инициализировать параметры

Интерфейс..

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Пользовательский фантом (Интерфейсы ksType6)

Интерфейс параметров пользовательского фантома.

Аналог данных параметров при использовании API экспортных функций - Type6.

Примечание:

Этот интерфейс, в отличие от ksType1, не устанавливает параметры поворота и масштабирования группы. Сдвиг группы должен осуществлять сам пользователь.

Указатель на интерфейс можно получить при помощи метода ksPhantom::GetPhantomParam.

Смотрите также: ksPhantom

ksType6 – свойства

gr – Указатель на временную группу объектов, которая отображается в виде фантома

Интерфейс..

Тип данных:long.

Синтаксис Automation:

gr = iType6.gr	Получить свойство(*)
iType6.gr = gr	Установить свойство (*)
gr = iType6.GetGr()	Получить свойство (**)
iType6.SetGr (gr)	Установить свойство (**)

ksType6 – методы

Init – Инициализировать параметры

Интерфейс..

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры фантома.

Интерфейсы видов, слоев, запроса к системе

Вид(Интерфейс ksViewParam)

Справка системы КОМПАС...

KOMPAS.chm::/378_Glava44_Obshchie_svedenija_.htm

Интерфейс параметров вида.

Аналог данных параметров при использовании API экспортных функций - ViewParam.

Примечание:

1. Указатель на интерфейс должен быть получен с помощью метода KompasObject::GetParamStruct с параметром ko_ViewParam.
2. Параметры могут быть получены с помощью метода ksDocument2D::ksGetObjParam с параметром ALLPARAM.

   Смотрите также: KompasObject

ksViewParam – свойства

angle – Угол поворота вида в системе координат листа чертежа

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iViewParam.angle	Получить свойство(*)
iViewParam.angle = angle	Установить свойство (*)
angle = iViewParam.GetAngle()	Получить свойство (**)
iViewParam.SetAngle(angle)	Установить свойство (**)

color – Цвет вида в активном состоянии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

color = iViewParam.color	Получить свойство(*)
iViewParam.color = color	Установить свойство (*)
color = iViewParam.GetColor()	Получить свойство (**)
iViewParam.SetColor(color)	Установить свойство (**)

name – Имя вида

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iViewParam.name	Получить свойство(*)
iViewParam.name = name	Установить свойство (*)
name = iViewParam.GetName()	Получить свойство (**)
iViewParam.SetName(name)	Установить свойство (**)

scale_ – Масштаб вида

Интерфейс...

Тип данных:double.

Синтаксис Automation:

scale_ = iViewParam.scale_	Получить свойство(*)
iViewParam.scale_ = scale_	Установить свойство (*)
scale_ = iViewParam.GetScale_()	Получить свойство (**)
iViewParam.SetScale_(scale_)	Установить свойство (**)

state – Состояние вида

Интерфейс...

Тип данных:short.

Синтаксис Automation:

state = iViewParam.state	Получить свойство(*)
iViewParam.state = state	Установить свойство (*)
state = iViewParam.GetState()	Получить свойство (**)
iViewParam.SetState(state)	Установить свойство (**)

Состояния видов...

x, y – Координаты точки привязки вида

Интерфейс...Тип данных:double.

Синтаксис Automation:

x = iViewParam.x	Получить свойство(*)
iViewParam.x = x	Установить свойство (*)
x = iViewParam.GetX()	Получить свойство (**)
iViewParam.SetX(x)	Установить свойство (**)

ksViewParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры вида.

Слой (Интерфейс ksLayerParam)

Справка системы КОМПАС...

Интерфейс параметров слоя.

Аналог данных параметров при использовании API экспортных функций - LayerParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLayerParam – свойства

color – Цвет слоя в активном состоянии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

color = iLayerParam.color	Получить свойство(*)
iLayerParam.color = color	Установить свойство (*)
color = iLayerParam.GetColor()	Получить свойство (**)
iLayerParam.SetColor(color)	Установить свойство (**)

name – Имя слоя

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iLayerParam.name	Получить свойство(*)

iLayerParam.name = name	Установить свойство (*)
name = iLayerParam.GetName()	Получить свойство (**)
iLayerParam.SetName(name)	Установить свойство (**)

state – Состояние слоя

Интерфейс...

Тип данных:short.

Синтаксис Automation:

state = iLayerParam.state	Получить свойство(*)
iLayerParam.state = state	Установить свойство (*)
state = iLayerParam.GetState()	Получить свойство (**)
iLayerParam.SetState(state)	Установить свойство (**)

Состояния слоев...

ksLayerParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры слоя.

Запрос к системе (Интерфейс ksRequestInfo)

Интерфейс параметров запроса к системе.

Аналог данных параметров при использовании API экспортных функций - RequestInfo.

Примечания:

1. Данный интерфейс позволяет установить параметры процессов:
   ksDocument2D::ksCursor
   ksDocument2D::ksPlacement
   ksDocument2D::ksCommandWindow.
2. Методами данного интерфейса можно получить/изменить:

- строку или идентификатор строки приглашения,
- строку или идентификатор строки заголовка окна,

- строку меню или идентификатор меню состава команд,
- строку с именем стандартного курсора или идентификатор курсора,
- признак динамического запроса,
- идентификатор приложения (dll), в котором размещены ресурсы процесса,
- функцию обратной связи процесса.

3. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.
4. В случае динамического запроса функция обратной связи вызывается по перемещению курсора. В случае статического запроса функция обратной связи вызывается по щелчку мыши.

   Смотрите также:KompasObject

RequestInfo – свойства

commInstance – Идентификатор приложения (dll), в котором размещены ресурсы процесса

Интерфейс...

Тип данных:long.

Синтаксис Automation:

commInstance = iRequestInfo.commInstance	Получить свойство(*)
iRequestInfo.commInstance = commInstance	Установить свойство (*)
commInstance = iRequestInfo.GetCommInstance()	Получить свойство (**)
iRequestInfo.SetCommInstance (commInstance)	Установить свойство (**)

Примечание:

Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64.

Рекомендуется использовать функцию ksRequestInfo::commInstanceEx.

commInstanceEx – Идентификатор приложения (dll), в котором размещены ресурсы процесса

Интерфейс...

Тип данных: VARIANT.

Синтаксис Automation:

commInstance = Object.commInstanceEx	Получить свойство(*)
Object.commInstanceEx = commInstance	Установить свойство (*)
commInstance = Object.GetCommInstanceEx()	Получить свойство (**)
Object.SetCommInstanceEx(commInstance)	Установить свойство (**)

Примечание:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данное свойство вместо ksRequestInfo::commInstance.

commandsString – Строка меню состава команд

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

commandsString = iRequestInfo.commandsString	Получить свойство(*)
iRequestInfo.commandsString = commandsString	Установить свойство (*)
commandsString = iRequestInfo.GetCommandsString()	Получить свойство (**)
iRequestInfo.SetCommandsString (commandsString)	Установить свойство (**)

Примечание:

Это свойство позволяет сформировать одноуровневое меню (без вложенных пунктов).

Восклицательный знак разделяет пункты меню, например, "!Квадрат !Треугольник !Ромб".

cursor – Строка с именем стандартного курсора

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

cursor = iRequestInfo.cursor	Получить свойство(*)
iRequestInfo.cursor = cursor	Установить свойство (*)
cursor = iRequestInfo.GetCursor()	Получить свойство (**)
iRequestInfo.SetCursor (cursor)	Установить свойство (**)

Примечание:

Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (Idefin2d.h):

 OCR_SELECT

 OCR_SNAP

 OCR_DEFAULT

OCR_CATCH

cursorId – Идентификатор курсора из ресурса

Интерфейс...

Тип данных:long.

Синтаксис Automation:

cursorId = iRequestInfo.cursorId	Получить свойство(*)
iRequestInfo.cursorId = cursorId	Установить свойство (*)
cursorId = iRequestInfo.GetCursorId()	Получить свойство (**)
iRequestInfo.SetCursorId (cursorId)	Установить свойство (**)

Примечание:

Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (ldefin2d.h):

OCR_SELECT

OCR_SNAP

OCR_DEFAULT

OCR_CATCH

dynamic – Признак динамического запроса

Интерфейс...

Тип данных:long.

Значения признака:

0	- статический запрос,
1	- динамический запрос.

Синтаксис Automation:

dynamic = iRequestInfo.dynamic	Получить свойство(*)
iRequestInfo.dynamic = dynamic	Установить свойство (*)
dynamic = iRequestInfo.GetDynamic()	Получить свойство (**)
iRequestInfo.SetDynamic (dynamic)	Установить свойство (**)

Примечание:

В случае динамического запроса функция обратной связи вызывается по перемещению курсора. В случае статического запроса функция обратной связи вызывается по щелчку мыши.

menuId – Идентификатор меню состава команд из ресурса

Интерфейс...

Тип данных:long.

Синтаксис Automation:

menuId = iRequestInfo.menuId	Получить свойство(*)
iRequestInfo.menuId = menuId	Установить свойство (*)
menuId = iRequestInfo.GetMenuId()	Получить свойство (**)
iRequestInfo.SetMenuId (menuId)	Установить свойство (**)

prompt – Строка приглашения

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

prompt = iRequestInfo.prompt	Получить свойство(*)
iRequestInfo.prompt = prompt	Установить свойство (*)
prompt = iRequestInfo.GetPrompt()	Получить свойство (**)
iRequestInfo.SetPrompt (prompt)	Установить свойство (**)

promptId – Идентификатор приглашения из ресурса

Интерфейс...

Тип данных:long.

Синтаксис Automation:

promptId = iRequestInfo.promptId	Получить свойство(*)
iRequestInfo.promptId = promptId	Установить свойство (*)
promptId = iRequestInfo.GetPromptId()	Получить свойство (**)
iRequestInfo.SetPromptId (promptId)	Установить свойство (**)

title – Строка заголовка окна

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

title = iRequestInfo.title	Получить свойство(*)
iRequestInfo.title = title	Установить свойство (*)
title = iRequestInfo.GetTitle()	Получить свойство (**)
iRequestInfo.SetTitle (title)	Установить свойство (**)

titleId – Идентификатор строки заголовка окна из ресурса

Интерфейс...

Тип данных:long.

Синтаксис Automation:

titleId = iRequestInfo.titleId	Получить свойство(*)
iRequestInfo.titleId = titleId	Установить свойство (*)
titleId = iRequestInfo.GetTitleId()	Получить свойство (**)
iRequestInfo.SetTitleId (titleId)	Установить свойство (**)

TakeProcessObject – Объект, редактируемый в подпроцессе

Интерфейс...

Тип данных: Указатель на интерфейс IUnknown (в **COM**) или IDispatch (в **Automation**)

Синтаксис Automation:

TakeProcessObject = Object.GetTakeProcessObject( )	Получить свойство (*)
Object.SetTakeProcessObject(Index, TakeProcessObject)	Установить свойство (**)

RequestInfo – методы

GetCallBackC – Получить имя функции обратной связи для процесса Cursor

Интерфейс...

Синтаксис Automation:

BSTR GetCallBackC();

Возвращаемое значение:

- строка с именем функции обратной связи.

GetCallBackCm – Получить имя функции обратной связи для процесса CommandWindow

Интерфейс...

Синтаксис Automation:

BSTR GetCallBackCm();

Возвращаемое значение:

- строка с именем функции обратной связи.

GetCallBackP – Получить имя функции обратной связи для процесса Placement

Интерфейс...

Синтаксис Automation:

BSTR GetCallBackP();

Возвращаемое значение:

- строка с именем функции обратной связи.

GetTakeObjectCallBack – Получить имя (в Automation) или адрес (в COM) функции обратной связи подчиненного процесса

Интерфейс...

Синтаксис Automation:

BSTR GetTakeObjectCallBack();

Возвращаемое значение:

- имя функции.

Синтаксис COM:

USERTAKEOBJECTCALLBACKPROC GetTakeObjectCallBack();

Возвращаемое значение:

- адрес функции.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры запроса к системе.

SetCallBackC – Установить имя функции обратной связи для процесса Cursor

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackC (LPCTSTR methodName,

long hInst,

LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Если разрабатываемое приложение dll, тогда заполняется параметр hInst, a dispatchOCX = NULL.
2. Если разрабатываемое приложение ActiceX DLL, тогда заполняется параметр dispatchOCX, a hInst = 0.
3. Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64. Рекомендуется использовать функцию ksRequestInfo::SetCallBackCEx.

SetCallBackCEx – Установить имя функции обратной связи для процесса Cursor

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackCEx(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечания:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksRequestInfo::SetCallBackC.

SetCallBackCm – Установить имя функции обратной связи для процесса CommandWindow

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackCm (LPCTSTR methodName,

long hInst,

LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Если разрабатываемое приложение dll, тогда заполняется параметр hInst, а dispatchOCX = NULL.
2. Если разрабатываемое приложение ActiceX DLL, тогда заполняется параметр dispatchOCX, а hInst = 0.

3. Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64. Рекомендуется использовать функцию ksRequestInfo::SetCallBackCmEx.

SetCallBackCmEx – Установить имя функции обратной связи для процесса CommandWindow

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackCmEx(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечания:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksRequestInfo::SetCallBackCm.

SetCallBackP – Установить имя функции обратной связи для процесса Placement

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackP (LPCTSTR methodName,

long hInst,

LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,

dispatchOCX - указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Если разрабатываемое приложение dll, тогда заполняется параметр hInst, a dispatchOCX = NULL.
2. Если разрабатываемое приложение ActiceX DLL, тогда заполняется параметр dispatchOCX, a hInst = 0.
3. Функция устарела. Использование данной функции может привести к ошибке в библиотеке, собранной с конфигурацией x64. Рекомендуется использовать функцию ksRequestInfo::SetCallBackPEx.

SetCallBackPEx – Установить имя функции обратной связи для процесса Placement

Интерфейс...

Синтаксис Automation:

BOOL SetCallBackPEx(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName - строка с именем функции обратной связи,
hInst - идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX - указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечания:

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

В библиотеках, использующих автоматизацию, рекомендуется использовать данную функцию вместо ksRequestInfo::SetCallBackP.

SetCursorText – Установить текст курсора

Интерфейс...

Синтаксис Automation:

BOOL SetCursorText(BSTR Text);

Примечание

Функция работает при запущенном процессе.

Текст сразу выдается над курсором.

SetTakeObjectCallBack – Установить функцию обратной связи для подчиненного процесса

Интерфейс...

Синтаксис Automation:

BOOL SetTakeObjectCallBack(LPCTSTR methodName, VARIANT hInst, LPDISPATCH dispatchOCX);

Входные параметры:

methodName	- строка с именем функции обратной связи,
hInst	- идентификатор приложения (dll), в котором реализована CallBack-функция,
dispatchOCX	- указатель на интерфейс IDispatch, в котором реализована CallBack-функция.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Прототип CallBack-функции:

Синтаксис Automation (пример для Visual C):

BOOL WINAPI TakeObjectCallBack(LPDISPATCH _object);

Входной параметр:

_object	- указатель на интерфейс объекта ksEntity для подпроцессов создания объектов, - указатель на интерфейс сопряжения ksMateConstraint для подпроцессов создания сопряжений.

Синтаксис COM:

BOOL SetTakeProcessObject (USERTAKEOBJECTCALLBACKPROC callBack);

Входной параметр:

callBack - адрес функции.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Тип функции обратной связи для подчиненного процесса

typedef BOOL (__stdcall * USERTAKEOBJECTCALLBACKPROC)(LPUNKNOWN);

Прототип CallBack-функции:

Синтаксис COM:

BOOL __stdcall SelectFilterProc (LPUNKNOWN _object);

Входной параметр:

_object - указатель на интерфейс объекта ksEntity для подпроцессов создания объектов,
- указатель на интерфейс сопряжения ksMateConstraint для подпроцессов создания сопряжений.

Примечание:

Функция вызывается при завершении подпроцесса создания\редактирования подчиненного объекта

HINSTANCE библиотеки нужно передать через VARIANT, как VT_I4 в Win32 и VT_I8 в x64.

Для правильного формирования _variant_t HINSTANCE нужно передавать через приведение к (LONG_PTR).

Местоположение - привязка (Интерфейс ksPlacementParam)

Интерфейс параметров местоположения (привязки).

Аналог данных параметров при использовании API экспортных функций - PlacementParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksPlacementParam – свойства

angle – Угол поворота в системе координат вида

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iPlacementParam.angle	Получить свойство(*)
iPlacementParam.angle = angle	Установить свойство (*)
angle = iPlacementParam.GetAngle()	Получить свойство (**)
iPlacementParam.SetAngle(angle)	Установить свойство (**)

scale_ – Масштаб

Интерфейс...

Тип данных:double.

Синтаксис Automation:

scale_ = iPlacementParam.scale_	Получить свойство(*)
iPlacementParam.scale_ = scale_	Установить свойство (*)
scale_ = iPlacementParam.GetScale_()	Получить свойство (**)
iPlacementParam.SetScale_(scale_)	Установить свойство (**)

xBase, yBase – Координаты базовой точки в системе координат вида

Интерфейс...Тип данных:double.

Синтаксис Automation:

xBase = iPlacementParam.xBase	Получить свойство(*)
iPlacementParam.xBase = xBase	Установить свойство (*)
xBase = iPlacementParam.GetXBase()	Получить свойство (**)
iPlacementParam.SetXBase(xBase)	Установить свойство (**)

ksPlacementParam – методы

Init – Инициализировать параметры.

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры местоположения.
2. Свойство scale = 1.

Интерфейсы параметров графических примитивов

Отрезок (Интерфейс ksLineSegParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_LINESEGM.htm

Интерфейс параметров отрезка.

Аналог данных параметров при использовании API экспортных функций - LineSegParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParam Struct.

Смотрите также: KompasObject

ksLineSegParam – свойства

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iRectParam.style	Получить свойство (*)
iRectParam.style = style	Установить свойство (*)
style = iRectParam.GetStyle()	Получить свойство (**)
iRectParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

x1, y1 – Координаты начальной точки отрезка

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x1 = iLineSegParam.x1	Получить свойство (*)
iLineSegParam.x1 = x1	Установить свойство (*)
x1 = iLineSegParam.GetX1()	Получить свойство (**)
iLineSegParam.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты конечной точки отрезка

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x2 = iRectParam.x2	Получить свойство (*)
iRectParam.x2 = x2	Установить свойство (*)
x2 = iRectParam.GetX2()	Получить свойство (**)
iRectParam.SetX2(x2)	Установить свойство (**)

ksLineSegParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрезка.

Дуга окружности по углу раствора (Интерфейс ksArcByAngleParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CIRCLEARC.htm

Интерфейс параметров дуги по центру и двум углам.

Аналог данных параметров при использовании API экспортных функций - ArcParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksArcByAngleParam – свойства

ang1 – Начальный угол дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang1 = iArcByAngleParam.ang1		Получить свойство (*)
iArcByAngleParam.ang1 = ang1		Установить свойство (*)
ang1	=	Получить свойство (**)
iArcByAngleParam.GetAngle1()		

iArcByAngleParam.SetAngle1(ang1) Установить свойство (**)

ang2 – Конечный угол дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang2 = iArcByAngleParam.ang2 Получить свойство (*)
iArcByAngleParam.ang2 = ang2 Установить свойство (*)
ang2 = iArcByAngleParam.GetAngle2() Получить свойство (**)
iArcByAngleParam.SetAngle2(ang2) Установить свойство (**)

dir – Направление построения дуги

Интерфейс...

Тип данных:short.

Значения свойства:

1 - против часовой стрелки,
-1 - по часовой стрелке.

Синтаксис Automation:

dir = iArcByAngleParam.dir Получить свойство (*)
iArcByAngleParam.dir = dir Установить свойство (*)
dir = iArcByAngleParam.GetDirection() Получить свойство (**)
iArcByAngleParam.SetDirection(dir) Установить свойство (**)

rad – Радиус дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

rad = iArcByAngleParam.rad Получить свойство (*)
iArcByAngleParam.rad = rad Установить свойство (*)
rad = iArcByAngleParam.GetRadius() Получить свойство (**)
iArcByAngleParam.SetRadius(rad) Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iArcByAngleParam.style	Получить свойство (*)
iArcByAngleParam.style = style	Установить свойство (*)
style = iArcByAngleParam.GetStyle()	Получить свойство (**)
iArcByAngleParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iArcByAngleParam.xc	Получить свойство (*)
iArcByAngleParam.xc = xc	Установить свойство (*)
xc = iArcByAngleParam.GetXc()	Получить свойство (**)
iArcByAngleParam.SetXc(xc)	Установить свойство (**)

ksArcByAngleParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры дуги.

Дуга окружности по точкам (Интерфейс ksArcByPointParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CIRCLEARC.htm

Интерфейс параметров дуги по центру и двум точкам.

Аналог данных параметров при использовании API экспортных функций - ArcParam1.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksArcByPointParam – свойства

dir – Направление построения дуги

Интерфейс...

Тип данных:short.

Значения свойства:

1	- против часовой стрелки,
-1	- по часовой стрелке.

Синтаксис Automation:

dir = iArcByPointParam.dir	Получить свойство (*)
iArcByPointParam.dir = dir	Установить свойство (*)
dir = iArcByPointParam.GetDirection()	Получить свойство (**)
iArcByPointParam.SetDirection(dir)	Установить свойство (**)

rad – Радиус дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

rad = iArcByPointParam.rad	Получить свойство (*)
iArcByPointParam.rad = rad	Установить свойство (*)
rad = iArcByPointParam.GetRadius()	Получить свойство (**)
iArcByPointParam.SetRadius(rad)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iArcByPointParam.style	Получить свойство (*)
iArcByPointParam.style = style	Установить свойство (*)
style = iArcByPointParam.GetStyle()	Получить свойство (**)
iArcByPointParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
xc = iArcByPointParam.xc                Получить свойство (* )
iArcByPointParam.xc = xc                Установить свойство (* )
xc = iArcByPointParam.GetXc()           Получить свойство (**)
iArcByPointParam.SetXc( xc )            Установить свойство (**)
```

x1, y1 – Координаты начальной точки дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
x1 = iArcByPointParam.x1                Получить свойство (* )
iArcByPointParam.x1 = x1                Установить свойство (* )
x1 = iArcByPointParam.GetX1()           Получить свойство (**)
iArcByPointParam.SetX1( x1 )            Установить свойство (**)
```

x2, y2 – Координаты конечной точки дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
x2 = iArcByPointParam.x2                Получить свойство (* )
iArcByPointParam.x2 = x2                Установить свойство (* )
x2 = iArcByPointParam.GetX2()           Получить свойство (**)
iArcByPointParam.SetX2( x2 )            Установить свойство (**)
```

ksArcByPointParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры дуги.

Математическая точка (Интерфейс ksMathPointParam)

Интерфейс параметров математической точки.

Аналог данных параметров при использовании API экспортных функций - MathPointParam.

Примечание:

Указатель на интерфейс можно получить при помощи методов:

- KompasObject::GetParamStruct,
- ksRectParam::GetpBot,
- ksRectParam::GetpTop,

Смотрите также: KompasObject

ksMathPointParam – свойства

x, y – Координаты точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iMathPointParam.x	Получить свойство (*)
iMathPointParam.x = x	Установить свойство (*)
x = iMathPointParam.GetX()	Получить свойство (**)
iMathPointParam.SetX(x)	Установить свойство (**)

ksMathPointParam – методы

Init – Инициализировать параметры

Интерфейс...

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры точки.

Интерфейсы точек касания и сопряжения

Точки касания (Интерфейс ksTAN)

Справка системы КОМПАС..

KOMPAS.chm::/1396_vspomogatelnye.htm#TANGENT2_LINE

Интерфейс массива координат точек касания (структур TAN).

Аналог данных параметров при использовании API экспортных функций - TAN - структура параметров прямой, касательной к двум кривым.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruc.

Смотрите также: KompasObject

ksTAN – методы

GetX1 – Получить координату x1 первой точки касания по заданному индексу в массиве

Интерфейс...

Синтаксис Automation:

double GetX1 (long index);

Входной параметр:

index - номер индекса в массиве точек (допустимое значение от 0 до 3).

Возвращаемое значение:

- координата x1 первой точки касания.

GetX2 – Получить координату x2 второй точки касания по заданному индексу в массиве

Интерфейс...

Синтаксис Automation:

double GetX2 (long index);

Входной параметр:

index - номер индекса в массиве точек (допустимое значение от 0 до 3).

Возвращаемое значение:

- координата x2 первой точки касания.

GetY1 – Получить координату y1 первой точки касания по заданному индексу в массиве

Интерфейс...

Синтаксис Automation:

double GetY1 (long index);

Входной параметр:

index - номер индекса в массиве точек (допустимое значение от 0 до 3).

Возвращаемое значение:

- координата y1 первой точки касания.

GetY2 – Получить координату y2 второй точки касания по заданному индексу в массиве

Интерфейс...

Синтаксис Automation:

double GetY2 (long index);

Входной параметр:

index - номер индекса в массиве точек (допустимое значение от 0 до 3).

Возвращаемое значение:

- координата y2 первой точки касания.

Точки сопряжения (Интерфейс ksCON)

Справка системы КОМПАС...

KOMPAS.chm::/1432_Okruznosti.htm#okr_kasateln_k_dvum_krivum

Интерфейс массива координат точек сопряжения.

Аналог данных параметров при использовании API экспортных функций - CON - структура параметров сопряжения двух кривых окружностью.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCON – методы

GetCount – Получить количество найденных сопряжений

Интерфейс...

Синтаксис Automation:

long GetCount();

Возвращаемое значение:

- количество найденных сопряжений.

GetXc – Получить координату x центра сопрягающей окружности

Интерфейс...

Синтаксис Automation:

double GetXc (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата x центра сопрягающей окружности.

GetX1 – Получить координату x первой точки сопряжения

Интерфейс...

Синтаксис Automation:

double GetX1 (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата x первой точки сопряжения.

GetX2 – Получить координату x второй точки сопряжения

Интерфейс...

Синтаксис Automation:

double GetX2 (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата x второй точки сопряжения.

GetYc – Получить координату y центра сопрягающей окружности

Интерфейс...

Синтаксис Automation:

double GetYc (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата y центра сопрягающей окружности.

GetY1 – Получить координату y первой точки сопряжения

Интерфейс...

Синтаксис Automation:

double GetY1 (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата у первой точки сопряжения.

GetY2 – Получить координату у второй точки сопряжения

Интерфейс...

Синтаксис Automation:

double GetY2 (long index);

Входной параметр:

index	- номер индекса сопрягающей окружности в массиве (допустимое значение от 0 до 7).

Возвращаемое значение:

- координата у второй точки сопряжения.

Точка (Интерфейс ksPointParam)

Справка системы КОМПАС...

KOMPAS.chm::/1391_Postroenie_tochek.htm#simple_point

Интерфейс параметров точки.

Аналог данных параметров при использовании API экспортных функций - PointParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject.

ksPointParam – свойства

style – Стиль отрисовки точки

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iPointParam.style	Получить свойство (*)
iPointParam.style = style	Установить свойство (*)
style = iPointParam.GetStyle()	Получить свойство (**)
iPointParam.SetStyle(style)	Установить свойство (**)

Системные стили точек..

x, y – Координаты точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iPointParam.x	Получить свойство (*)
iPointParam.x = x	Установить свойство (*)
x = iPointParam.GetX()	Получить свойство (**)
iPointParam.SetX(x)	Установить свойство (**)

ksPointParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры точки.

Вспомогательная прямая (Интерфейс ksLineParam)

Интерфейс параметров вспомогательной прямой.

Аналог данных параметров при использовании API экспортных функций - LineParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLineParam – свойства

angle – Угол наклона прямой относительно оси X

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iLineParam.angle	Получить свойство (*)
iLineParam.angle = angle	Установить свойство (*)
angle = iLineParam.GetAngle()	Получить свойство (**)

iLineParam.SetAngle(angle) Установить свойство (**)

x, y – Координаты точки на прямой

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iLineParam.x Получить свойство (*)
iLineParam.x = x Установить свойство (*)
x = iLineParam.GetX() Получить свойство (**)
iLineParam.SetX(x) Установить свойство (**)

ksLineParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE

Примечание:

Метод обнуляет все параметры вспомогательной линии.

Окружность (Интерфейс ksCircleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1432_Okruznosti.htm#okr_po_centru_i_tochke

Интерфейс параметров окружности.

Аналог данных параметров при использовании API экспортных функций - CircleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также; KompasObject

ksCircleParam – свойства

rad – Радиус окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

rad = iCircleParam.rad	Получить свойство (*)
iCircleParam.rad = rad	Установить свойство (*)
rad = iCircleParam.GetRadius()	Получить свойство (**)
iCircleParam.SetRadius(rad)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iCircleParam.style	Получить свойство (*)
iCircleParam.style = style	Установить свойство (*)
style = iCircleParam.GetStyle()	Получить свойство (**)
iCircleParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iCircleParam.xc	Получить свойство (*)
iCircleParam.xc = xc	Установить свойство (*)
xc = iCircleParam.GetXc()	Получить свойство (**)
iCircleParam.SetXc(xc)	Установить свойство (**)

ksCircleParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрезка.

Эллипс (Интерфейс ksEllipseParam)

Интерфейс параметров эллипса.

Аналог интерфейса при использовании API экспортных функций - структура параметров EllipseParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksEllipseParam – свойства

a, b – Длина полуосей эллипса

Интерфейс..

Тип данных:double.

Синтаксис Automation:

a = iEllipseParam.a	Получить свойство (*)
iEllipseParam.a = a	Установить свойство (*)
a = iEllipseParam.GetA()	Получить свойство (**)
iEllipseParam.SetA(a)	Установить свойство (**)

angle – Угол наклона оси эллипса a к оси X

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iEllipseParam.angle	Получить свойство (*)
iEllipseParam.angle = angle	Установить свойство (*)
angle = iEllipseParam.GetAngle()	Получить свойство (**)
iEllipseParam.SetAngle(angle)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iEllipseParam.style	Получить свойство (*)
iEllipseParam.style = style	Установить свойство (*)
style = iEllipseParam.GetStyle()	Получить свойство (**)
iEllipseParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iEllipseParam.xc	Получить свойство (*)
iEllipseParam.xc = xc	Установить свойство (*)
xc = iEllipseParam.GetXc()	Получить свойство (**)
iEllipseParam.SetXc(xc)	Установить свойство (**)

ksEllipseParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры эллипса.

Дуга эллипса (Интерфейс ksEllipseArcParam)

Интерфейс параметров дуги эллипса.

Аналог данных параметров при использовании API экспортных функций - EllipseArcParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksEllipseArcParam – свойства

a, b – Длины полуосей эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

a = iEllipseArcParam.a	Получить свойство (*)
iEllipseArcParam.a = a	Установить свойство (*)
a = iEllipseArcParam.GetA()	Получить свойство (**)
iEllipseArcParam.SetA(a)	Установить свойство (**)

angle – Угол наклона дуги эллипса (оси a) к оси X

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iEllipseArcParam.angle	Получить свойство (*)
iEllipseArcParam.angle = angle	Установить свойство (*)
angle = iEllipseArcParam.GetAngle( )	Получить свойство (**)
iEllipseArcParam.SetAngle(angle)	Установить свойство (**)

angleFirst – Начальный угол дуги относительно оси a

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angleFirst = iEllipseArcParam.angleFirst	Получить свойство (*)
iEllipseArcParam.angleFirst = angleFirst	Установить свойство (*)
angleFirst = iEllipseArcParam.GetAngleFirst( )	Получить свойство (**)
iEllipseArcParam.SetAngleFirst(angleFirst)	Установить свойство (**)

angleSecond – Конечный угол дуги относительно оси a

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angleSecond = iEllipseArcParam.angleSecond	Получить свойство (*)
iEllipseArcParam.angleSecond = angleSecond	Установить свойство (*)
angleSecond = iEllipseArcParam.GetAngleSecond( )	Получить свойство (**)
iEllipseArcParam.SetAngleSecond(angleSecond)	Установить свойство (**)

direction – Направление построения дуги эллипса

Интерфейс...

Тип данных:short.

Значения свойства:

1	- против часовой стрелки,

-1 - по часовой стрелке.

Синтаксис Automation:

direction = iEllipseArcParam.direction — Получить свойство (*)
iEllipseArcParam.direction = direction — Установить свойство (*)
direction = iEllipseArcParam.GetDirection( ) — Получить свойство (**)
iEllipseArcParam.SetDirection(direction) — Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iEllipseArcParam.style — Получить свойство (*)
iEllipseArcParam.style = style — Установить свойство (*)
style = iEllipseArcParam.GetStyle( ) — Получить свойство (**)
iEllipseArcParam.SetStyle(style) — Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра дуги эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iEllipseArcParam.xc — Получить свойство (*)
iEllipseArcParam.xc = xc — Установить свойство (*)
xc = iEllipseArcParam.GetXc( ) — Получить свойство (**)
iEllipseArcParam.SetXc(xc) — Установить свойство (**)

ksEllipseArcParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init( );

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры дуги эллипса

Дуга эллипса параметрическая (Интерфейс ksEllipseArcParam1)

Интерфейс параметров для параметрического построения дуги эллипса.

Аналог данных параметров при использовании API экспортных функций - EllipseArcParam1.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksEllipseArcParam1 – свойства

a – Полуось a дуги эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

a = iEllipseArcParam1.a	Получить свойство (*)
iEllipseArcParam1.a = a	Установить свойство (*)
a = iEllipseArcParam1.GetA()	Получить свойство (**)
iEllipseArcParam1.SetA(a)	Установить свойство (**)

angle – Угол наклона дуги эллипса (оси a) к оси X

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iEllipseArcParam1.angle	Получить свойство (*)
iEllipseArcParam1.angle = angle	Установить свойство (*)
angle = iEllipseArcParam1.GetAngle()	Получить свойство (**)
iEllipseArcParam1.SetAngle(angle)	Установить свойство (**)

b – Полуось b дуги эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

b = iEllipseArcParam1.b	Получить свойство (*)
iEllipseArcParam1.b = b	Установить свойство (*)
b = iEllipseArcParam1.GetB()	Получить свойство (**)
iEllipseArcParam1.SetB(b)	Установить свойство (**)

direction – Направление построения

Интерфейс...

Тип данных:double.

Значения свойства:

1	- против часовой стрелки,
-1	- по часовой стрелке.

Синтаксис Automation:

direction = iEllipseArcParam1.direction	Получить свойство (*)
iEllipseArcParam1.direction = direction	Установить свойство (*)
direction = iEllipseArcParam1.GetDirection()	Получить свойство (**)
iEllipseArcParam1.SetDirection(direction)	Установить свойство (**)

parFirst – Начальное значение параметра

Интерфейс...

Тип данных:double.

Синтаксис Automation:

parFirst = iEllipseArcParam1.parFirst	Получить свойство (*)
iEllipseArcParam1.parFirst = parFirst	Установить свойство (*)
parFirst = iEllipseArcParam1.GetParFirst()	Получить свойство (**)
iEllipseArcParam1.SetParFirst(parFirst)	Установить свойство (**)

parSecond – Конечное значение параметра

Интерфейс...

Тип данных:double.

Синтаксис Automation:

parSecond = iEllipseArcParam1.parSecond	Получить свойство (*)
iEllipseArcParam1.parSecond = parSecond	Установить свойство (*)
parSecond = iEllipseArcParam1.GetParSecond()	Получить свойство (**)
iEllipseArcParam1.SetParSecond(parSecond)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iEllipseArcParam1.style	Получить свойство (*)
iEllipseArcParam1.style = style	Установить свойство (*)
style = iEllipseArcParam1.GetStyle()	Получить свойство (**)
iEllipseArcParam1.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра дуги эллипса

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iEllipseArcParam1.xc	Получить свойство (*)
iEllipseArcParam1.xc = xc	Установить свойство (*)
xc = iEllipseArcParam1.GetXc()	Получить свойство (**)
iEllipseArcParam1.SetXc(xc)	Установить свойство (**)

ksEllipseArcParam1 – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры для параметрического построения дуги эллипса.

Прямоугольники

Прямоугольник по двум вершинам (Интерфейс ksRectParam)

Справка системы КОМПАС...

KOMPAS.chm::/Prjamougolqnik.htm#rectanguler

Интерфейс параметров прямоугольника по диагональным точкам.

Аналог данных параметров при использовании API экспортных функций - RectParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject.

ksRectParam – методы

GetpBot – Получить параметры левой нижней точки прямоугольника

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpBot();

Возвращаемое значение:

- указатель на интерфейс параметров математической точки ksMathPointParam.

GetpTop – Получить параметры правой верхней точки прямоугольника

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTop();

Возвращаемое значение:

- указатель на интерфейс параметров математической точки ksMathPointParam

SetpBot – Установить параметры левой нижней точки прямоугольника

Интерфейс...

Синтаксис Automation:

BOOL SetpBot(LPDISPATCH pBot);

Входной параметр:

pBot	- указатель на интерфейс параметров математической точки ksMathPointParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetpTop – Установить параметры правой верхней точки прямоугольника

Интерфейс...

Синтаксис Automation:

BOOL SetpTop (LPDISPATCH pTop);

Входной параметр:

pTop	- указатель на интерфейс параметров математической точки ksMathPointParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Прямоугольник по центру (Интерфейс ksRectangleParam)

Справка системы КОМПАС: команда Прямоугольник по центру и вершине

KOMPAS.chm::/Prjamougolqnik.htm#CENTER_VERTEX

Интерфейс параметров прямоугольника.

Аналог данных параметров при использовании API экспортных функций - RectangleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRectangleParam – свойства

ang – Угол вектора направления от первой точки ко второй (угол наклона стороны прямоугольника, выходящей из базовой точки)

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iRectangleParam.ang	Получить свойство (*)

iRectangleParam.ang = ang	Установить свойство (*)
ang = iRectangleParam.GetAng()	Получить свойство (**)
iRectangleParam.SetAng(ang)	Установить свойство (**)

height – Высота прямоугольника

Интерфейс...

Тип данных:double.

Синтаксис Automation:

height = iRectangleParam.height	Получить свойство (*)
iRectangleParam.height = height	Установить свойство (*)
height = iRectangleParam.GetHeight()	Получить свойство (**)
iRectangleParam.SetHeight(height)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iRectangleParam.style	Получить свойство (*)
iRectangleParam.style = style	Установить свойство (*)
style = iRectangleParam.GetStyle()	Получить свойство (**)
iRectangleParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

width – Ширина прямоугольника (длина стороны, характеризующейся углом наклона ang)

Интерфейс...

Тип данных:double.

Синтаксис Automation:

width = iRectangleParam.width	Получить свойство (*)
iRectangleParam.width = width	Установить свойство (*)
width = iRectangleParam.GetWidth()	Получить свойство (**)
iRectangleParam.SetWidth(width)	Установить свойство (**)

x, y – Координаты базовой точки прямоугольника – одной из его вершин

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iRectangleParam.x	Получить свойство (*)
iRectangleParam.x = x	Установить свойство (*)
x = iRectangleParam.GetX()	Получить свойство (**)
iRectangleParam.SetX(x)	Установить свойство (**)

ksRectangleParam – методы

GetPCorner – Получить указатель на интерфейс динамического массива параметров углов прямоугольника ksDynamicArray типа CORNER_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPCorner();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа CORNER_ARR.

Смотрите также: ksCornerParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE

Примечание:

Метод обнуляет все параметры прямоугольника.

SetPCorner – Установить указатель на интерфейс динамического массива параметров углов прямоугольника ksDynamicArray типа CORNER_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPCorner (LPDISPATCH pCorner);

Входной параметр:

pCorner	- указатель на интерфейс динамического массива ksDynamicArray типа CORNER_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Смотрите также: ksCornerParam

Правильный многоугольник (Интерфейс ksRegularPolygonParam

Справка системы КОМПАС...

KOMPAS.chm::/Mnogougolqnik.htm

Интерфейс параметров правильного многоугольника.

Аналог данных параметров при использовании API экспортных функций - RegularPolygonParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

RegularPolygonParam – свойства

ang – Угол радиус-вектора, направленного от центра к первой вершине

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iRegularPolygonParam.ang	Получить свойство (*)
iRegularPolygonParam.ang = ang	Установить свойство (*)
ang = iRegularPolygonParam.GetAng()	Получить свойство (**)
iRegularPolygonParam.SetAng(ang)	Установить свойство (**)

count – Количество вершин многоугольника

Интерфейс...

Тип данных:long.

Синтаксис Automation:

```
count = iRegularPolygonParam.count                  Получить свойство (* )
iRegularPolygonParam.count = count                  Установить свойство (* )
count = iRegularPolygonParam.GetCount()             Получить свойство (**)
iRegularPolygonParam.SetCount( count )              Установить свойство (**)
```

describe – Признак описанного или вписанного многоугольника

Интерфейс...

Тип данных:BOOL.

Значения свойства:

```
FALSE    - вписанный многоугольник,
TRUE     - описанный многоугольник.
```

Синтаксис Automation:

```
describe = iRegularPolygonParam.describe            Получить свойство (* )
iRegularPolygonParam.describe = describe            Установить свойство (* )
describe = iRegularPolygonParam.GetDescribe()       Получить свойство (**)
iRegularPolygonParam.SetDescribe( describe )        Установить свойство (**)
```

radius – Радиус вписанной или описанной окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
radius = iRegularPolygonParam.radius                Получить свойство (* )
iRegularPolygonParam.radius = radius                Установить свойство (* )
radius = iRegularPolygonParam.GetRadius()           Получить свойство (**)
iRegularPolygonParam.SetRadius( radius )            Установить свойство (**)
```

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

```
style = iRegularPolygonParam.style                  Получить свойство (* )
iRegularPolygonParam.style = style                  Установить свойство (* )
```

style = iRegularPolygonParam.GetStyle()	Получить свойство (**)
iRegularPolygonParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

xc, yc – Координаты центра вписанной или описанной окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iRegularPolygonParam.xc	Получить свойство (*)
iRegularPolygonParam.xc = xc	Установить свойство (*)
xc = iRegularPolygonParam.GetXc()	Получить свойство (**)
iRegularPolygonParam.SetXc(xc)	Установить свойство (**)

RegularPolygonParam – методы

GetPCorner – Получить указатель на интерфейс динамического массива параметров углов прямоугольника ksDynamicArray типа CORNER_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPCorner();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа CORNER_ARR.

Смотрите также: ksCornerParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE

Примечания:

1. Метод обнуляет все параметры правильного многоугольника.
2. Динамические массивы не создаются.

SetPCorner – Установить указатель на интерфейс динамического массива параметров углов прямоугольника ksDynamicArray типа CORNER_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPCorner (LPDISPATCH pCorner);

Входной параметр:

pCorner	- указатель на интерфейс динамического массива ksDynamicArray типа CORNER_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Смотрите также: ksCornerParam

Скругление/усечение углов прямоугольников / многоугольников (Интерфейс ksCornerParam)

Справка системы КОМПАС: фаска

`KOMPAS.chm::/172_22_1_Faska.htm`

Справка системы КОМПАС: скругление

`KOMPAS.chm::/173_22_3_Skruglenie.htm`

Интерфейс параметров скругленных (или усеченных) углов для прямоугольников и правильных многоугольников.

Аналог данных параметров при использовании API экспортных функций - CornerParam.

Примечание:

Интерфейс используется для задания скруглений и фасок на определенных углах правильного многоугольника ksRegularPolygonParam и прямоугольника ksRectangleParam.

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCornerParam – свойства

fillet – Признак фаски или скругления

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- скругление,
FALSE	- фаска.

Синтаксис Automation:

fillet = iCornerParam.fillet	Получить свойство (*)
iCornerParam.fillet = fillet	Установить свойство (*)
fillet = iCornerParam.GetFillet()	Получить свойство (**)
iCornerParam.SetFillet(fillet)	Установить освойство (**)

index – Индекс угла (0, 1, 2,...)

Интерфейс...

Тип данных:long.

Синтаксис Automation:

index = iCornerParam.index	Получить свойство (*)
iCornerParam.index = index	Установить свойство (*)
index = iCornerParam.GetIndex()	Получить свойство (**)
iCornerParam.SetIndex(index)	Установить свойство (**)

l1 – Длина фаски первого сегмента или радиус скругления

Интерфейс...

Тип данных:double.

Синтаксис Automation:

l1 = iCornerParam.l1	Получить свойство (*)
iCornerParam.l1 = l1	Установить свойство (*)
l1 = iCornerParam.GetL1()	Получить свойство (**)
iCornerParam.SetL1(l1)	Установить свойство (**)

l2l – Длина фаски второго сегмента

Интерфейс...

Тип данных:double.

Синтаксис Automation:

l2 = iCornerParam.l2	Получить свойство (*)
iCornerParam.l2 = l2	Установить свойство (*)
l2 = iCornerParam.GetL2()	Получить свойство (**)
iCornerParam.SetL2(l2)	Установить свойство (**)

ksCornerParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры углов для прямоугольников и многоугольников.

Ломаная линия (Интерфейс ksPolylineParam)

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#POLYLINE

Интерфейс параметров ломаной линии.

Аналог данных параметров при использовании API экспортных функций - PolylineParamEx.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksPolylineParam – свойства

closed – Признак замкнутости ломаной

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- ломаная замкнута,
FALSE	- ломаная разомкнута.

Синтаксис Automation:

closed = iPolylineParam.closed	Получить свойство (*)
iPolylineParam.closed = closed	Установить свойство (*)
closed = iPolylineParam.GetClosed()	Получить свойство (**)
iPolylineParam.SetClosed(closed)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iPolylineParam.style	Получить свойство (*)
iPolylineParam.style = style	Установить свойство (*)
style = iPolylineParam.GetStyle()	Получить свойство (**)
iPolylineParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

ksPolylineParam – методы

GetpMathPoint – Получить указатель на динамический массив математических точек ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpMathPoint();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POINT_ARR.

Смотрите также: ksMathPointParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры ломаной линии.
2. Создается динамический массив pMathPoint типа POINT_ARR.

SetpMathPoint– Установить указатель на динамический массив математических точек ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpMathPoint (LPDISPATCH pMathPoint);

Входной параметр:

pMathPoint	- указатель на интерфейс динамического массива ksDynamicArray типа POINT_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также: ksMathPointParam

NURBS

Точка NURBS (Интерфейс ksNurbsPointParam)

Интерфейс параметров точки кривой NURBS.

Аналог данных параметров при использовании API экспортных функций - NurbsPointParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksNurbsPointParam – свойства

weight – Вес точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

weight = iNurbsPointParam.weight	Получить свойство (*)
iNurbsPointParam.weight = weight	Установить свойство (*)

weight = iNurbsPointParam.GetWeight() Получить свойство (**)
iNurbsPointParam.SetWeight(weight) Установить свойство (**)

Синтаксис Automation:

Значение свойства должно быть больше нуля.

x, y – Координаты базовой точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iNurbsPointParam.x Получить свойство (*)
iNurbsPointParam.x = x Установить свойство (*)
x = iNurbsPointParam.GetX() Получить свойство (**)
iNurbsPointParam.SetX(x) Установить свойство (**)

ksNurbsPointParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры узла для кривой NURBS.

Кривая NURBS (Интерфейс ksNurbsParam)

Интерфейс параметров кривой NURBS.

Аналог данных параметров при использовании API экспортных функций - NurbsParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksNurbsParam – свойства

close – Признак замыкания сплайна

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE	- незамкнутый,
TRUE	- замкнутый.

Синтаксис Automation:

close = iNurbsParam.close	Получить свойство (*)
iNurbsParam.close = close	Установить свойство (*)
close = iNurbsParam.GetClose()	Получить свойство (**)
iNurbsParam.SetClose(close)	Установить свойство (**)

degree – Порядок NURBS (степень полинома + 1), от 3 до 10

Интерфейс...

Тип данных:short.

Синтаксис Automation:

degree = iNurbsParam.degree	Получить свойство (*)
iNurbsParam.degree = degree	Установить свойство (*)
degree = iNurbsParam.GetDegree()	Получить свойство (**)
iNurbsParam.SetDegree(degree)	Установить свойство (**)

periodic – Признак периодичности сплайна

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE	- непериодический сплайн,
TRUE	- периодический сплайн.

Синтаксис Automation:

periodic = iNurbsParam.periodic	Получить свойство (*)
periodic = iNurbsParam.GetPeriodic()	Получить свойство (**)

Примечания:

1. Свойство используется только для функции ksGetObjParam.
2. Свойство используется только для чтения.

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iNurbsParam.style	Получить свойство (*)
iNurbsParam.style = style	Установить свойство (*)
style = iNurbsParam.GetStyle()	Получить свойство (**)
iNurbsParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

ksNurbsParam – методы

GetPKnot – Получить динамический массив узлов сплайна ksDynamicArray типа DOUBLE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPKnot();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray узлов сплайна типа DOUBLE_ARR.

Примечания:

Если для описания NURBS выбран вариант ALLPARAM или SHEET_ALLPARAM, то способ заполнения массива точек определяется значением параметра Close:

- Если Close=0 (NURBS замкнут) - параметры с разжатым узловым вектором,
- Если Close=1 (NURBS разомкнут) - параметры с зажатым узловым вектором.

Если для описания NURBS выбран вариант NURBS_CLAMPED_PARAM или NURBS_CLAMPED_SHEETPARAM, то, вне зависимости от значения параметра Close, возвращаются параметры с зажатым узловым вектором.

GetPPoint – Получить динамический массив точек сплайна типа NURBS_POINT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPPoint();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray точек сплайна типа NURBS_POINT_ARR.

Примечания:

Если для описания NURBS выбран вариант ALLPARAM или SHEET_ALLPARAM, то способ заполнения массива точек определяется значением параметра Close:

- Если Close=0 (NURBS замкнут) - параметры с разжатым узловым вектором,
- Если Close=1 (NURBS разомкнут) - параметры с зажатым узловым вектором.

Если для описания NURBS выбран вариант NURBS_CLAMPED_PARAM или NURBS_CLAMPED_SHEETPARAM, то, вне зависимости от значения параметра Close, возвращаются параметры с зажатым узловым вектором.

Смотрите также: ksNurbsPointParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры NURBS.
2. Динамические массивы не создаются.

SetPKnot – Установить динамический массив узлов сплайна ksDynamicArray типа DOUBLE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPKnot (LPDISPATCH pKnot);

Входной параметр:

pKnot - указатель на интерфейс динамического массива ksDynamicArray типа DOUBLE_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

SetPPoint – Установить динамический массив точек сплайна ksDynamicArray типа NURBS_POINT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPPoint (LPDISPATCH pPoint);

Входной параметр:

pPoint - указатель на интерфейс динамического массива ksDynamicArray точек сплайна типа NURBS_POINT_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Смотрите также: ksNurbsPointParam

Кривая Безье

Узел кривой Безье (Интерфейс ksBezierPointParam)

Интерфейс параметров узла для кривой Безье.

Аналог данных параметров при использовании API экспортных функций - BezierPointParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksBezierPointParam – свойства

ang – Угол наклона касательной к кривой в базовой точке

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iBezierPointParam.ang Получить свойство (*)
iBezierPointParam.ang = ang Установить свойство (*)
ang = iBezierPointParam.GetAng() Получить свойство (**)

iBezierPointParam.SetAng(ang) Установить свойство (**)

left – Расстояние от базовой точки до левой точки узла

Интерфейс...

Тип данных:double.

Синтаксис Automation:

left = iBezierPointParam.left Получить свойство (*)
iBezierPointParam.left = left Установить свойство (*)
left = iBezierPointParam.GetLeft() Получить свойство (**)
iBezierPointParam.SetLeft(left) Установить свойство (**)

right – Расстояние от базовой точки до правой точки узла

Интерфейс...

Тип данных:double.

Синтаксис Automation:

right = iBezierPointParam.right Получить свойство (*)
iBezierPointParam.right = right Установить свойство (*)
right = iBezierPointParam.GetRight() Получить свойство (**)
iBezierPointParam.SetRight(right) Установить свойство (**)

x, y – Координаты базовой точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x = iBezierPointParam.x Получить свойство (*)
iBezierPointParam.x = x Установить свойство (*)
x = iBezierPointParam.GetX() Получить свойство (**)
iBezierPointParam.SetX(x) Установить свойство (**)

ksBezierPointParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры узла для кривой Безье.

Кривая Безье (Интерфейс ksBezierParam)

Интерфейс параметров кривой Безье.

Аналог данных параметров при использовании API экспортных функций - BezierParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksBezierParam – свойства

closed – Признак замыкания кривой

Интерфейс...

Тип данных:short.

Значения свойства:

0 - кривая не замкнута,
1 - кривая замкнута.

Синтаксис Automation:

closed = iBezierParam.closed	Получить свойство (*)
iBezierParam.closed = closed	Установить свойство (*)
closed = iBezierParam.GetClosed()	Получить свойство (**)
iBezierParam.SetClosed(closed)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iBezierParam.style	Получить свойство (*)
iBezierParam.style = style	Установить свойство (*)
style = iBezierParam.GetStyle()	Получить свойство (**)
iBezierParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

ksBezierParam – методы

GetMathPointArr – Получить указатель на интерфейс динамического массива параметров математических точек сплайна ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetMathPointArr();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray параметров математических точек типа POINT_ARR.

Смотрите также: ksMathPointParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод обнуляет все параметры кривой Безье.

SetMathPointArr – Установить динамический массив параметров математических точек сплайна ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetMathPointArr (LPDISPATCH pMathPoint);

Входной параметр:

pMathPoint	- указатель на интерфейс динамического массива ksDynamicArray параметров математических точек типа POINT_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Смотрите также: ksMathPointParam

Коническое сечение (Интерфейс ksConicArcParam)

Интерфейс параметров конического сечения.

Аналог данных параметров при использовании API экспортных функций - ConicArcParam

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksConicArcParam – свойства

A, B, C, D, E, F – Коэффициенты канонического уравнения

Интерфейс...

Тип данных:double.

Синтаксис Automation:

A = iConicArcParam.A	Получить свойство (*)
iConicArcParam.A = A	Установить свойство (*)
A = iConicArcParam.GetA()	Получить свойство (**)
iConicArcParam.SetA(A)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iConicArcParam.style	Получить свойство (*)
iConicArcParam.style = style	Установить свойство (*)
style = iConicArcParam.GetStyle()	Получить свойство (**)
iConicArcParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

x1, y1 – Координаты начальной точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x1 = iConicArcParam.x1	Получить свойство (*)
iConicArcParam.x1 = x1	Установить свойство (*)
x1 = iConicArcParam.GetX1()	Получить свойство (**)
iConicArcParam.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты конечной точки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x2 = iConicArcParam.x2	Получить свойство (*)
iConicArcParam.x2 = x2	Установить свойство (*)
x2 = iConicArcParam.GetX2()	Получить свойство (**)
iConicArcParam.SetX2(x2)	Установить свойство (**)

ksConicArcParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры для построения конического сечения.

Контур (Интерфейс ksContourParam)

Справка системы КОМПАС...

`KOMPAS.chm::/CM_ASSEMBLYCONTOUR.htm`

Интерфейс параметров контура.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksContourParam- свойства

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iContourParam.style	Получить свойство (*)
iContourParam.style = style	Установить свойство (*)
style = iContourParam.GetStyle()	Получить свойство (**)
iContourParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

ksContourParam – методы

Init – Инициализировать параметры

Интерфейс..

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры контура.

Эквидистанта (Интерфейс ksEquidistantParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_EQUID_TO_OBJ.htm

Интерфейс параметров эквидистанты.

Аналог данных параметров при использовании API экспортных функций - EquidistantParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksEquidistantParam – свойства

cutMode – Тип обхода углов контура

Интерфейс...

Тип данных:short.

Значения свойства:

0	- обход срезом,
1	- обход дугой.

Синтаксис Automation:

cutMode = iEquidistantParam.cutMode	Получить свойство (*)
iEquidistantParam.cutMode = cutMode	Установить свойство (*)
cutMode = iEquidistantParam.GetCutMode()	Получить свойство (**)
iEquidistantParam.SetCutMode(cutMode)	Установить свойство (**)

degState – Признак построения вырожденных сегментов эквидистанты

Интерфейс...

Тип данных:BOOL.

Значения свойства:

0	- вырожденные сегменты не отрисовываются,
1	- вырожденные сегменты отрисовываются.

Синтаксис Automation:

degState = iEquidistantParam.degState	Получить свойство (*)
iEquidistantParam.degState = degState	Установить свойство (*)
degState = iEquidistantParam.GetDegState()	Получить свойство (**)
iEquidistantParam.SetDegState(degState)	Установить свойство (**)

geoObj – Графический объект – базовая кривая эквидистанты

Интерфейс...

Тип данных:long.

Синтаксис Automation:

geoObj = iEquidistantParam.geoObj	Получить свойство (*)
iEquidistantParam.geoObj = geoObj	Установить свойство (*)
geoObj = iEquidistantParam.GetGeoObj()	Получить свойство (**)
iEquidistantParam.SetGeoObj(geoObj)	Установить свойство (**)

radRight – Радиус эквидистанты справа по направлению базовой кривой

Интерфейс...

Тип данных:double.

Синтаксис Automation:

radRight = iEquidistantParam.radRight	Получить свойство (*)
iEquidistantParam.radRight = radRight	Установить свойство (*)
radRight = iEquidistantParam.GetRadRight()	Получить свойство (**)
iEquidistantParam.SetRadRight(radRight)	Установить свойство (**)

radLeft – Радиус эквидистанты слева по направлению базовой кривой

Интерфейс...

Тип данных:double.

Синтаксис Automation:

radLeft = iEquidistantParam.radLeft	Получить свойство (*)
iEquidistantParam.radLeft = radLeft	Установить свойство (*)
radLeft = iEquidistantParam.GetRadLeft()	Получить свойство (**)
iEquidistantParam.SetRadLeft(radLeft)	Установить свойство (**)

side – Признак, с какой стороны от базового объекта строить эквидистанту

Интерфейс...

Тип данных:short.

Значения свойства:

0	- слева по направлению,
1	- справа по направлению,
2	- с двух сторон.

Синтаксис Automation:

side = iEquidistantParam.side	Получить свойство (*)
iEquidistantParam.side = side	Установить свойство (*)
side = iEquidistantParam.GetSide()	Получить свойство (**)
iEquidistantParam.SetSide(side)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iEquidistantParam.style	Получить свойство (*)
iEquidistantParam.style = style	Установить свойство (*)
style = iEquidistantParam.GetStyle()	Получить свойство (**)
iEquidistantParam.SetStyle(style)	Установить свойство (**)

Системные стили линий...

ksEquidistantParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры эквидистанты.

Растровая вставка (Интерфейс ksRasterParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTRASTER.htm

Интерфейс параметров растрового объекта.

Аналог данных параметров при использовании API экспортных функций - RasterParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRasterParam – свойства

embeded – Признак внедрения растра или ссылки на него

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- внедрять данные в объект,

FALSE - связывать объект с файлом-источником данных на диске.

Синтаксис Automation:

```
embeded = iRasterParam.embeded                  Получить свойство (* )
iRasterParam.embeded = embeded                  Установить свойство (* )
embeded = iRasterParam.GetEmbeded()             Получить свойство (**)
iRasterParam.SetEmbeded( embeded )              Установить свойство (**)
```

Примечание:

Значение FALSE в данный момент не используется (связь растрового объекта с файлом на диске в КОМПАС-ГРАФИК не реализована).

fileName – Полный путь к файлу с растровым изображением

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

```
fileName = iRasterParam.fileName                Получить свойство (* )
iRasterParam.fileName = fileName                Установить свойство (* )
fileName = iRasterParam.GetFileName()           Получить свойство (**)
iRasterParam.SetFileName( fileName )            Установить свойство (**)
```

ksRasterParam – методы

GetPlace – Получить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlace();

Возвращаемое значение:

- указатель на интерфейс ksPlacementParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры растрового объекта.

SetPlace – Установить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

BOOL SetPlace (LPDISPATCH place);

Входной параметр:

place	- указатель на интерфейс ksPlacementParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Вставка фрагмента (Интерфейс ksInsertFragmentParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Интерфейс параметров вставки фрагментов.

Аналог данных параметров при использовании API экспортных функций - InsertFragmentParamEx.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksInsertFragmentParam – свойства

comment – Имя вставки фрагмента

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

comment = iInsertFragmentParam.comment	Получить свойство (*)
iInsertFragmentParam.comment = comment	Установить свойство (*)
comment = iInsertFragmentParam.GetComment()	Получить свойство (**)
iInsertFragmentParam.SetComment(comment)	Установить свойство (**)

fileName – Имя файла фрагмента

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

fileName = iInsertFragmentParam.fileName	Получить свойство (*)
iInsertFragmentParam.fileName = fileName	Установить свойство (*)
fileName = iInsertFragmentParam.GetFileName()	Получить свойство (**)
iInsertFragmentParam.SetFileName(fileName)	Установить свойство (**)

Примечание:

Для локального фрагмента свойство равно нулю.

insertType – Способ вставки фрагмента

Интерфейс...

Тип данных:short.

Значения свойства:

0	- взять в документ,
1	- вставить внешней ссылкой,
3	- вставить локальный фрагмент.

Синтаксис Automation:

insertType = iInsertFragmentParam.insertType	Получить свойство (*)
iInsertFragmentParam.insertType = insertType	Установить свойство (*)
insertType = iInsertFragmentParam.GetInsertType()	Получить свойство (**)
iInsertFragmentParam.SetInsertType(insertType)	Установить свойство (**)

multiLayer – Признак размещения объектов из вставленного фрагмента по слоям

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- объекты на тех же слоях, что и во фрагменте-источнике,
FALSE	- объекты на одном слое.

Синтаксис Automation:

multiLayer = iInsertFragmentParam.multiLayer	Получить свойство (*)
iInsertFragmentParam.multiLayer = multiLayer	Установить свойство (*)
multiLayer = iInsertFragmentParam.GetMultiLayer()	Получить свойство (**)
iInsertFragmentParam.SetMultiLayer(multiLayer)	Установить свойство (**)

ksInsertFragmentParam – методы

GetPlace – Получить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlace();

Возвращаемое значение:

- указатель на интерфейс параметров местоположения ksPlacementParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры вставки фрагментов.

SetPlace – Установить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

BOOL SetPlace(LPDISPATCH place);

Входной параметр:

place	- указатель на интерфейс параметров местоположения ksPlacementParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Вставка фрагмента (Интерфейс ksInsertFragmentParamEx)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Интерфейс параметров вставки фрагментов (расширенный).

Аналог данных параметров при использовании API экспортных функций - InsertFragmentParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksInsertFragmentParamEx – свойства

comment – Имя вставки фрагмента

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

comment = iInsertFragmentParam.comment	Получить свойство (*)
iInsertFragmentParam.comment = comment	Установить свойство (*)
comment = iInsertFragmentParam.GetComment()	Получить свойство (**)
iInsertFragmentParam.SetComment(comment)	Установить свойство (**)

fileName – Имя файла фрагмента

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

fileName = iInsertFragmentParam.fileName	Получить свойство (*)
iInsertFragmentParam.fileName = fileName	Установить свойство (*)
fileName = iInsertFragmentParam.GetFileName()	Получить свойство (**)
iInsertFragmentParam.SetFileName(fileName)	Установить свойство (**)

Примечание:

Для локального фрагмента свойство равно нулю.

insertType – Способ вставки фрагмента

Интерфейс...

Тип данных:short.

Значения свойства:

0	- взять в документ,
1	- вставить внешней ссылкой,
3	- вставить локальный фрагмент.

Синтаксис Automation:

insertType = iInsertFragmentParam.insertType	Получить свойство (*)
iInsertFragmentParam.insertType = insertType	Установить свойство (*)
insertType = iInsertFragmentParam.GetInsertType()	Получить свойство (**)
iInsertFragmentParam.SetInsertType(insertType)	Установить свойство (**)

multiLayer – Признак размещения объектов из вставленного фрагмента по слоям

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- объекты на тех же слоях, что и во фрагменте-источнике,
FALSE	- объекты на одном слое.

Синтаксис Automation:

multiLayer = iInsertFragmentParam.multiLayer	Получить свойство (*)
iInsertFragmentParam.multiLayer = multiLayer	Установить свойство (*)
multiLayer = iInsertFragmentParam.GetMultiLayer()	Получить свойство (**)
iInsertFragmentParam.SetMultiLayer(multiLayer)	Установить свойство (**)

scaleProjLinesSize – Признак масштабирования длины выносных линий размеров

Интерфейс...

Тип данных:short.

Значения свойства:

0	- выносные линии не масштабируются,
1	- выносные линии масштабируются.

Синтаксис Automation:

scaleProjLinesSize = iKompasObject.ScaleProjLinesSize	Получить свойство (*)
iKompasObject.ScaleProjLinesSize = scaleProjLinesSize	Установить свойство (*)
scaleProjLinesSize = iKompasObject.GetScaleProjLinesSize()	Получить свойство (**)
iKompasObject.SetScaleProjLinesSize (scaleProjLinesSize)	Установить свойство (**)

ksInsertFragmentParamEx – методы

GetPlace – Получить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPlace();

Возвращаемое значение:

- указатель на интерфейс параметров местоположения ksPlacementParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры вставки фрагментов.

SetPlace – Установить указатель на интерфейс параметров местоположения

Интерфейс...

Синтаксис Automation:

BOOL SetPlace(LPDISPATCH place);

Входной параметр:

Place	- указатель на интерфейс параметров местоположения ksPlacementParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Интерфейс параметров осевой линии (Интерфейс ksAxisLineParam)

Интерфейс параметров осевой линии.

Аналог данных параметров при использовании API экспортных функций - AxisLineParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

ksAxisLineParam – методы

GetBegPoint – Получить указатель на интерфейс параметров начальной точки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetBegPoint();

Возвращаемое значение:

- указатель на интерфейс ksMathPointParam.

GetEndPoint – Получить указатель на интерфейс параметров конечной точки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetEndPoint();

Возвращаемое значение:

- указатель на интерфейс ksMathPointParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Параметры графического документа (Интерфейс ksDocumentParam)

Справка системы КОМПАС...

KOMPAS.chm::/DOCUMENT_INFO_DIALOG.htm

Интерфейс параметров документа.

Аналог интерфейса при использовании API экспортных функций структура параметров DocumentParam.

ksDocumentParam - интерфейс Automation

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksDocumentParam – свойства

author – Автор

Интерфейс...

Тип данных:BSTR.

Синтаксис Automation:

author = iDocumentParam.author	Получить свойство (*)
iDocumentParam.author = author	Установить свойство (*)
author = iDocumentParam.GetAuthor()	Получить свойство (**)
iDocumentParam.SetAuthor(author)	Установить свойство (**)

comment – Комментарий

Интерфейс...

Тип данных:BSTR.

Синтаксис Automation:

comment = iDocumentParam.comment	Получить свойство (*)
iDocumentParam.comment = comment	Установить свойство (*)
comment = iDocumentParam.GetComment()	Получить свойство (**)
iDocumentParam.SetComment(comment)	Установить свойство (**)

fileName – Имя файла

Интерфейс...

Тип данных:BSTR.

Синтаксис Automation:

```
regime = iDocumentParam.regime              Получить свойство (* )
iDocumentParam.regime = regime              Установить свойство (* )
regime = iDocumentParam.GetRegime()         Получить свойство (**)
iDocumentParam.SetRegime(regime)            Установить свойство (**)
```

regime – Режим редактирования

Интерфейс...

Тип данных:short.

Значения свойства:

```
0                    - видимый,
1                    - "слепой".
```

Синтаксис Automation:

```
fileName = iDocumentParam.fileName          Получить свойство (* )
iDocumentParam.fileName = fileName          Установить свойство (* )
fileName = iDocumentParam.GetFileName()     Получить свойство (**)
iDocumentParam.SetFileName( fileName )      Установить свойство (**)
```

ksDocumentParam – методы

GetLayoutParam – Получить указатель на интерфейс параметров оформления документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetLayoutParam();

Возвращаемое значение:

- указатель на интерфейс параметров оформления документа ksSheetPar.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры чертежа или фрагмента.

type – Тип документа

Интерфейс...

Тип данных:short.

Значения свойства:

Из перечисления DocType.

Синтаксис Automation:

type = iDocumentParam.type	Получить свойство (*)
iDocumentParam.type = type	Установить свойство (*)
type = iDocumentParam.GetType()	Получить свойство (**)
iDocumentParam.SetType(type)	Установить свойство (**)

Фрагмент (Интерфейс ksFragment)

Справка системы КОМПАС...

KOMPAS.chm::/454_Glava52_Obshchie_svedenija_.htm

Интерфейс фрагмента.

Примечание:

Указатель на интерфейс можно получить при помощи метода ksDocument2D::GetFragment.

Смотрите также: KompasObject

ksFragment – методы

ksCloseLocalFragmentDefinition – Закончить определение локального фрагмента

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данного метода при использовании API экспортных функций - CloseLocalFragmentDefinition.

Синтаксис Automation:

long ksCloseLocalFragmentDefinition();

Возвращаемое значение:

указатель на определение фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Все объекты вида, вводимые между операторами ksLocalFragmentDefinition и ksCloseLocalFragmentDefinition, принадлежат локальному фрагменту.

Возвращаемый указатель используется в функциях ksInsertFragment, ksDeleteObj.

ksFragmentDefinition – Определить фрагмент для вставки

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данного метода при использовании API экспортных функций - FragmentDefinition.

Синтаксис Automation:

long ksFragmentDefinition (LPCTSTR fileName,

LPCTSTR comment,

short insertType);

Входные параметры:

fileName	- имя файла фрагмента,
comment	- имя вставки,
insertType	- тип вставки (действителен для внешнего фрагмента):
	0 - взять в документ,
	1 - внешней ссылкой.

Возвращаемое значение:

указатель на определение фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция определяет данные для вставки фрагмента (внешнего или находящегося непосредственно в документе) с указанным именем по ссылке.
2. Непосредственно вставка фрагмента (вставка ссылки) производится

   с помощью метода ksFragment::ksInsertFragment.
3. Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где
   c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

   |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае определение создается для фрагмента из библиотеки фрагментов.

4. Определение для данного фрагмента одно - вставок может быть сколько угодно.

ksInsertFragment – Вставить фрагмент в документ

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Интерфейс...

Аналог данного метода при использовании API экспортных функций - InsertFragment.

Замечание:

Данный метод устарел. Рекомендуется использовать вместо него метод ksInsertFragmentEx.

Синтаксис Automation:

long ksInsertFragment (long p,

BOOL curentLayer,

LPDISPATCH par);

Входные параметры:

p	- указатель на фрагмент,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на интерфейс ksPlacementParam.

Возвращаемое значение:

указатель на вставку фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

ksInsertFragmentEx – Вставить фрагмент в документ

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данного метода при использовании API экспортных функций - ksInsertFragmentEx.

Синтаксис Automation:

long ksInsertFragmentEx (long p,

BOOL curentLayer,

LPDISPATCH par,

BOOL scaleProjLinesSize);

Входные параметры:

p	- указатель на фрагмент,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на интерфейс ksPlacementParam,
scaleProjLinesSize	- признак масштабирования длины выносных линий размеров: TRUE - выносные линии масштабируются, FALSE - выносные линии не масштабируются.

Возвращаемое значение:

указатель на вставку фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

ksLocalFragmentDefinition – Определить данные для последующей вставки локального фрагмента по ссылке

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данного метода при использовании API экспортных функций - LocalFragmentDefinition.

Синтаксис Automation:

long ksLocalFragmentDefinition (LPCTSTR comment);

Входной параметр:

comment	- имя вставки.

Возвращаемое значение:

указатель на определение фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Вставка ссылки на фрагмент производится с помощью функции ksInsertFragment. Указатель на определение фрагмента возвращается при окончании описания - функцией ksCloseLocalFragmentDefinition.

Отдельного файла фрагмента нет; "тело" фрагмента хранится в документе.

ksReadFragment – Вставить фрагмент россыпью в документ

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данного метода при использовании API экспортных функций - ReadFragment.

Синтаксис Automation:

long ksReadFragment (LPCTSTR fileName,

BOOL curentLayer,

LPDISPATCH par);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на интерфейс ksPlacementParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

ksReadFragmentToGroup – Вставить фрагмент россыпью во временную группу

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksReadFragmentToGroup.

Замечание:

Данный метод устарел. Рекомендуется использовать вместо него метод ksReadFragmentToGroupEx.

Синтаксис Automation:

long ksReadFragmentToGroup (LPCTSTR fileName,

BOOL curentLayer,

LPDISPATCH par);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на интерфейс ksPlacementParam.

Возвращаемое значение:

указатель на временную группу	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

ksReadFragmentToGroupEx – Вставить фрагмент россыпью во временную группу

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_INSERTFRAGMENT.htm`

Аналог данного метода при использовании API экспортных функций - ksReadFragmentToGroupEx.

Синтаксис Automation:

long ksReadFragmentToGroup (LPCTSTR fileName,

BOOL curentLayer,

LPDISPATCH par

BOOL scaleProjLinesSize);

Входные параметры:

fileName	- имя фрагмента,

curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на интерфейс ksPlacementParam,
scaleProjLinesSize	- признак масштабирования длины выносных линий размеров: TRUE - выносные линии масштабируются, FALSE - выносные линии не масштабируются.

Возвращаемое значение:

указатель на временную группу	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

ksWriteFragment – Записать группу во фрагмент

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_INSERTFRAGMENT.htm`

Аналог данного метода при использовании API экспортных функций - WriteFragment.

Синтаксис Automation:

long ksWriteFragment (long gr,

LPCTSTR fileName,

LPCTSTR comment,

double xb,

double yb);

Входные параметры:

gr	- указатель на группу (0 - записывается группа селектирования),
filename	- имя файла фрагмента,
comment	- комментарий для фрагмента,
xb, yb	- координаты точки привязки.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Основная надпись (Интерфейс ksStamp)

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Интерфейс основной надписи.

Примечание:

Указатель на интерфейс можно получить при помощи методов ksDocument2D::GetStamp, ksDocument2D::GetStampEx, ksSpcDocument::GetStamp, ksSpcDocument::GetStampEx, ksDocumentTxt::GetStamp, ksDocumentTxt::GetStampEx.

ksStamp – свойства

reference – Указатель на основную надпись

Интерфейс...

Тип данных:long.

Синтаксис Automation:

ref = iStamp.reference	Получить свойство (*)
iStamp.reference = ref	Установить свойство (*)
ref = iStamp.GetReference()	Получить свойство (**)
iStamp.SetReference (ref)	Установить свойство (**)

ksStamp – методы

GetSheetNumb – Получить номер листа

Интерфейс...

Синтаксис Automation:

long GetSheetNumb ();

Возвращаемое значение:

номер листа

ksClearStamp – Очистить основную надпись

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данного метода при использовании API экспортных функций - ClearStamp.

Синтаксис Automation:

long ksClearStamp (long numb);

Входной параметр:

numb	- номер очищаемой ячейки основной надписи (0 - очищается вся основная надпись).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод действителен для чертежа и спецификации.

ksCloseStamp – Закрыть основную надпись

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данного метода при использовании API экспортных функций - CloseStamp.

Синтаксис Automation:

long ksCloseStamp();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод действителен для чертежа и спецификации.

ksColumnNumber – Определить номер ячейки основной надписи

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ColumnNumber.

Синтаксис Automation:

long ksColumnNumber (long numb);

Входной параметр:

numb - номер ячейки основной надписи.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Метод используется в режиме редактирования основной надписи.

ksGetStampColumnText – Получить текст ячейки основной надписи

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данного метода при использовании API экспортных функций - GetStampColumnText.

Синтаксис Automation:

LPDISPATCH ksGetStampColumnText (long* numb);

Выходной параметр:

numb - номер ячейки основной надписи.

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray строк текста типа TEXT_LINE_ARR или NULL, если все ячейки пройдены.

Смотрите также : ksTextLineParam

Примечание:

После выполнения метод смещается на следующую ячейку.

Если номер графы не определен с помощью функции ksColumnNumber, метод начинает работу с первой графы.

После использования полученный массив желательно удалить.

Если numb != NULL, метод возвращает номер графы.

Функция используется в режиме редактирования штампа.

ksOpenStamp – Открыть основную надпись

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данного метода при использовании API экспортных функций - OpenStamp.

Синтаксис Automation:

long ksOpenStamp();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод действителен для чертежа и спецификации.

Операторы ksStamp::ksColumnNumber и ksStamp::ksTextLin, вводимые между операторами ksStamp::ksOpenStamp и ksStamp::ksCloseStamp, принадлежат основной надписи.

ksStamp::ksColumnNumber определяет номер ячейки, куда помещать текст.

Номера определены в соответствии с ГОСТом на данную основную надпись.

ksStamp::ksCloseStamp возвращает указатель на штамп.

ksSetStampColumnText – Задать текст в ячейке

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetStampColumnText.

Синтаксис Automation:

long ksSetStampColumnText (long numb,

LPDISPATCH textArr);

Входные параметры:

numb	- номер ячейки основной надписи,
textArr	- указатель на интерфейс динамического массива ksDynamicArray строк текста типа TEXT_LINE_ARR.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод используется в режиме редактирования штампа.

ksSetTextLineAlign – Установить выравнивание текста

Интерфейс...

Справка системы КОМПАС...

kompas.chm::/528_65_5_1_Izmenenie_parametrov.htm

Аналог данного метода при использовании API экспортных функций - ksSetTextLineAlign.

Синтаксис Automation:

long ksSetTextLineAlign (short align);

Входной параметр:

align	- признак выравнивания: 0 - по левому краю, 1 - по центру, 2 - по правому краю, 3 - по ширине.

Возвращаемое значение:

предыдущий признак	- в случае успешного завершения,
-1	- в случае неудачи.

ksTextLine – Задать компоненту строки текста или строку текста полностью

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TextLine.

Синтаксис Automation:

ksTextLine(LPDISPATCH textItem);

Входной параметр:

textItem	- указатель на интерфейс ksTextItemParam для компоненты текста или указатель на интерфейс ksTextLineParam для всей строки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Данная функция предназначена для задания сложно-структурированного текста, отдельные компоненты которого имеют различные параметры.

Интерфейс результатов редактирования объекта (Интерфейс ksObject2DNotifyResult, IObject2DNotifyResult)

Интерфейс результатов редактирования объекта.

ksObject2DNotifyResult	- интерфейс Automation
IObject2DNotifyResult	- интерфейс COM

В Automation получить интерфейс результата можно при помощи метода ksDocument2D::GetObject2DNotifyResult.

В COM получить интерфейс результата можно при помощи экспортной функции ksGetObject2DNotifyResult.

Примечание:

В процессе обработки события Object2DNotifyEnum можно получить информацию о редактировании объекта.

ksObject2DNotifyResult,IObject2DNotifyResult – методы

GetAngle – Получить угол поворота объекта

Интерфейс...

Синтаксис Automation:

double GetAngle();

Синтаксис COM:

double GetAngle();

Возвращаемое значение:

- угол поворота объекта.

GetCopyObject – Получить копию объекта, если выполнялась операция копирования

Интерфейс...

Синтаксис Automation:

long GetCopyObject();

Синтаксис COM:

long GetCopyObject();

Возвращаемое значение:

- указатель на объект.

GetSheetPoint – Получить координаты точки в системе координат листа

Интерфейс...

Синтаксис Automation:

BOOL GetSheetPoint (BOOL from, double *x, double *y);

Синтаксис COM:

BOOL GetSheetPoint (BOOL from, double *x, double *y);

Входные параметры:

from	- TRUE - начало сдвига, центр поворота, центр масштабирования, первая точка оси симметрии, начало копирования, - FALSE - конец сдвига, вторая точка оси симметрии, конец копирования.

Выходные параметры:

x, y,	- значения координат точки в системе координат листа.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

GetNotifyType – Получить тип события

Интерфейс...

Синтаксис Automation:

long GetNotifyType();

Синтаксис COM:

long GetNotifyType();

Возвращаемое значение:

- тип события.

GetProcessType – Тип процесса

Интерфейс...

Синтаксис:

long GetProcessType();

Возвращаемое значение:

Тип процесса из перечисления ProcessTypeEnum	- в случае успеха.

GetScale – Получить коэффициенты масштабирования по координатным осям

Интерфейс...

Синтаксис Automation:

BOOL GetScale (double *sx, double *sy);

Синтаксис COM:

BOOL GetScale (double *sx, double *sy);

Выходные параметры:

Sx	- масштаб по оси X,
Sy	- масштаб по оси Y.

Возвращаемое значение:

TRUE	- в случае успеха,
FALSE	- в случае неудачи.

IsCopy – Получить признак копирования исходных объектов

Интерфейс...

Синтаксис Automation:

BOOL IsCopy();

Синтаксис COM:

BOOL IsCopy();

Возвращаемое значение:

TRUE	- процесс с копированием,
FALSE	- процесс без копирования.

IsRedoMode – Признак работы команды Redo

Интерфейс...

Синтаксис Automation:

BOOL IsRedoMode();

Синтаксис COM:

BOOL IsRedoMode();

Возвращаемое значение:

TRUE - если работает команда Redo.

Примечание:

Свойство требуется проверять в событиях, если необходимо отличать событие, полученное от визуальной работы пользователя, от действий, вызванных командой **Повторить** (т.е. вернуть отмененное действие).

См. также: ksObject2DNotifyResult::IsUndoMode

IsUndoMode – Признак работы команды Undo

Интерфейс...

Синтаксис Automation:

BOOL IsUndoMode();

Синтаксис COM:

BOOL IsUndoMode();

Возвращаемое значение:

TRUE - если работает команда Undo.

Примечание:

Свойство требуется проверять в событиях, если необходимо отличать событие, полученное от визуальной работы пользователя, от действий, вызванных командой **Отменить**.

Например, при отмене создания объекта посылается событие удаления объекта.

При отмене удаления объекта посылается событие создания объекта.

См. также: ksObject2DNotifyResult::IsRedoMode

Источник событий менеджера выделенных объектов (Интерфейс SelectionMngNotify)

Источник событий менеджера выделенных объектов.

В Automation источник событий для графического документа SelectionMngNotify можно получить при помощи метода ksDocument2D::GetSelectionMngNotify.

Для документа-модели источником является Менеджер выделенных объектов ksSelectionMng.

В COM контейнером для подписки на данный интерфейс является указатель на графический документ или документ-модель.

Интерфейс событий менеджера выделенных объектов...

Примечание:

Интерфейс позволяет следить за выделением объектов в документе.

Источник событий объектов графического документа (Интерфейс Object2DNotify)

Источник событий объектов графического документа.

В Automation источник событий для графического документа Object2DNotify можно получить при помощи метода ksDocument2D::GetObject2DNotify.

В COM контейнером для подписки на данный интерфейс является указатель на графический документ или документ-модель.

Интерфейс событий объектов графического документа...

Примечание:

Интерфейс позволяет следить и управлять редактированием объектов графического документа.

Источник событий графического документа (Интерфейс Document2DNotify)

Источник событий графического документа.

В Automation источник событий для подписки Document2DNotify можно получить при помощи метода ksDocument2D::GetDocument2DNotify.

В COM источником для подписки является графический документ.

Интерфейс событий графического документа...

Интерфейс привязки (Интерфейс ksSnapOptions)

Справка системы КОМПАС...

KOMPAS.chm::/85_8_2_1_Globalqnaja_privjazka.htm

Интерфейс параметров привязок в графическом документе.

Аналог данных параметров при использовании API экспортных функций - SnapOptrions.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksSnapOptions – свойства

angSnap – Глобальная "Угловая привязка"

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

```
angSnap = iSnapOptrions.angSnap                 Получить свойство (* )
iSnapOptrions.angSnap = angSnap                 Установить свойство (* )
angSnap = iSnapOptrions.GetAngSnap()            Получить свойство (**)
iSnapOptrions.SetAngSnap (angSnap)              Установить свойство (**)
```

angleStep – Шаг угловой привязки (в градусах)

Интерфейс...

Тип данных: double.

Синтаксис Automation:

```
angleStep = iSnapOptrions.angleStep             Получить свойство (* )
iSnapOptrions.angleStep = angleStep             Установить свойство (* )
angleStep = iSnapOptrions.GetAngleStep()        Получить свойство (**)
iSnapOptrions.SetAngleStep (angleStep)          Установить свойство (**)
```

commonOpt – Общие настройки привязок

Интерфейс...

Тип данных:long.

Общие настройки привязок...

Синтаксис Automation:

```
commonOpt = iSnapOptrions.commonOpt             Получить свойство (* )
iSnapOptrions.commonOpt = commonOpt             Установить свойство (* )
commonOpt = iSnapOptrions.GetCommonOpt()        Получить свойство (**)
iSnapOptrions.SetCommonOpt (commonOpt)          Установить свойство (**)
```

grid – Глобальная привязка "По сетке"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

```
grid = iSnapOptrions.grid                Получить свойство (* )
iSnapOptrions.grid = grid                Установить свойство (* )
grid = iSnapOptrions.GetGrid()           Получить свойство (**)
iSnapOptrions.SetGrid (grid)             Установить свойство (**)
```

intersect – Глобальная привязка "Пересечение"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- привязка включена,
FALSE	- привязка выключена.

Синтаксис Automation:

```
intersect = iSnapOptrions.intersect              Получить свойство (* )
iSnapOptrions.intersect = intersect              Установить свойство (* )
intersect = iSnapOptrions.GetIntersect()         Получить свойство (**)
iSnapOptrions.SetIntersect (intersect)           Установить свойство (**)
```

localSnap –Тип локальной привязки

Интерфейс...

Тип данных:short.

Типы локальной привязки...

Синтаксис Automation:

```
localSnap = iSnapOptrions.localSnap              Получить свойство (* )
iSnapOptrions.localSnap = localSnap              Установить свойство (* )
localSnap = iSnapOptrions.GetLocalSnap()         Получить свойство (**)
iSnapOptrions.SetLocalSnap (localSnap)           Установить свойство (**)
```

nearestMiddle – Глобальная привязка "Середина"

Интерфейс...

Тип данных: BOOL.

Значения свойства:

TRUE	- привязка включена,
FALSE	- привязка выключена.

Синтаксис Automation:

nearestMiddle = iSnapOptrions.nearestMiddle — Получить свойство (*)
iSnapOptrions.nearestMiddle = nearestMiddle — Установить свойство (*)
nearestMiddle = iSnapOptrions.GetNearestMiddle() — Получить свойство (**)
iSnapOptrions.SetNearestMiddle (nearestMiddle) — Установить свойство (**)

nearestPoint – Глобальная привязка "Ближайшая точка"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

nearestPoint = iSnapOptrions.nearestPoint — Получить свойство (*)
iSnapOptrions.nearestPoint = nearestPoint — Установить свойство (*)
nearestPoint = iSnapOptrions.GetNearestPoint() — Получить свойство (**)
iSnapOptrions.SetNearestPoint (nearestPoint) — Установить свойство (**)

pointOnCurve – Глобальная привязка "Точка на кривой"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

pointOnCurve = iSnapOptrions.pointOnCurve — Получить свойство (*)
iSnapOptrions.pointOnCurve = pointOnCurve — Установить свойство (*)
pointOnCurve = iSnapOptrions.GetPointOnCurve() — Получить свойство (**)
iSnapOptrions.SetPointOnCurve (pointOnCurve) — Установить свойство (**)

tangentToCurve – Глобальная привязка "Касание"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

tangentToCurve = iSnapOptrions.tangentToCurve Получить свойство (*)
iSnapOptrions.tangentToCurve = tangentToCurve Установить свойство (*)
tangentToCurve = iSnapOptrions.GetTangentToCurve() Получить свойство (**)
iSnapOptrions.SetTangentToCurve (tangentToCurve) Установить свойство (**)

xyAlign – Глобальная привязка "Выравнивание"

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE - привязка включена,
FALSE - привязка выключена.

Синтаксис Automation:

xyAlign = iSnapOptrions.xyAlign Получить свойство (*)
iSnapOptrions.xyAlign = xyAlign Установить свойство (*)
xyAlign = iSnapOptrions.GetXyAlign() Получить свойство (**)
iSnapOptrions.SetXyAlign (xyAlign) Установить свойство (**)

ksSnapOptions – методы

GetCommonOptValue – Получить значение одного из флагов признака commonOpt

Интерфейс...

Синтаксис Automation:

BOOL GetCommonOptValue (long commonOpt);

Входной параметр:

commonOpt - битовый флаг, характеризующий определенную общую настройку привязок.

Возвращаемое значение:

TRUE - данный битовый флаг включен,
FALSE - данный битовый флаг выключен.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры привязок в графическом документе.

SetCommonOptValue – Установить значение одного из флагов признака commonOpt

Синтаксис Automation:

BOOL SetCommonOptValue (long commonOpt,

BOOL state);

Входные параметры:

commonOpt	- битовый флаг, характеризующий определенную общую настройку привязок,
state	- состояние указанного битового флага: TRUE - включен, FALSE - выключен.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Интерфейс параметров перекрывающихся объектов (Интерфейсы ksOverlapObjectOptions)

Справка системы КОМПАС...

KOMPAS.chm::/343_36_7_Ochistka_fona.htm

Интерфейс параметров перекрывающихся объектов.

ksOverlapObjectOptions	- интерфейс Automation

Аналог данных параметров при использовании API экспортных функций - OverlapObjectOptions.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

ksOverlapObjectOptions – свойства

gap – Зазор при перекрывании объектов

Интерфейс...

Тип данных:double.

Синтаксис Automation:

gap = iOverlapObjectOptions.gap	Получить свойство (*)
iOverlapObjectOptions.gap = gap	Установить свойство (*)
gap = iOverlapObjectOptions.GetGap()	Получить свойство (**)
iOverlapObjectOptions.SetGap(gap)	Установить свойство (**)

overlap – Перекрытие объектов

Интерфейс...

Тип данных:BOOL.

Синтаксис Automation:

overlap = iOverlapObjectOptions.overlap	Получить свойство (*)
iOverlapObjectOptions.overlap = overlap	Установить свойство (*)
overlap = iOverlapObjectOptions.GetOverlap()	Получить свойство (**)
iOverlapObjectOptions.SetOverlap(overlap)	Установить свойство (**)

Значения свойства:

TRUE	- перекрывать штриховки и линии при пересечении,
FALSE	- не перекрывать.

ksOverlapObjectOptions – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Интерфейс математических функций (Интерфейс ksMathematic2D)

Интерфейс математических функций.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetMathematic2D.

Смотрите также: KompasObject

ksMathematic2D – методы

ksAngle – Получить угол (в градусах) между осью OX и вектором, заданным двумя точками

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Angle.

Синтаксис Automation:

double ksAngle (double x1,

double y1,

double x2,

double y2);

Входные параметры:

x1, y1	- координаты начальной точки вектора,
x2, y2	- координаты конечной точки вектора.

Возвращаемое значение:

угол в градусах.

ksAtanD – Получить арктангенс аргумента

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данного метода при использовании API экспортных функций - ATanD.

Синтаксис Automation:

double ksAtanD (double x);

Входной параметр:

x	- тангенс угла.

Возвращаемое значение:

- угол (в градусах).

ksCalcInertiaProperties – Получить плоские массово-центровочные характеристики кривой или группы кривых

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MIX.htm

Аналог данного метода при использовании API экспортных функций - ksCalcInertiaProperties.

Синтаксис Automation:

long ksCalcInertiaProperties (long p,

LPDISPATCH properties,

short dimension);

Входные параметры:

p	- указатель на кривую или группу кривых,
dimension	- размерность длины.

Выходные параметры:

properties	- указатель на интерфейс параметров для расчета МЦХ ksInertiaParam.

Размерности длины...

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

Кривые должны быть замкнутыми и представлять собой наружные и внутренние контура.

ksCalcMassInertiaProperties – Получить объемные массово-центровочные характеристики тел вращения или выдавливания, заданных кривой или группой кривых

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_MIX3D.htm

Аналог данного метода при использовании API экспортных функций - ksCalcMassInertiaProperties.

Синтаксис Automation:

long ksCalcMassInertiaProperties (long p,

LPDISPATCH properties,

double density,

double param);

Входные параметры:

p	- указатель на кривую или группу кривых,
density	- плотность тела (г/куб.мм),
param	- параметр тела (для тела вращения - угол раствора в градусах, для тела выдавливания - толщина).

Выходные параметры:

properties	- указатель на интерфейс параметров для расчета массово-центровочных характеристик ksMassInertiaParam.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечания:

1. Кривые должны быть замкнутыми и представлять собой наружные и внутренние контура.
2. Тип тела (тело вращения или выдавливания) устанавливается перед вызовом метода в интерфейсе ksMassInertiaParam.

ksCosD – Получить косинус аргумента

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данного метода при использовании API экспортных функций - CosD.

Синтаксис Automation:

double ksCosD (double x);

Входной параметр:

x - угол в градусах.

Возвращаемое значение:

- косинус аргумента.

ksCouplingCircleCircle – Получить параметры сопрягающих окружностей определенного радиуса и точки сопряжения для двух окружностей

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCouplingCircleCircle.

Синтаксис Automation:

BOOL ksCouplingCircleCircle (double xc1,

double yc1,

double radc1,

double xc2,

double yc2,

double radc2,

double rad,

LPDISPATCH param);

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
rad1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
Rad	- радиус второй окружности.

Выходной параметр:

param	- указатель на интерфейс массива координат точек сопряжения ksCON.

Возвращаемое значение:

TRUE	- точки касания обнаружены,
FALSE	- точки касания не обнаружены.

ksCouplingLineCircle – Получить параметры сопрягающих окружностей определенного радиуса и точки касания при сопряжении окружности и прямой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCouplingLineCircle.

Синтаксис Automation:

BOOL ksCouplingLineCircle (double xc,

double yc,

double radc,

double x1,

double y1,

double angle1,

double rad,

LPDISPATCH param);

Входные параметры:

xc, yc	- координаты центра окружности,
radc	- радиус окружности,
x1, y1	- координаты точки на прямой,
angle1	- угол наклона прямой,
rad	- радиус окружности сопряжения.

Выходной параметр:

param	- указатель на интерфейс массива координат точек сопряжения ksCON.

Возвращаемое значение:

TRUE	- точки касания обнаружены,
FALSE	- точки касания не обнаружены.

ksCouplingLineLine – Получить параметры окружностей, касательных к двум прямым

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CouplingLineLine.

Синтаксис Automation:

BOOL ksCouplingLineLine (double x1,

double y1,

double angle1,

double x2,

double y2,

double angle2,

double rad,

LPDISPATCH param);

Входные параметры:

x1, y1	- координаты точки на первой прямой,
angle1	- угол наклона первой прямой,
x2, y2	- координаты точки на второй прямой,
angle2	- угол наклона второй прямой.

Выходной параметр:

param	- указатель на интерфейс массива координат точек сопряжения ksCON.

Возвращаемое значение:

TRUE	- точки касания обнаружены,
FALSE	- точки касания не обнаружены.

ksDistanceCurveCurve – Добавить проекцию отображения – развертка

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistanceCurveCurve.

Синтаксис Automation:

long ksDistanceCurveCurve(long p1, long p2, double * distanse, double * t1, double * t2);

Входные параметры:

p1	- указатель на первую кривую,
p2	- указатель на вторую кривую.

Выходные параметры:

distanse	- расстояние,
t1	- параметр на кривой 1 в точке с минимальным удалением,
t2	- параметр на кривой 2 в точке с минимальным удалением.

Возвращаемое значение:

1	- успешное завершение,
0	- построить касательную нельзя (кривые совпадают или одна кривая вложена в другую),
-1	- первый объект не существует,
-2	- второй объект не существует,
-3	- кривые расположены в разных видах,
-4	- не совпадают СК определения кривых (геометрическая и аннотационная),
-5	- первый объект не является кривой,
-6	- второй объект не является кривой,
-7	- ошибка.

ksDistancePntArc – Получить расстояние между точкой и дугой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistancePntArc.

Синтаксис Automation:

double ksDistancePntArc (double x,

double y,

double xac,

double yac,

double rada,

double fa1,

double fa2,

short directa);

Входные параметры:

x, y	- координаты точки,
xac, yac	- координаты центра дуги,
rada	- радиус дуги,
fa1, fa2	- начальный и конечный углы дуги,
directa	- направление дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке.

Возвращаемое значение:

- расстояние между точкой и дугой.

ksDistancePntCircle – Получить расстояние между точкой и окружностью

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistancePntCircle.

Синтаксис Automation:

double ksDistancePntCircle (double x,

double y,

double xc,

double yc,

double rad);

Входные параметры:

x, y	- координаты точки,
xc, yc	- координаты центра окружности,
rad	- радиус окружности.

Возвращаемое значение:

- расстояние между точкой и окружностью.

ksDistancePntLine – Получить расстояние между точкой и прямой, заданной точкой и углом

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistancePntLine.

Синтаксис Automation:

double ksDistancePntLine (double x,

double y,

double x1,

double y1,

double angle);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты точки на прямой,
angle	- угол наклона прямой.

Возвращаемое значение:

- расстояние между точкой и прямой.

ksDistancePntLineForPoint – Получить расстояние между точкой и прямой, заданной двумя точками

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistancePntLineForPoint.

Синтаксис Automation:

double ksDistancePntLineForPoint (double x,

double y,

double x1,

double y1,

double x2,

double y2);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты первой точки на прямой,
x2, y2	- координаты второй точки на прямой.

Возвращаемое значение:

- расстояние между точкой и прямой.

ksDistancePntLineSeg – Получить расстояние между точкой и отрезком

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistancePntLineSeg.

Синтаксис Automation:

double ksDistancePntLineSeg (double x,

double y,

double x1,

double y1,

double x2,

double y2);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка.

Возвращаемое значение:

- расстояние между точкой и отрезком.

ksDistancePntPnt – Получить расстояние между двумя точками

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DistancePntPnt.

Синтаксис Automation:

double ksDistancePntPnt (double x1,

double y1,

double x2,

double y2);

Входные параметры:

x1, y1	- координаты первой точки,
x2, y2	- координаты второй точки.

Возвращаемое значение:

- расстояние между двумя точками.

ksDistancePntPntOnCurve – Получить расстояние между двумя точками на кривой

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_DISTANCE_2CURVES.htm

Аналог данного метода при использовании API экспортных функций - ksDistancePntPntOnCurve.

Синтаксис Automation:

double ksDistancePntPntOnCurve (long curve,

double x1,

double y1,

double x2,

double y2);

Входные параметры:

curve	- указатель на кривую,
x1, y1	- координаты первой точки,
x2, y2	- координаты второй точки.

Возвращаемое значение:

расстояние между двумя точками на кривой - в случае успеха,
0 - в случае неудачи.

Примечание:

Если заданные точки не находятся на кривой, то определяется расстояние между их проекциями на кривую.

ksDistanceT1T2OnCurve – Получить расстояние между двумя точками на кривой по параметрам кривой в данных точках

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksDistanceT1T2OnCurve.

Синтаксис Automation:

long ksDistanceT1T2OnCurve (long curve,

double t1,

double t2);

Входные параметры:

curve - указатель на кривую,
t1 - параметр кривой в точке 1,
t2 - параметр кривой в точке 2.

Возвращаемое значение:

- Расстояние между двумя точками на кривой.

ksEqualPoints – Определить эквивалентность (совпадение) двух точек

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksEqualPoints.

Синтаксис Automation:

long ksEqualPoints (double x1,

double y1,

double x2,

double y2);

Входные параметры:

x1, y1 - координаты первой точки,

x2, y2 - координаты второй точки.

Возвращаемое значение:

1 - точки совпадают,
0 - точки не совпадают.

ksGetCurveMinMaxParametr – Получить минимальный и максимальный параметр кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurveMinMaxParametr.

Синтаксис Automation:

long ksGetCurveMinMaxParametr (long curve,

double *tMin,

double *tMax);

Входные параметры:

curve - указатель на кривую.

Выходные параметры:

tMin - минимальный параметр кривой,
tMax - максимальный параметр кривой.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetCurvePerimeter – Получить периметр кривой

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_PERIMETER.htm`

Аналог данного метода при использовании API экспортных функций - ksGetCurvePerimeter.

Синтаксис Automation:

double ksGetCurvePerimeter (long curve,

short dimension);

Входные параметры:

curve - указатель на кривую,
dimension - размерность длины.

Размерности длины...

Возвращаемое значение:

- периметр кривой.

ksGetCurvePerpendicular – Получить угол нормали к кривой в заданной точке

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurvePerpendicular.

Синтаксис Automation:

double ksGetCurvePerpendicular (long curve,

double x,

double y);

Входные параметры:

curve - указатель на кривую,
x, y - координаты точки.

Возвращаемое значение:

- угол нормали к кривой в заданной точке (в градусах).

Примечание:

Если указанная точка не находится на кривой, она проецируется на кривую.

ksGetCurvePoint – Преобразовать параметр кривой t в координаты вида

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurvePoint.

Синтаксис Automation:

long ksGetCurvePoint (long curve,

double t,

double *x,

double *y);

Входные параметры:

curve - указатель на кривую,
t - параметр кривой.

Выходные параметры:

kx, ky - координаты проекции точки на кривую,

Возвращаемое значение:

1 - в случае удачного завершения.
0 - в случае неудачи.

Примечание:

См. также ksMathematic2D::ksGetCurvePointProjectionEx.

ksGetCurvePointProjection – Получить координаты проекции точки на кривую

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurvePointProjection.

Синтаксис Automation:

long ksGetCurvePointProjection (long curve,

double x,

double y,

double* kx,

double* ky);

Входные параметры:

curve - указатель на кривую,
x, y - координаты проецируемой точки.

Выходные параметры:

kx, ky - координаты проекции точки на кривую.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. См. также ksMathematic2D::ksGetCurvePointProjectionEx.
2. При выполнении метода ksGetCurvePointProjection проекция на кривую из данной точки может быть не найдена, например, если проекция находится на продолжении кривой. В таком случае возвращаются координаты ближайшей к проекции начальной или конечной точки кривой.
3. В метод ksGetCurvePointProjection нужно передавать координаты в системе координат кривой, то есть в системе координат макроэлемента, которому принадлежит кривая. Возвращаются координаты также в системе координат кривой.

ksGetCurvePointProjectionEx – Получить координаты проекции точки на кривую

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurvePointProjectionEx.

Синтаксис Automation:

long ksGetCurvePointProjectionEx (long curve,

double x, double y,

double *kx, double *ky,

double *t);

Входные параметры:

curve	- указатель на кривую,
x, y	- координаты проецируемой точки.

Выходные параметры:

kx, ky	- координаты проекции точки на кривую,
t	- параметр кривой.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. См. также ksMathematic2D::ksGetCurvePointProjection.
2. При выполнении метода ksGetCurvePointProjectionEx проекция на кривую из данной точки может быть не найдена, например, если проекция находится на продолжении кривой. В таком случае возвращаются координаты ближайшей к проекции начальной или конечной точки кривой.
3. В метод ksGetCurvePointProjectionEx нужно передавать координаты в системе координат кривой, то есть в системе координат макроэлемента, которому принадлежит кривая. Возвращаются координаты также в системе координат кривой.

ksLinePointTangentCurve – Получить прямую через точку перпендикулярно данной кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksLinePointTangentCurve.

Синтаксис Automation:

BOOL ksLinePointTangentCurve(long p, double x, double y, LPDISPATCH param);

Входные параметры:

p	- указатель на кривую,
x, y	- координаты точки.

Выходной параметр:

param	- указатель на интерфейс динамического массива ksDynamicArray углов касательных типа DOUBLE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

ksMovePointOnCurveEx – Переместить точку на расстояние len по кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMovePointOnCurveEx.

Синтаксис Automation:

BOOL ksMovePointOnCurveEx(long curve, double* x, double* y, double* t, double len, long dir, long ext);

Входные параметры:

curve	- указатель на кривую,
len	- расстояние, на которое нужно сместить точку,
dir	- направление продвижения точки: 1 - в направлении построения кривой -1 - в обратном направлении,
ext	- учитывать продолжение кривой.

Выходной параметр:

x, y	- координаты точки.

t - параметр кривой.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

- Если в функцию перелается указатель на параметр кривой t, он используется в качестве начального параметра кривой для сдвига.
- Если указатель не передается, начальный параметр кривой определяется по координатам x, y.

ksGetCurvePerpendicularByT – Функция возвращает угол нормали к кривой в заданной точке по параметру кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetCurvePerpendicularByT.

Синтаксис Automation:

double ksGetCurvePerpendicularByT(long curve, double t);

Входные параметры:

p - указатель на кривую,
t - параметр кривой.

Выходной параметр:

param - указатель на интерфейс динамического массива ksDynamicArray углов касательных типа DOUBLE_ARR.

Возвращаемое значение:

- угол нормали.

ksIntersectArcArc – Получить координаты точек пересечения двух дуг окружностей

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectArcArc.

Синтаксис Automation:

BOOL ksIntersectArcArc (double xac,

double yac,

double rada,

double fa1,

double fa2,

short directa,

double xbc,

double ybc,

double radb,

double fb1,

double fb2,

short directb,

LPDISPATCH param);

Входные параметры:

xac, yac	- координаты центра первой дуги,
rada	- радиус первой дуги,
fa1, fa2	- углы начальной и конечной точек первой дуги,
directa	- направление первой дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке,
xbc, ybc	- координаты центра второй дуги,
radb	- радиус второй дуги,
fb1, fb2	- углы начальной и конечной точек второй дуги,
directb	- направление второй дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectArcLin – Получить координаты точек пересечения дуги окружности и прямой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectArcLine.

Синтаксис Automation:

BOOL ksIntersectArcLin (double xc,

double yc,

double rad,

double f1,

double f2,

long n,

double x,

double y,

double ang,

LPDISPATCH param);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
f1, f2	- углы начальной и конечной точек дуги,
n	- направление дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке,
x, y	- координаты точки на прямой,
ang	- угол наклона прямой.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectCirArc – Получить координаты точек пересечения окружности и дуги

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectCirArc.

Синтаксис Automation:

BOOL ksIntersectCirArc (double xcc,

double ycc,

double radc,

double xac,

double yac,

double rada,

double fa1,

double fa2,

short directa,

LPDISPATCH param);

Входные параметры:

xcc, ycc	- координаты центра окружности,
radc	- радиус окружности,
xac, yac	- координаты центра дуги,
rada	- радиус дуги,
fa1, fa2	- углы начальной и конечной точек дуги,
directa	- направление дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectCirCir – Получить координаты точек пересечения двух окружностей

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectCirCir.

Синтаксис Automation:

BOOL ksIntersectCirCir (double xc1,

double yc1,

double radius1,

double xc2,

double yc2,

double radius2,

LPDISPATCH param);

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
radius1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
radius2	- радиус второй окружности,

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectCirLin – Получить координаты точек пересечения окружности и прямой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectCirLin.

Синтаксис Automation:

BOOL ksIntersectCirLin (double xc,

double yc,

double rad,

double xl,

double yl,

double angle,

LPDISPATCH param);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,

x1, y1 - координаты точки на прямой,
angle - угол наклона прямой.

Выходной параметр:

param - указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE - пересечение данных объектов обнаружено,
FALSE - пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectCurvCurv – Получить координаты точек пересечения двух кривых

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksIntersectCurvCurv.

Синтаксис Automation:

long ksIntersectCurvCurv (long p1,

long p2,

LPDISPATCH param);

Входные параметры:

p1 - указатель на первую кривую,
p2 - указатель на вторую кривую.

Выходной параметр:

param - указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

1 - пересечение данных объектов обнаружено,
0 - пересечение не обнаружено (кривые не пересекаются или совпадают),
-1 - первый объект не существует,
-2 - второй объект не существует,
-3 - кривые расположены в разных видах,
-4 - один (или оба) из указанных объектов - не геометрический (не отрезок, не окружность, а, например, линия-выноска),

-5	- первый объект не является кривой,
-6	- второй объект не является кривой,
-7	- ошибка: массив типа POINT_ARR предварительно не создан.

Смотрите также: ksMathPointParam

ksIntersectLinLin – Получить координаты точки пересечения двух прямых

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectLinLin.

Синтаксис Automation:

BOOL ksIntersectLinLin (double x1,

double y1,

double angle1,

double x2,

double y2,

double angle2,

LPDISPATCH param);

Входные параметры:

x1, y1	- координаты точки на первой прямой,
angle1	- угол наклона первой прямой,
x2, y2	- координаты точки на второй прямой,
angle2	- угол наклона второй прямой,

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectLinSArc – Получить координаты точек пересечения отрезка и дуги

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectLinSArc.

Синтаксис Automation:

BOOL ksIntersectLinSArc (double x1,

double y1,

double x2,

double y2,

double xc,

double yc,

double rad,

double f1,

double f2,

short direct,

LPDISPATCH param);

Входные параметры:

x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка,
xc, yc	- координаты центра дуги,
rad	- радиус дуги,
f1, f2	- углы начальной и конечной точек дуги,
direct	- направление дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectLinSCir – Получить координаты точек пересечения отрезка и окружности

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectLinSCir.

Синтаксис Automation:

BOOL ksIntersectLinSCir (double x1,

double y1,

double x2,

double y2,

double xc,

double yc,

double rad,

LPDISPATCH param);

Входные параметры:

x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка,
xc, yc	- координаты центра окружности,
rad	- радиус окружности.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных объектов обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectLinSLine – Получить координаты точки пересечения отрезка и прямой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectLinSLine.

Синтаксис Automation:

BOOL ksIntersectLinSLine (double x1,

double y1,

double x2,

double y2,

double x,

double y,

double ang,

LPDISPATCH param);

Входные параметры:

x1, y1	- координаты начальной точки отрезка,

x2, y2 - координаты конечной точки отрезка,
x, y - координаты точки на прямой,
ang - угол наклона прямой.

Выходной параметр:

param - указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE - пересечение данных объектов обнаружено,
FALSE - пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksIntersectLinSLinS– Получить координаты точки пересечения двух отрезков

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IntersectLinSLinS.

Синтаксис Automation:

BOOL ksIntersectLinSLinS (double x11,

double y11,

double x12,

double y12,

double x21,

double y21,

double x22,

double y22,

LPDISPATCH param);

Входные параметры:

x11, y11 - координаты начальной точки первого отрезка,
x12, y12 - координаты конечной точки первого отрезка,
x21, y21 - координаты начальной точки второго отрезка,
x22, y22 - координаты конечной точки второго отрезка.

Выходной параметр:

param - указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- пересечение данных отрезков обнаружено,
FALSE	- пересечение не обнаружено.

Смотрите также: ksMathPointParam

ksMovePointOnCurve – Получить координаты точки, находящейся на указанном вдоль кривой расстоянии от указанной точки

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMovePointOnCurve.

Синтаксис Automation:

long ksMovePointOnCurve (long curve,

double* x,

double* y,

double len,

long dir);

Входные параметры:

curve	- указатель на кривую,
len	- расстояние, на которое нужно сместить точку,
dir	- направление продвижения точки:
	1 - в направлении построения кривой,
	-1 - в обратном направлении.

Выходные параметры:

x, y	- координаты точки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если точка не находится на кривой, она проецируется на кривую.

ksPerpendicular – Получить координаты точки пересечения отрезка и перпендикуляра к нему, проходящего через заданную точку

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Perpendicular.

Синтаксис Automation:

BOOL ksPerpendicular (double x,

double y,

double x1,

double y1,

double x2,

double y2,

double* xp,

double* yp);

Входные параметры:

x, y	- координаты произвольной точки,
x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка.

Выходные параметры:

xp, yp	- координаты точки пересечения отрезка и перпендикуляра к нему, проходящего через заданную точку.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksPointsOnCurve – Получить указатель на интерфейс динамического массива точек, равномерно расположенных на кривой

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPointsOnCurve.

Синтаксис Automation:

LPDISPATCH ksPointsOnCurve (long curve,

long count);

Входные параметры:

curve - указатель на кривую,
count - количество точек.

Возвращаемое значение:

- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Смотрите также: ksMathPointParam

ksPointsOnCurveByStep – Получить массив точек, расположенных на кривой с заданным шагом

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksPointsOnCurveByStep.

Синтаксис Automation:

LPDISPATCH ksPointsOnCurveByStep (long curve,

double step);

Входные параметры:

curve - указатель на кривую,
step - шаг точек вдоль кривой.

Возвращаемое значение:

- указатель на динамический массив математических точек ksDynamicArray типа POINT_ARR.

ksRotate – Повернуть точку относительно центра

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Rotate.

Синтаксис Automation:

BOOL ksRotate (double x,

double y,

double xc,

double yc,

double ang,

double* xr,

double* yr);

Входные параметры:

x, y	- координаты базовой точки,
xc, yc	- координаты центра поворота,
ang	- угол поворота.

Выходные параметры:

xr, yr	- координаты точки после поворота.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksSinD – Получить синус аргумента

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm`

Аналог данного метода при использовании API экспортных функций - SinD.

Синтаксис Automation:

double ksSinD (double x);

Входной параметр:

x	- угол в градусах.

Возвращаемое значение:

- синус аргумента.

ksSymmetry – Получить координаты точки, симметричной относительно заданной оси

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Symmetry.

Синтаксис Automation:

BOOL ksSymmetry (double x,

double y,

double x1,

double y1,

double x2,

double y2,

double* xc,

double* yc);

Входные параметры:

x, y	- координаты базовой точки,
x1, y1	- координаты первой точки на оси симметрии,
x2, y2	- координаты второй точки на оси симметрии.

Выходные параметры:

xc, yc	- координаты симметричной точки.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

ksTanD – Получить тангенс аргумента

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данного метода при использовании API экспортных функций - TanD.

Синтаксис Automation:

double ksTanD (double x);

Входной параметр:

x	- угол в градусах.

Возвращаемое значение:

	- тангенс аргумента.

ksTanCircleCircle – Получить координаты точек касания прямых к двум окружностям

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TanCircleCircle.

Синтаксис Automation:

BOOL ksTanCircleCircle (double xc1,

double yc1,

double radius1,

double xc2,

double yc2,

double radius2,

LPDISPATCH param);

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
radius1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
radius2	- радиус второй окружности.

Выходной параметр:

param	- указатель на интерфейс массива координат точек касания ksTAN.

Возвращаемое значение:

TRUE	- точки касания обнаружены,
FALSE	- точки касания не обнаружены.

ksTanCurvCurv – Получить координаты точек касания прямых к двум кривым

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTanCurvCurv.

Синтаксис Automation:

long ksTanCurvCurv (long p1,
long p2,
LPDISPATCH pointArr1,
LPDISPATCH pointArr2);

Входные параметры:

p1	- указатель на первую кривую,
p2	- указатель на вторую кривую.

Выходные параметры:

pointArr1	- динамический массив ksDynamicArray математических точек POINT_ARR на кривой p1,

pointArr2 - динамический массив математических точек ksDynamicArray POINT_ARR на кривой p2.

Возвращаемое значение:

1 - успешное завершение,
0 - построить касательную нельзя (кривые совпадают или одна кривая вложена в другую),
-1 - первый объект не существует,
-2 - второй объект не существует,
-3 - кривые расположены в разных видах,
-4 - не совпадают СК определения кривых (геометрическая и аннотационная),
-5 - первый объект не является кривой,
-6 - второй объект не является кривой,
-7 - ошибка.

ksTanLineAngCircle – Получить точки касания окружности и прямой, проходящей под заданным углом

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TanLineAngCircle.

Синтаксис Automation:

BOOL ksTanLineAngCircle (double xc,

double yc,

double rad,

double ang,

LPDISPATCH param);

Входные параметры:

xc, yc - координаты центра окружности,
rad - радиус окружности,
ang - угол касательной прямой.

Выходной параметр:

param - указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE - точки касания данных объектов обнаружены,
FALSE - точки касания не обнаружены.

Смотрите также: ksMathPointParam

ksTanLinePointCircle – Получить точки касания окружности и прямой, проходящей через заданную точку

Интерфейс...

Аналог данного метода при использовании API экспортных функций - TanLinePointCircle.

Синтаксис Automation:

BOOL ksTanLinePointCircle (double x,

double y,

double xc,

double yc,

double rad,

LPDISPATCH param);

Входные параметры:

x, y	- координаты точки,
xc, yc	- координаты центра окружности,
rad	- радиус окружности.

Выходной параметр:

param	- указатель на интерфейс динамического массива параметров математической точки типа POINT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- точки касания данных объектов обнаружены,
FALSE	- точки касания не обнаружены.

Смотрите также: ksMathPointParam

ksTanLinePointCurve – Получить точки касания кривой и прямой, проведенной через заданную точку

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksTanLinePointCurve.

Синтаксис Automation:

BOOL ksTanLinePointCurve (double x,

double y,

long pCur,

LPDISPATCH param);

Входные параметры:

pCur	- указатель на кривую,
x, y	- координаты точки.

Выходной параметр:

param	- указатель на интерфейс динамического массива ksDynamicArray параметров математической точки типа POINT_ARR.

Возвращаемое значение:

TRUE	- точки касания обнаружены,
FALSE	- точки касания не обнаружены.

Смотрите также: ksMathPointParam

Интерфейсы параметров формата и компоновки чертежа

Интерфейс параметров ассоциативного вида (Интерфейс ksAssociationViewParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_ARBITRARY_VIEW.htm

Интерфейс параметров ассоциативного вида.

Аналог данного интерфейса при использовании API экспортных функций - AssociationViewParam.

Примечание:

1. Указатель на интерфейс должен быть получен с помощью метода KompasObject::GetParamStruct с параметром ko_AssociationViewParam.
2. Параметры могут быть получены с помощью метода ksDocument2D::ksGetObjParam с параметром ASSOCIATION_VIEW_PARAM.

ksAssociationViewParam – свойства

disassembly – Разнесение объекта

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- включено,
FALSE	- выключено.

Синтаксис Automation:

disassembly = iAssViewParam.disassembly	Получить свойство (*)
iAssViewParam.disassembly = disassembly	Установить свойство (*)
disassembly = iAssViewParam.GetDisassembly()	Получить свойство (**)
iAssViewParam. SetDisassembly(disassembly)	Установить свойство (**)

fileName – Имя файла документа трехмерного объекта

Интерфейс...

Тип данных:BSTR.

Синтаксис Automation:

fileName = iAssViewParam.fileName	Получить свойство (*)
iAssViewParam.fileName = fileName	Установить свойство (*)
fileName = iAssViewParam.GetFileName()	Получить свойство (**)
iAssViewParam.SetFileName(fileName)	Установить свойство (**)

hiddenLinesShow – Признак отрисовки невидимых линий

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- отрисовка включена,
FALSE	- отрисовка выключена.

Синтаксис Automation:

hiddenLinesShow = iAssViewParam.hiddenLinesShow	Получить свойство (*)
iAssViewParam.hiddenLinesShow = hiddenLinesShow	Установить свойство (*)
hiddenLinesShow = iAssViewParam.GetHiddenLinesShow()	Получить свойство (**)
iAssViewParam. SetHiddenLinesShow(hiddenLinesShow)	Установить свойство (**)

hiddenLinesStyle – Номер стиля отрисовки невидимых линий

Интерфейс...

Тип данных:long.

Значения свойства:

номер стиля	
0	- умолчательный стиль.

Синтаксис Automation:

visibleLinesStyle = iAssViewParam.visibleLinesStyle	Получить свойство (*)
iAssViewParam.visibleLinesStyle = visibleLinesStyle	Установить свойство (*)
visibleLinesStyle = iAssViewParam.GetVisibleLinesStyle()	Получить свойство (**)
iAssViewParam. SetVisibleLinesStyle(visibleLinesStyle)	Установить свойство (**)

projectionName – Имя проекции (из списка проекций в документе-источнике)

Интерфейс...

Тип данных:BSTR.

Синтаксис Automation:

projectionName = iAssViewParam.projectionName	Получить свойство (*)
iAssViewParam.projectionName = projectionName	Установить свойство (*)
projectionName = iAssViewParam.GetProjectionName()	Получить свойство (**)
iAssViewParam. SetProjectionName(projectionName)	Установить свойство (**)

projectionLink – Состояние проекционной связи

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- включена,
FALSE	- выключена.

Синтаксис Automation:

projectionLink = iAssViewParam.projectionLink	Получить свойство (*)
iAssViewParam.projectionLink = projectionLink	Установить свойство (*)
projectionLink = iAssViewParam.GetProjectionLink()	Получить свойство (**)

iAssViewParam. Установить свойство (**)
SetProjectionLink(projectionLink)

projBodies – Проецировать ли тела

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- проецирование включено,
FALSE	- проецирование выключено.

Синтаксис Automation:

projBodies = iAssViewParam.projBodies Получить свойство (*)
iAssViewParam.projBodies = projBodies Установить свойство (*)
projBodies = iAssViewParam.GetProjBodies() Получить свойство (**)
iAssViewParam. Установить свойство (**)
SetProjBodies(projBodies)

projSurfaces – Проецировать ли поверхности

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- проецирование включено,
FALSE	- проецирование выключено.

Синтаксис Automation:

projSurfaces = iAssViewParam.projSurfaces Получить свойство (*)
iAssViewParam.projSurfaces = projSurfaces Установить свойство (*)
projSurfaces = iAssViewParam.GetProjSurfaces() Получить свойство (**)
iAssViewParam. Установить свойство (**)
SetProjSurfaces(projSurfaces)

projThreads – Проецировать ли резьбы

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- проецирование включено,
FALSE	- проецирование выключено.

Синтаксис Automation:

```
projThreads = iAssViewParam.projThreads    Получить свойство (* )
iAssViewParam.projThreads = projThreads    Установить свойство (* )
projThreads =                              Получить свойство (**)
iAssViewParam.GetProjThreads()
iAssViewParam.                             Установить свойство (**)
SetProjThreads(projThreads)
```

sameHatch – Признак одинаковой штриховки всех деталей сборки

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- штриховка одинаковая,
FALSE	- штриховка разная.

Синтаксис Automation:

```
sameHatch = iAssViewParam.sameHatch      Получить свойство (* )
iAssViewParam.sameHatch = sameHatch      Установить свойство (* )
iAssViewParam.GetSameHatch()             Получить свойство (**)
iAssViewParam.                           Установить свойство (**)
SetSameHatch(sameHatch)
```

Примечание:

Свойство используется только для видов типа разрез или сечение.

section – Признак разрез/сечение

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- сечение,
FALSE	- разрез.

Синтаксис Automation:

```
section = iAssViewParam.section          Получить свойство (* )
iAssViewParam.section = section          Установить свойство (* )
section = iAssViewParam.GetSection()     Получить свойство (**)
iAssViewParam.SetSection(section)        Установить свойство (**)
```

Примечание:

Свойство используется только для видов типа разрез или сечение.

tangentEdgesShow – Признак отрисовки видимых линий перехода

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- отрисовка включена,
FALSE	- отрисовка выключена.

Синтаксис Automation:

tangentEdgesShow = iAssViewParam.tangentEdgesShow	Получить свойство (*)
iAssViewParam.tangentEdgesShow = tangentEdgesShow	Установить свойство (*)
tangentEdgesShow = iAssViewParam.GetTangentEdgesShow()	Получить свойство (**)
iAssViewParam. SetTangentEdgesShow(tangentEdgesShow)	Установить свойство (**)

tangentEdgesStyle – Номер стиля отрисовки видимых линий перехода

Интерфейс...

Тип данных:long.

Значения свойства:

номер стиля	
0	- умолчательный стиль.

Синтаксис Automation:

tangentEdgesStyle = iAssViewParam.tangentEdgesStyle	Получить свойство (*)
iAssViewParam.tangentEdgesStyle = tangentEdgesStyle	Установить свойство (*)
tangentEdgesStyle = iAssViewParam.GetTangentEdgesStyle()	Получить свойство (**)
iAssViewParam. SetTangentEdgesStyle(tangentEdgesStyle)	Установить свойство (**)

visibleLinesStyle – Номер стиля отрисовки видимых ребер и очерков

Интерфейс...

Тип данных:long.

Значения свойства:

номер стиля
0 - умолчательный стиль.

Синтаксис Automation:

hiddenLinesStyle = iAssViewParam.hiddenLinesStyle Получить свойство (*)
iAssViewParam.hiddenLinesStyle = hiddenLinesStyle Установить свойство (*)
hiddenLinesStyle = iAssViewParam.GetHiddenLinesStyle() Получить свойство (**)
iAssViewParam. Установить свойство (**)
SetHiddenLinesStyle(hiddenLinesStyle)

ksAssociationViewParam – методы

GetHatchParam – Получить указатель на интерфейс параметров штриховки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetHatchParam();

Возвращаемое значение:

- Указатель на интерфейс ksHatchParam.

Примечание:

Свойство используется только для видов типа разрез или сечение.

GetViewParam – Получить указатель на интерфейс параметров обычного вида

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetViewParam();

Возвращаемое значение:

- Указатель на интерфейс ksViewParam.

Примечание:

Свойство используется только для видов типа разрез или сечение.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

.

SetDimensionLayoutScaling – Установить признак масштабирования аннотационных объектов вида

Интерфейс...

Синтаксис Automation:

BOOL SetDimensionLayoutScaling(BOOL scaling);

Входной параметр:

scaling	TRUE	- масштабировать аннотационные объекты вида,
	FALSE	- не масштабировать объекты.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Используется только для изменения параметров вида ksDocument2D::ksSetObjParam.
2. Используется для всех видов, в том числе и для обычного (неассоциативного) вида.

viewType – Тип вида

Интерфейс...

Тип данных:short.

Значения свойства:

- Тип вида LtViewType.

Синтаксис Automation:

viewType = iAssociationView.viewType	Получить свойство (*)
viewType = iAssociationView.GetViewType()	Получить свойство (**)

Примечание:

Свойство доступно только для чтения.

Параметры стандартного листа (Интерфейс ksStandartSheet)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_FORMAT_SHEET.htm

Интерфейс параметров стандартного листа.

Аналог данных параметров при использовании API экспортных функций - StandartSheet.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksStandartSheet – свойства

format – Формат листа

Интерфейс...

Тип данных:short.

Стандартные форматы...

Синтаксис Automation:

format = iStandartSheet.format	Получить свойство (*)
iStandartSheet.format = format	Установить свойство (*)
format = iStandartSheet.GetFormat()	Получить свойство (**)
iStandartSheet.SetFormat (format)	Установить свойство (**)

multiply – Кратность формата

Интерфейс...

Тип данных:short.

Синтаксис Automation:

multiply = iStandartSheet.multiply	Получить свойство (*)
iStandartSheet.multiply = multiply	Установить свойство (*)
multiply = iStandartSheet.GetMultiply()	Получить свойство (**)
iStandartSheet.SetMultiply (multiply)	Установить свойство (**)

ksStandartSheet – методы

direct – Расположение основной надписи

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE	- вдоль короткой стороны,
TRUE	- вдоль длинной стороны.

Синтаксис Automation:

direct = iStandartSheet.direct	Получить свойство (*)
iStandartSheet.direct = direct	Установить свойство (*)
direct = iStandartSheet.GetDirect()	Получить свойство (**)
iStandartSheet.SetDirect (direct)	Установить свойство (**)

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры стандартного листа.

Размеры листа (Интерфейс ksSheetSize)

Справка системы КОМПАС...

KOMPAS.chm::/362_41_1_Listy.htm

Интерфейс параметров размеров листа.

Аналог данных параметров при использовании API экспортных функций - SheetSize.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksSheetSize – свойства

height – Высота листа

Интерфейс...

Тип данных:double.

Синтаксис Automation:

height = iSheetSize.height	Получить свойство (*)

iSheetSize.height = height	Установить свойство (*)
height = iSheetSize.GetHeight()	Получить свойство (**)
iSheetSize.SetHeight(height)	Установить свойство (**)

width – Ширина листа

Интерфейс...

Тип данных:double.

Синтаксис Automation:

width = iSheetSize.width	Получить свойство (*)
iSheetSize.width = width	Установить свойство (*)
width = iSheetSize.GetWidth()	Получить свойство (**)
iSheetSize.SetWidth(width)	Установить свойство (**)

ksSheetSize – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры размеров листа.

Оформление (Интерфейс ksSheetPar)

Справка системы КОМПАС...

`KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm`

Интерфейс параметров оформления.

Аналог данных параметров при использовании API экспортных функций - SheetPar.

Примечания:

1. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.
2. Стандартный формат или нестандартный, определяется значением параметра type из параметров документа ksDocumentParam.

   Смотрите также: KompasObject

ksSheetPar – свойства

layoutName – Для чертежа – имя библиотеки оформления, для спецификации – имя библиотеки стилей спецификации

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

layoutName = iSheetPar.layoutName	Получить свойство (*)
iSheetPar.layoutName = layoutName	Установить свойство (*)
layoutName = iSheetPar.GetLayoutName()	Получить свойство (**)
iSheetPar.SetLayoutName(layoutName)	Установить свойство (**)

Примечание:

Если строка пустая, берется библиотека Graphic.lyt.

shtType – Для чертежа – тип штампа из указанной библиотеки, для спецификации – номер стиля из указанной библиотеки

Интерфейс...

Тип данных:short.

Синтаксис Automation:

shtType = iSheetPar.shtType	Получить свойство (*)
iSheetPar.shtType = shtType	Установить свойство (*)
shtType = iSheetPar.GetShtType()	Получить свойство (**)
iSheetPar.SetShtType(shtType)	Установить свойство (**)

ksSheetPar – методы

GetSheetParam – Получить указатель на интерфейс параметров листа документа ksStandartSheet или ksSheetSize

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSheetParam();

Возвращаемое значение:

- указатель на интерфейс параметров листа документа ksStandartSheet (для стандартного листа) или ksSheetSize (для нестандартного листа).

Примечание:

Тип возвращаемого интерфейса определяется значением параметра type из параметров документа ksDocumentParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры оформления документа.

Оформление (Интерфейс ksSheetOptions)

Справка системы КОМПАС...

KOMPAS.chm::/1132_133_1_Obshchie_svedenija.htm

Интерфейс параметров оформления листа документа.

Аналог данных параметров при использовании API экспортных функций - SheetOptions.

Примечание:

1. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.
2. Данный интерфейс используется при передаче данных методом ksGetDocOptions.

ksSheetOptions – свойства

layoutName – Имя библиотеки оформления

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

layoutName = iSheetOptions.layoutName	Получить свойство (*)
iSheetOptions.layoutName = layoutName	Установить свойство (*)
layoutName = iSheetOptions.GetLayoutName()	Получить свойство (**)
iSheetOptions.SetLayoutName(layoutName)	Установить свойство (**)

sheetType – Тип листа

Интерфейс...

Тип данных:BOOL.

Синтаксис Automation:

sheetType = iSheetOptions.sheetType	Получить свойство (*)
iSheetOptions.sheetType = sheetType	Установить свойство (*)
sheetType = iSheetOptions.GetSheetType()	Получить свойство (**)
iSheetOptions.SetSheetType (sheetType)	Установить свойство (**)

Значения свойства:

TRUE	- пользовательский формат,
FALSE	- стандартный лист.

Примечание:

Свойство необходимо учитывать при вызове метода GetSheetParam. Значение свойства должно соответствовать значению входного параметра type. Осталась для совместимости со старыми библиотеками.

shtType – Для чертежа – тип основной надписи, для спецификации – имя библиотеки стилей спецификации

Интерфейс...

Тип данных:short.

Синтаксис Automation:

shtType = iSheetOptions.shtType	Получить свойство (*)
iSheetOptions.shtType = shtType	Установить свойство (*)
shtType = iSheetOptions.GetShtType()	Получить свойство (**)
iSheetOptions.SetShtType(shtType)	Установить свойство (**)

ksSheetOptions – методы

GetSheetParam – Получить указатель на интерфейс параметров листа документа

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSheetParam (BOOL type);

Входной параметр:

type	- признак формата листа: FALSE - стандартный лист, TRUE - пользовательский формат.

Возвращаемое значение:

- указатель на интерфейс параметров листа ksStandartSheet или ksSheetSize.

Примечание:

При вызове метода необходимо учитывать свойство sheetType. Его значение должно соответствовать значению входного параметра type. Осталась для совместимости со старыми библиотеками.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

МЦХ плоской фигуры (Интерфейс ksInertiaParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_MIX.htm

Интерфейс параметров для расчета МЦХ плоской фигуры.

Аналог данных параметров при использовании API экспортных функций - InertiaParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksInertiaParam – свойства

A – Угол между первой главной осью и осью x

Интерфейс...

Тип данных:double.

Синтаксис Automation:

A = iInertiaParam.A Получить свойство (*)
A = iInertiaParam.GetA() Получить свойство (**)

Примечание:

Свойство только для чтения.

F – Площадь

Интерфейс...

Тип данных:double.

Синтаксис Automation:

F = iInertiaParam.F Получить свойство (*)
F = iInertiaParam.GetF() Получить свойство (**)

Примечание:

Свойство только для чтения.

Ix, Iy – Моменты инерции относительно осей x и y

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
Ix = iInertiaParam.Ix                Получить свойство (* )
Ix = iInertiaParam.GetLx()           Получить свойство (**)
```

Примечание:

Свойства только для чтения.

Ixıy – Центробежный момент инерции относительно исходных осей x и y

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
Ixy = iInertiaParam.Ixy              Получить свойство (* )
Ixy = iInertiaParam.GetLxy()         Получить свойство (**)
```

Примечание:

Свойство только для чтения.

jıx, jy – Главные центральные моменты инерции

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
jx = iInertiaParam.jx                Получить свойство (* )
jx = iInertiaParam.GetJx()           Получить свойство (**)
```

Примечание:

Свойство только для чтения.

mx, my – Моменты инерции относительно осей, параллельных осям x и y и проходящих через центр тяжести

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
mx = iInertiaParam.mx                Получить свойство (* )
mx = iInertiaParam.GetMx()           Получить свойство (**)
```

Примечание:

Свойства только для чтения.

mxy – Центробежный момент инерции относительно осей, параллельных осям x и y

Интерфейс...

Тип данных:double.

Синтаксис Automation:

mxy = iInertiaParam.mxy	Получить свойство (*)
mxy = iInertiaParam.GetMxy()	Получить свойство (**)

Примечание:

Свойство только для чтения.

xc, yc – Координаты центра тяжести

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iInertiaParam.xc	Получить свойство (*)
xc = iInertiaParam.GetXc()	Получить свойство (**)

Примечание:

Свойства только для чтения.

Интерфейсы параметров стилей объектов

Стиль линии

Стиль линии (Интерфейс ksCurveStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1093_130_1_Nastrojka_stilja_lin.htm

Интерфейс параметров стиля кривой.

Аналог данных параметров при использовании API экспортных функций - CurveStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также

KompasObject

ksCurveStyleParam – свойства

color – Цвет

Интерфейс...

Тип данных:long.

Синтаксис Automation:

color = iCurveStyleParam.color	Получить свойство (*)
iCurveStyleParam.color = color	Установить свойство (*)
color = iCurveStyleParam.GetColor()	Получить свойство (**)
iCurveStyleParam.SetColor(color)	Установить свойство (**)

curveType – Тип кривой

Интерфейс...

Тип данных:short.

Значения свойства:

0	- сплошная,
1	- прерывистая,
2	- прерывистая, содержащая "картинки".

Синтаксис Automation:

curveType = iCurveStyleParam.curveType	Получить свойство (*)
iCurveStyleParam.curveType = curveType	Установить свойство (*)
curveType = iCurveStyleParam.GetCurveType()	Получить свойство (**)
iCurveStyleParam.SetCurveType(curveType)	Установить свойство (**)

Смотрите также: ksCurveStyleParam::widthPen

even – Тип окончания прерывистой кривой

Интерфейс...

Тип данных:short.

Значения свойства:

1	- кривая всегда оканчивается штрихами,
0	- кривая оканчивается "как получится".

Синтаксис Automation:

even = iCurveStyleParam.even	Получить свойство (*)
iCurveStyleParam.even = even	Установить свойство (*)
even = iCurveStyleParam.GetEven()	Получить свойство (**)
iCurveStyleParam.SetEven(even)	Установить свойство (**)

name – Имя стиля

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iCurveStyleParam.name	Получить свойство (*)
iCurveStyleParam.name = name	Установить свойство (*)
name = iCurveStyleParam.GetName()	Получить свойство (**)
iCurveStyleParam.SetName(name)	Установить свойство (**)

paperWidth – Толщина линии на бумаге

Интерфейс...

Тип данных:double.

Синтаксис Automation:

paperWidth = iCurveStyleParam.paperWidth	Получить свойство (*)
iCurveStyleParam.paperWidth = paperWidth	Установить свойство (*)
paperWidth = iCurveStyleParam.GetPaperWidth()	Получить свойство (**)
iCurveStyleParam.SetPaperWidth(paperWidth)	Установить свойство (**)

Примечание:

Толщина линии, заданная этим свойством, используется, если свойство widthPen установлено в пользовательский тип толщины линии.

При инициализации параметров интерфейса методом Init также устанавливается пользовательский способ задания толщины линии.

screenWidth –Толщина линии на экране

Интерфейс...

Тип данных:double.

Синтаксис Automation:

screenWidth = iCurveStyleParam.screenWidth	Получить свойство (*)
iCurveStyleParam.screenWidth = screenWidth	Установить свойство (*)
screenWidth = iCurveStyleParam.GetScreenWidth()	Получить свойство (**)
iCurveStyleParam.SetScreenWidth(screenWidth)	Установить свойство (**)

Примечание:

Толщина линии, заданная этим свойством, используется, если свойство widthPen установлено в пользовательский тип толщины линии.

При инициализации параметров интерфейса методом Init также устанавливается пользовательский способ задания толщины линии.

widthPen – Способ задания толщины линии

Интерфейс...

Тип данных:short.

Значения свойства:

0	- толщина задается пользователем,
1	- толщина как у основной линии,
2	- толщина как у тонкой линии,
3	- толщина как у утолщенной линии.

Синтаксис Automation:

widthPen = iCurveStyleParam. widthPen	Получить свойство (*)
iCurveStyleParam. widthPen = widthPen	Установить свойство (*)
widthPen = iCurveStyleParam.GetWidthPen ()	Получить свойство (**)
iCurveStyleParam.SetWidthPen(widthPen)	Установить свойство (**)

Примечание:

Если выбрано задание толщины пользователем, то используются значения толщины линии, заданные свойствами paperWidth и screenWidth.

ksCurveStyleParam – методы

GetPPattern – Получить (для прерывистой кривой) указатель на динамический массив параметров участков штриховой кривой ksDynamicArray типа CURVE_PATTERN_ARR или CURVE_PATTERN_ARR_EX

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPPattern (long type);

Возвращаемое значение:

- указатель на динамический массив ksDynamicArray типа CURVE_PATTERN_ARR или CURVE_PATTERN_ARR_EX.

Примечание:

Типу массива CURVE_PATTERN_ARR соответствует интерфейс ksCurvePattern.

Типу массива CURVE_PATTERN_ARR_EX соответствует интерфейс ksCurvePatternEx.

Смотрите также :ksCurveStyleParam::curveType

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pattern типа CURVE_PATTERN_ARR.

SetPPattern – Установить (для прерывистой кривой) указатель на динамический массив параметров участков штриховой кривой ksDynamicArray типа CURVE_PATTERN_ARR или CURVE_PATTERN_ARR_EX

Интерфейс...

Синтаксис Automation:

BOOL SetPPattern (LPDISPATCH pPattern);

Входной параметр:

pPattern - указатель на динамический массив ksDynamicArray типа CURVE_PATTERN_ARR или CURVE_PATTERN_ARR_EX.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Типу массива CURVE_PATTERN_ARR соответствует интерфейс ksCurvePattern.

Типу массива CURVE_PATTERN_ARR_EX соответствует интерфейс ksCurvePatternEx.

Смотрите также: ksCurveStyleParam::curveType

Участок штриховой линии (Интерфейс ksCurvePattern)

Справка системы КОМПАС...

KOMPAS.chm::/1093_130_1_Nastrojka_stilja_lin.htm

Интерфейс параметров участка штриховой кривой.

Аналог данных параметров при использовании API экспортных функций - CurvePattern.

Примечание:

Указатель на интерфейс можно получить при помощи метода

KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCurvePattern – свойства

inviisibleSeg – Длина невидимого участка линии

Интерфейс..

Тип данных:double.

Синтаксис Automation:

invisibleSeg = iCurvePattern.invisibleSeg	Получить свойство (*)
iCurvePattern.invisibleSeg = invisibleSeg	Установить свойство (*)
invisibleSeg = iCurvePattern.GetInvisibleSeg()	Получить свойство (**)
iCurvePattern.SetInvisibleSeg(invisibleSeg)	Установить свойство (**)

visibleSeg – Длина видимого участка линии

Интерфейс..

Тип данных:double.

Синтаксис Automation:

visibleSeg = iCurvePattern.visibleSeg	Получить свойство (*)
iCurvePattern.visibleSeg = visibleSeg	Установить свойство (*)
visibleSeg = iCurvePattern.GetVisibleSeg()	Получить свойство (**)
iCurvePattern.SetVisibleSeg(visibleSeg)	Установить свойство (**)

ksCurvePattern – методы

Init – Инициализировать параметры

Интерфейс..

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры участка линии.

“Картинка”, входящая в стиль линии (Интерфейс ksCurvePicture)

Справка системы КОМПАС...

KOMPAS.chm::/1093_130_1_Nastrojka_stilja_lin.htm

Интерфейс параметров "картинки", включаемой в стиль линии.

Аналог данных параметров при использовании API экспортных функций - CurvePicture.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct .

Смотрите также: KompasObject

ksCurvePicture – методы

GetPolygon – Получить указатель на динамический массив ломаных линий, описывающий "картинку" (точки в координатах листа относительно 0 картинки) типа POLYLINE_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPolygon();

Возвращаемое значение:

- указатель на динамический массив типа POLYLINE_ARR ksDynamicArray.

Смотрите также: ksMathPointParam

GetFill – Получить указатель на динамический массив, описывающий границу заливки типа POLYLINE_ARR (точки в координатах листа относительно 0 картинки)

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFill();

Возвращаемое значение:

- указатель на динамический массив типа POLYLINE_ARR ksDynamicArray.

Смотрите также: ksMathPointParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив polygon типа POLYLINE_ARR.
3. Создается динамический массив fill типа POLYLINE_ARR.

SetFill – Установить указатель на динамический массив, описывающий границу заливки типа POLYLINE_ARR (точки в координатах листа относительно 0 картинки)

Интерфейс...

Синтаксис Automation:

BOOL SetFill (LPDISPATCH fill);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входной параметр:

fill	- указатель на динамический массив типа POLYLINE_ARR ksDynamicArray.

Смотрите также: ksMathPointParam

SetPolygon – Установить указатель на динамический массив ломаных линий, описывающий "картинку" (точки в координатах листа относительно 0 картинки) типа POLYLINE_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetPolygon (LPDISPATCH polygon);

Входной параметр:

polygon	- указатель на динамический массив типа POLYLINE_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также: ksMathPointParam

Стиль линии с "картинкой" (Интерфейс ksCurvePatternEx)

Справка системы КОМПАС...

KOMPAS.chm::/1093_130_1_Nastrojka_stilja_lin.htm

Интерфейс параметров участка "расширенной" штриховой кривой.

Аналог данных параметров при использовании API экспортных функций - CurvePatternEx.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCurvePatternEx – свойства

dx, dy – Положение базовой точки (нуля) "картинки" относительно начала образца

Интерфейс...

Тип данных:double.

Синтаксис Automation:

dx = iCurvePatternEx.dx	Получить свойство (*)
iCurvePatternEx.dx = dx	Установить свойство (*)
dx = iCurvePatternEx.GetDx()	Получить свойство (**)
iCurvePatternEx.SetDx(dx)	Установить свойство (**)

frwName – Имя фрагмента-источника картинки

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

```
frwName = iCurvePatternEx.frwName                    Получить свойство (* )
iCurvePatternEx.frwName = frwName                    Установить свойство (* )
frwName = iCurvePatternEx.GetFrwName()               Получить свойство (**)
iCurvePatternEx.SetFrwName(frwName)                  Установить свойство (**)
```

Примечание

Картинка в виде фрагмента используется только в методе AddStyle (при создании стиля).

Смотрите также: ksCurvePatternEx::pictureType

invisibleSeg – Длина невидимого участка линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
invisibleSeg = iCurvePatternEx.invisibleSeg          Получить свойство (* )
iCurvePatternEx.invisibleSeg = invisibleSeg          Установить свойство (* )
invisibleSeg = iCurvePatternEx.GetInvisibleSeg()     Получить свойство (**)
iCurvePatternEx.SetInvisibleSeg( invisibleSeg )      Установить свойство (**)
```

pictureType – Тип картинки

Интерфейс...

Тип данных:short.

Значения свойства:

0 - картинка в виде массивов ломаных линий (доступ к массиву с помощью методов ksCurvePatternEx::GetCurvePicture и ksCurvePatternEx::SetCurvePicture),

1 - картинка в виде фрагмента (имя фрагмента-источника задается свойством ksCurvePatternEx::frwName).

Синтаксис Automation:

```
pictureType = iCurvePatternEx.pictureType            Получить свойство (* )
iCurvePatternEx.pictureType = pictureType            Установить свойство (* )
pictureType = iCurvePatternEx.GetPictureType()       Получить свойство (**)
iCurvePatternEx.SetPictureType( pictureType )        Установить свойство (**)
```

visibleSeg – Длина видимого участка линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

visibleSeg = iCurvePatternEx.visibleSeg	Получить свойство (*)
iCurvePatternEx.visibleSeg = visibleSeg	Установить свойство (*)
visibleSeg = iCurvePatternEx.GetVisibleSeg()	Получить свойство (**)
iCurvePatternEx.SetVisibleSeg(visibleSeg)	Установить свойство (**)

ksCurvePatternEx – методы

GetCurvePicture – Получить указатель на интерфейс картинки в виде массивов полилиний ksCurvePicture

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetCurvePicture();

Возвращаемое значение:

- указатель на интерфейс ksCurvePicture

Смотрите также: ksCurvePatternEx::pictureType

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

SetCurvePicture – Установить указатель на интерфейс картинки в виде массивов полилиний ksCurvePicture

Интерфейс...

Синтаксис Automation:

BOOL SetCurvePicture (LPDISPATCH picture);

Входной параметр:

picture - указатель на интерфейс ksCurvePicture

Возвращаемое значение:

TRUE - в случае удачного завершения.

Смотрите также

ksCurvePatternEx::pictureType

Стиль штриховки

Линия штриховки (Интерфейс ksHatchLineParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Интерфейс параметров линии штриховки.

Аналог данного интерфейса параметров при использовании API экспортных функций - структура HatchLineParam .

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksHatchLineParam – свойства

ang – Угол наклона линии (в градусах)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Тип данных:double.

Синтаксис Automation:

ang = iHatchLineParam.ang	Получить свойство (*)
iHatchLineParam.ang = ang	Установить свойство (*)
ang = iHatchLineParam.GetAng()	Получить свойство (**)
iHatchLineParam.SetAng(ang)	Установить свойство (**)

dx, dy – Смещение следующей линии

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Тип данных:double.

Синтаксис Automation:

dx = iHatchLineParam.dx	Получить свойство (*)
iHatchLineParam.dx = dx	Установить свойство (*)
dx = iHatchLineParam.GetDx()	Получить свойство (**)
iHatchLineParam.SetDx(dx)	Установить свойство (**)

style – Стиль линии

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Тип данных:short.

Системные стили линий...

Синтаксис Automation:

style = iHatchLineParam.style	Получить свойство (*)
iHatchLineParam.style = style	Установить свойство (*)
style = iHatchLineParam.GetStyle()	Получить свойство (**)
iHatchLineParam.SetStyle(style)	Установить свойство (**)

typeCurvStyle – Признак стиля линии

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Тип данных:short.

Значения свойства:

1	- пользовательский,
0	- системный.

Синтаксис Automation:

typeCurvStyle = iHatchLineParam.typeCurvStyle	Получить свойство (*)
iHatchLineParam.typeCurvStyle = typeCurvStyle	Установить свойство (*)
typeCurvStyle = iHatchLineParam.GetTypeCurvStyle()	Получить свойство (**)

iHatchLineParam.SetTypeCurvStyle(typeCurvStyle) Установить свойство (**)

Смотрите также: ksHatchLineParam::style

x, y – Координаты точки привязки

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Тип данных:double.

Синтаксис Automation:

x = iHatchLineParam.x	Получить свойство (*)
iHatchLineParam.x = x	Установить свойство (*)
x = iHatchLineParam.GetX()	Получить свойство (**)
iHatchLineParam.SetX(x)	Установить свойство (**)

ksHatchLineParam – методы

GetCurPar – Получить указатель на интерфейс пользовательского стиля линии

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Синтаксис Automation:

LPDISPATCH GetCurPar();

Возвращаемое значение:

- указатель на интерфейс пользовательского стиля линии ksCurveStyleParam.

Init – Инициализировать параметры

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры линии штриховки.

SetCurPar – Установить указатель на интерфейс пользовательского стиля линии

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_CHOICE_HATCH_LINE.htm

Синтаксис Automation:

BOOL SetCurPar (LPDISPATCH curPar);

Входной параметр:

curPar - указатель на интерфейс пользовательского стиля линии ksCurveStyleParam.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Стиль штриховки (Интерфейс ksHatchStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1104_131_1_Nastrojka_stilja_sht.htm

Интерфейс параметров стиля штриховки.

Аналог данных параметров при использовании API экспортных функций - HatchStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksHatchStyleParam – свойства

ang – Угол наклона линий штриховки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iHatchStyleParam.ang Получить свойство (*)

iHatchStyleParam.ang = ang	Установить свойство (*)
ang = iHatchStyleParam.GetAng()	Получить свойство (**)
iHatchStyleParam.SetAng(ang)	Установить свойство (**)

color – Цвет штриховки

Интерфейс...

Тип данных:long.

Синтаксис Automation:

color = iHatchStyleParam.color	Получить свойство (*)
iHatchStyleParam.color = color	Установить свойство (*)
color = iHatchStyleParam.GetColor()	Получить свойство (**)
iHatchStyleParam.SetColor(color)	Установить свойство (**)

isScalable – Признак изменения расстояния между линиями штриховки

Интерфейс...

Тип данных:short.

Значения свойства:

1	- по масштабу,
0	- по шагу.

Синтаксис Automation:

isScalable = iHatchStyleParam.isScalable	Получить свойство (*)
iHatchStyleParam.isScalable = isScalable	Установить свойство (*)
isScalable = iHatchStyleParam.GetIsScalable()	Получить свойство (**)
iHatchStyleParam.SetIsScalable(isScalable)	Установить свойство (**)

Примечание:

Шаг используется, если штриховка состоит из сплошных линий (например, металл, неметалл). Масштаб используется, если штриховка состоит из прерывистых линий (например, жидкость); в этом случае изменяется не только расстояние между линиями, но и размер штрихов и расстояние между ними.

mayChangeAngle – Признак, показывающий, разрешено ли менять угол наклона штриховки

Интерфейс...

Тип данных:short.

Значения свойства:

1 - угол наклона менять можно,
0 - угол наклона менять нельзя.

Синтаксис Automation:

mayChangeAngle = iHatchStyleParam.mayChangeAngle	Получить свойство (*)
iHatchStyleParam.mayChangeAngle = mayChangeAngle	Установить свойство (*)
mayChangeAngle = iHatchStyleParam.GetMayChangeAngle()	Получить свойство (**)
iHatchStyleParam.SetMayChangeAngle(mayChangeAngle)	Установить свойство (**)

mayChangeSpace – Признак, показывающий, разрешено ли менять шаг (масштаб) штриховки

Интерфейс...

Тип данных:short.

Значения свойства:

1 - шаг (масштаб) менять можно,
0 - шаг (масштаб) менять нельзя.

Синтаксис Automation:

mayChangeSpace = iHatchStyleParam.mayChangeSpace	Получить свойство (*)
iHatchStyleParam.mayChangeSpace = mayChangeSpace	Установить свойство (*)
mayChangeSpace = iHatchStyleParam.GetMayChangeSpace()	Получить свойство (**)
iHatchStyleParam.SetMayChangeSpace(mayChangeSpace)	Установить свойство (**)

mayChangeWidth – Признак, показывающий, разрешено ли менять ширину полосы штриховки

Интерфейс...

Тип данных:short.

Значения свойства:

1 - ширину полосы штриховки менять можно,
0 - ширину полосы штриховки менять нельзя.

Синтаксис Automation:

mayChangeWidth = iHatchStyleParam. mayChangeWidth	Получить свойство (*)
iHatchStyleParam.mayChangeWidth = mayChangeWidth	Установить свойство (*)
mayChangeWidth = iHatchStyleParam.GetMayChangeWidth()	Получить свойство (**)
iHatchStyleParam.SetMayChangeWidth(mayChangeWidth)	Установить свойство (**)

name – Имя стиля штриховки

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iHatchStyleParam.name	Получить свойство (*)
iHatchStyleParam.name = name	Установить свойство (*)
name = iHatchStyleParam.GetName()	Получить свойство (**)
iHatchStyleParam.SetName(name)	Установить свойство (**)

step – Шаг штриховки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

step = iHatchStyleParam.step	Получить свойство (*)
iHatchStyleParam.step = step	Установить свойство (*)
step = iHatchStyleParam.GetStep()	Получить свойство (**)
iHatchStyleParam.SetStep(step)	Установить свойство (**)

width – Ширина полосы штриховки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

width = iHatchStyleParam.width	Получить свойство (*)
iHatchStyleParam.width = width	Установить свойство (*)
width = iHatchStyleParam.GetWidth()	Получить свойство (**)
iHatchStyleParam.SetWidth(width)	Установить свойство (**)

ksHatchStyleParam – методы

GetArrLineParam – Получить указатель на интерфейс динамического массива структур параметров линий штриховки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrLineParam();

Возвращаемое значение:

- указатель на динамический массив ksDynamicArray типа HATCHLINE_ARR.

GetRefPoint – Получить указатель на интерфейс базовой точки

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetRefPoint();

Возвращаемое значение:

- указатель на интерфейс базовой точки ksMathPointParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры стиля штриховки.

SetArrLineParam – Установить указатель на интерфейс динамического массива структур параметров линий штриховки

Интерфейс...

Синтаксис Automation:

BOOL SetArrLineParam (LPDISPATCH arrLineParam);

Входной параметр:

arrLineParam - указатель на динамический массив ksDynamicArray типа HATCHLINE_ARR.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetRefPoint – Установить указатель на интерфейс базовой точки

Интерфейс...

Синтаксис Automation:

BOOL SetRefPoint (LPDISPATCH refPoint);

Входной параметр:

refPoint - указатель на интерфейс базовой точки ksMathPointParam.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Стиль текста (Интерфейс ksTextStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_TEXTSTYLE.htm

Интерфейс параметров стиля текста.

Аналог данных параметров при использовании API экспортных функций - TextStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи методов:

- KompasObject::GetParamStruct,
- ksSpcTuningStyleParam::GetSectionTextStyleFirst ,
- ksSpcTuningStyleParam::GetSectionTextStyleNext,
- ksSpcTuningStyleParam::GetObjectTextStyle.

Смотрите также: KompasObject

ksTextStyleParam – свойства

align – Выравнивание абзаца

Интерфейс...

Тип данных:short.

Значения свойства:

0	- влево,
1	- по центру,
2	- вправо,
3	- по ширине.

Синтаксис Automation:

align = iTextStyleParam.align	Получить свойство (*)
iTextStyleParam.align = align	Установить свойство (*)
align = iTextStyleParam.GetAlign()	Получить свойство (**)
iTextStyleParam.SetAlign(align)	Установить свойство (**)

bold – Толщина символов

Интерфейс...

Тип данных:short.

Значения свойства:

0	- нормальная,
1	- утолщенные символы.

Синтаксис Automation:

bold = iTextStyleParam.bold	Получить свойство (*)
iTextStyleParam.bold = bold	Установить свойство (*)
bold = iTextStyleParam.GetBold()	Получить свойство (**)
iTextStyleParam.SetBold(bold)	Установить свойство (**)

color – Цвет шрифта

Интерфейс...

Тип данных:long.

Синтаксис Automation:

color = iTextStyleParam.color	Получить свойство (*)
iTextStyleParam.color = color	Установить свойство (*)
color = iTextStyleParam.GetColor()	Получить свойство (**)
iTextStyleParam.SetColor(color)	Установить свойство (**)

fontName – Имя шрифта

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

```
fontName = iTextStyleParam.fontName                 Получить свойство (* )
iTextStyleParam.fontName = fontName                 Установить свойство (* )
fontName = iTextStyleParam.GetFontName()            Получить свойство (**)
iTextStyleParam.SetFontName( fontName )             Установить свойство (**)
```

height – Высота шрифта текста

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
height = iTextStyleParam.height                     Получить свойство (* )
iTextStyleParam.height = height                     Установить свойство (* )
height = iTextStyleParam.GetHeight()                Получить свойство (**)
iTextStyleParam.SetHeight( height )                 Установить свойство (**)
```

italic – Начертание символов

Интерфейс...

Тип данных:short.

Значения свойства:

```
0                   - прямое,
1                   - наклонное.
```

Синтаксис Automation:

```
italic = iTextStyleParam.italic                     Получить свойство (* )
iTextStyleParam.italic = italic                     Установить свойство (* )
italic = iTextStyleParam.GetItalic()                Получить свойство (**)
iTextStyleParam.SetItalic( italic )                 Установить свойство (**)
```

ksu – Коэффициент сужения символов

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
ksu = iTextStyleParam.ksu                           Получить свойство (* )
iTextStyleParam.ksu = ksu                           Установить свойство (* )
ksu = iTextStyleParam.GetKsu()                      Получить свойство (**)
iTextStyleParam.SetKsu( ksu )                       Установить свойство (**)
```

leftEdge – Отступ текста слева

Интерфейс...

Тип данных:double.

Синтаксис Automation:

leftEdge = iTextStyleParam.leftEdge	Получить свойство (*)
iTextStyleParam.leftEdge = leftEdge	Установить свойство (*)
leftEdge = iTextStyleParam.GetLeftEdge()	Получить свойство (**)
iTextStyleParam.SetLeftEdge(leftEdge)	Установить свойство (**)

name – Имя стиля

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iTextStyleParam.name	Получить свойство (*)
iTextStyleParam.name = name	Установить свойство (*)
name = iTextStyleParam.GetName()	Получить свойство (**)
iTextStyleParam.SetName(name)	Установить свойство (**)

posKS – Отступ красной строки (в миллиметрах)

Интерфейс...

Тип данных:double.

Синтаксис Automation:

posKS = iTextStyleParam.posKS	Получить свойство (*)
iTextStyleParam.posKS = posKS	Установить свойство (*)
posKS = iTextStyleParam.GetPosKS()	Получить свойство (**)
iTextStyleParam.SetPosKS(posKS)	Установить свойство (**)

rightEdge – Отступ текста справа

Интерфейс...

Синтаксис Automation:

rightEdge = iTextStyleParam.rightEdge	Получить свойство (*)
iTextStyleParam.rightEdge = rightEdge	Установить свойство (*)
rightEdge = iTextStyleParam.GetRightEdge()	Получить свойство (**)
iTextStyleParam.SetRightEdge(rightEdge)	Установить свойство (**)

step – Шаг строк

Интерфейс...

Тип данных:double.

Синтаксис Automation:

step = iTextStyleParam.step	Получить свойство (*)
iTextStyleParam.step = step	Установить свойство (*)
step = iTextStyleParam.GetStep()	Получить свойство (**)
iTextStyleParam.SetStep(step)	Установить свойство (**)

stepParPre – Интервал перед текстом (в миллиметрах)

Интерфейс...

Тип данных:double.

Синтаксис Automation:

stepParPre = iTextStyleParam.stepParPre	Получить свойство (*)
iTextStyleParam.stepParPre = stepParPre	Установить свойство (*)
stepParPre = iTextStyleParam.GetStepParPre()	Получить свойство (**)
iTextStyleParam.SetStepParPre(stepParPre)	Установить свойство (**)

stepParPst – Интервал после текста (в миллиметрах)

Интерфейс...

Тип данных:double.

Синтаксис Automation:

stepParPst = iTextStyleParam.stepParPst	Получить свойство (*)
iTextStyleParam.stepParPst = stepParPst	Установить свойство (*)
stepParPst = iTextStyleParam.GetStepParPst()	Получить свойство (**)
iTextStyleParam.SetStepParPst(stepParPst)	Установить свойство (**)

underline – Подчеркивание символов

Интерфейс...

Тип данных:short.

Значения свойства:

0	- подчеркивание отсутствует,
1	- одинарное подчеркивание.

Синтаксис Automation:

underline = iTextStyleParam.underline	Получить свойство (*)
iTextStyleParam.underline = underline	Установить свойство (*)

underline = iTextStyleParam.GetUnderline()	Получить свойство (**)
iTextStyleParam.SetUnderline(underline)	Установить свойство (**)

ksTextStyleParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Параметры стиля в библиотеке (Интерфейс ksLibraryStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1087_129_2_2_Obshchij_porjadok_.htm

Интерфейс параметров стиля в библиотеке стилей.

Аналог данных параметров при использовании API экспортных функций - LibraryStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLibraryStyleParam – свойства

styleId – Номер стиля в библиотеке

Интерфейс...

Тип данных:long.

Синтаксис Automation:

styleId = iLibraryStyleParam.styleId	Получить свойство (*)
iLibraryStyleParam.styleId = styleId	Установить свойство (*)
styleId = iLibraryStyleParam.GetStyleId()	Получить свойство (**)
iLibraryStyleParam.SetStyleId(styleId)	Установить свойство (**)

styleName – Имя стиля

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

styleName = iLibraryStyleParam.styleName	Получить свойство (*)
iLibraryStyleParam.styleName = styleName	Установить свойство (*)
styleName = iLibraryStyleParam.GetStyleName()	Получить свойство (**)
iLibraryStyleParam.SetStyleName(styleName)	Установить свойство (**)

ksLibraryStyleParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры стиля в библиотеке стилей.

Стиль из библиотеки (Интерфейс ksLibStyle)

Справка системы КОМПАС...

KOMPAS.chm::/1090_129_2_4_Nomer_stilja.htm

Интерфейс параметров для подключения стиля из библиотеки.

Аналог данных параметров при использовании API экспортных функций - LibStyle.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLibStyle – свойства

fileName – Имя файла библиотеки

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

fileName = iLibStyle.fileName	Получить свойство (*)
iLibStyle.fileName = fileName	Установить свойство (*)
fileName = iLibStyle.GetFileName()	Получить свойство (**)
iLibStyle.SetFileName(fileName)	Установить свойство (**)

styleNumber – Номер стиля в библиотеке

Интерфейс...

Тип данных:long.

Синтаксис Automation:

styleNumber = iLibStyle. styleNumber	Получить свойство (*)
iLibStyle.styleNumber = styleNumber	Установить свойство (*)
styleNumber = iLibStyle.GetStyleNumber()	Получить свойство (**)
iLibStyle.SetStyleNumber(styleNumber)	Установить свойство (**)

typeAllocation – Тип размещения стиля в документе

Интерфейс...

Тип данных:short.

Значения свойства:

1	- ссылкой на библиотеку,
0	- "телом" (стиль хранится в документе).

Синтаксис Automation:

typeAllocation = iLibStyle. typeAllocation	Получить свойство (*)
iLibStyle.typeAllocation = typeAllocation	Установить свойство (*)
typeAllocation = iLibStyle.GetTypeAllocation()	Получить свойство (**)
iLibStyle.SetTypeAllocation(typeAllocation)	Установить свойство (**)

ksLibStyle – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры для подключения стиля из библиотеки.

Интерфейсы параметров размеров

Настройки размеров (Интерфейс ksDimensionsOptions)

Справка системы КОМПАС...

KOMPAS.chm::/186_23_5_Nastrojka_razmerov_v_t.htm

Интерфейс параметров настроек размеров.

Аналог данных параметров при использовании API экспортных функций - DimensionsOptions.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksDimensionsOptions – свойства

anglePrecisionLevel – Точность углового размера

Интерфейс...

Тип данных:long.

Значения свойства:

0 - градусы,
1 - минуты,
2 - секунды.

Синтаксис Automation:

anglePrecisionLevel = iDimensionsOptions.anglePrecisionLevel	Получить свойство (*)
iDimensionsOptions.anglePrecisionLevel = anglePrecisionLevel	Установить свойство (*)
anglePrecisionLevel = iDimensionsOptions.GetAnglePrecisionLevel()	Получить свойство (**)
iDimensionsOptions.SetAnglePrecisionLevel(anglePrecisionLevel)	Установить свойство (**)

arrowLength – Длина стрелки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

arrowLength = iDimensionsOptions.arrowLength — Получить свойство (*)
iDimensionsOptions.arrowLength = arrowLength — Установить свойство (*)
arrowLength = iDimensionsOptions.GetArrowLength() — Получить свойство (**)
iDimensionsOptions.SetArrowLength(arrowLength) — Установить свойство (**)

decimalsCount – Количество знаков после запятой (от 0 до 9)

Интерфейс...

Тип данных:short.

Синтаксис Automation:

decimalsCount = iDimensionsOptions.decimalsCount — Получить свойство (*)
iDimensionsOptions.decimalsCount = decimalsCount — Установить свойство (*)
decimalsCount = iDimensionsOptions.GetDecimalsCount() — Получить свойство (**)
iDimensionsOptions.SetDecimalsCount(decimalsCount) — Установить свойство (**)

dimLineExtension – Выход размерной линии за текст

Интерфейс...

Тип данных:double.

Синтаксис Automation:

dimLineExtension = iDimensionsOptions.dimLineExtension — Получить свойство (*)
iDimensionsOptions.dimLineExtension = dimLineExtension — Установить свойство (*)
dimLineExtension = iDimensionsOptions.GetDimLineExtension() — Получить свойство (**)
iDimensionsOptions.SetDimLineExtension(dimLineExtension) — Установить свойство(**)

hiddenToleranceNumber – Квалитет

Интерфейс...

Тип данных:long.

Значения свойства:

-1 - не включен,
число от 1 до 17 - максимальный номер показываемого квалитета.

Синтаксис Automation:

hiddenToleranceNumber = iDimensionsOptions.hiddenToleranceNumber — Получить свойство (*)
iDimensionsOptions.hiddenToleranceNumber = hiddenToleranceNumber — Установить свойство (*)
hiddenToleranceNumber = iDimensionsOptions.GetHiddenToleranceNumber() — Получить свойство (**)
iDimensionsOptions.SetHiddenToleranceNumber(hiddenToleranceNumber) — Установить свойство(**)

proLineExtension – Выход выносных линий за размерную линию

Интерфейс...

Тип данных:double.

Синтаксис Automation:

proLineExtension = iDimensionsOptions.proLineExtension	Получить свойство (*)
iDimensionsOptions.proLineExtension = proLineExtension	Установить свойство (*)
proLineExtension = iDimensionsOptions.GetProLineExtension()	Получить свойство (**)
iDimensionsOptions.SetProLineExtension(proLineExtension)	Установить свойство (**)

textDistanceFromDimLine – Расстояние от размерной линии до текста

Интерфейс...

Тип данных:double.

Синтаксис Automation:

textDistanceFromDimLine = iDimensionsOptions.textDistanceFromDimLine	Получить свойство (*)
iDimensionsOptions.textDistanceFromDimLine = textDistanceFromDimLine	Установить свойство (*)
textDistanceFromDimLine = iDimensionsOptions.GetTextDistanceFromDimLine()	Получить свойство (**)
iDimensionsOptions.SetTextDistanceFromDimLine(textDistanceFromDimLine)	Установить свойство (**)

textDistanceFromProLine – Расстояние от выносных линий до текста

Интерфейс...

Тип данных:double.

Синтаксис Automation:

textDistanceFromProLine = iDimensionsOptions.textDistanceFromProLine	Получить свойство (*)
iDimensionsOptions.textDistanceFromProLine = textDistanceFromProLine	Установить свойство (*)
textDistanceFromProLine = iDimensionsOptions.GetTextDistanceFromProLine()	Получить свойство (**)
iDimensionsOptions.SetTextDistanceFromProLine(textDistanceFromProLine)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных:long.

Синтаксис Automation:

style = iDimensionsOptions.style	Получить свойство (*)
iDimensionsOptions.style = style	Установить свойство (*)
style = iDimensionsOptions.GetStyle()	Получить свойство (**)
iDimensionsOptions.SetStyle(style)	Установить свойство (**)

Системные стили текста...

ksDimensionsOptions – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры настроек размеров.

Объекты, составляющие размер (Интерфейс ksDimensionPartsParam)

Справка системы КОМПАС...

KOMPAS.chm::/176_Glava23_Obshchie_svedenija_.htm

Интерфейс параметров составляющих объектов размера.

Аналог данных параметров при использовании API экспортных функций - DimensionPartsParam.

Примечания:

1. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.
2. Данный интерфейс содержит указатели на все составляющие размера. Для их получения "рассыпается" копия размера в памяти. Размер в документе остается единым объектом.

   Смотрите также: KompasObject

ksDimensionPartsParam – свойства

curveExt – Указатель на продолжение базовой кривой (у радиального размера дуги) или линию–"указатель", проведенную от размерной надписи к дуге (у размера дуги)

Интерфейс...

Тип данных:long.

Синтаксис Automation:

curveExt = iDimensionPartsParam.curveExt	Получить свойство (*)
iDimensionPartsParam.curveExt = curveExt	Установить свойство (*)
curveExt = iDimensionPartsParam.GetCurveExt()	Получить свойство (**)
iDimensionPartsParam.SetCurveExt(curveExt)	Установить свойство (**)

dimLine – Указатель на размерную линию

Интерфейс...

Тип данных:long.

Синтаксис Automation:

dimLine = iDimensionPartsParam.dimLine	Получить свойство (*)
iDimensionPartsParam.dimLine = dimLine	Установить свойство (*)
dimLine = iDimensionPartsParam.GetDimLine()	Получить свойство (**)
iDimensionPartsParam.SetDimLine(dimLine)	Установить свойство (**)

dimLine1 – Указатель на продолжение размерной линии

Интерфейс...

Тип данных:long.

Синтаксис Automation:

dimLine1 = iDimensionPartsParam.dimLine1	Получить свойство (*)
iDimensionPartsParam.dimLine1 = dimLine1	Установить свойство (*)
dimLine1 = iDimensionPartsParam.GetDimLine1()	Получить свойство (**)
iDimensionPartsParam.SetDimLine1(dimLine1)	Установить свойство (**)

gr – Указатель на временную группу всех объектов размера, включая тексты

Интерфейс...

Тип данных:long.

Синтаксис Automation:

gr = iDimensionPartsParam.gr	Получить свойство (*)
iDimensionPartsParam.gr = gr	Установить свойство (*)
gr = iDimensionPartsParam.GetGr()	Получить свойство (**)
iDimensionPartsParam.SetGr(gr)	Установить свойство (**)

leg – Указатель на "ножку" выносной полки

Интерфейс...

Тип данных:long.

Синтаксис Automation:

leg = iDimensionPartsParam.leg	Получить свойство (*)
iDimensionPartsParam.leg = leg	Установить свойство (*)
leg = iDimensionPartsParam.GetLeg()	Получить свойство (**)
iDimensionPartsParam.SetLeg(leg)	Установить свойство (**)

line1 – Указатель на первую выносную линию

Интерфейс...

Тип данных:long.

Синтаксис Automation:

line1 = iDimensionPartsParam.line1	Получить свойство (*)
iDimensionPartsParam.line1 = line1	Установить свойство (*)
line1 = iDimensionPartsParam.GetLine1()	Получить свойство (**)
iDimensionPartsParam.SetLine1(line1)	Установить свойство (**)

line2 – Указатель на вторую выносную линию

Интерфейс...

Тип данных:long.

Синтаксис Automation:

line2 = iDimensionPartsParam.line2	Получить свойство (*)
iDimensionPartsParam.line2 = line2	Установить свойство (*)
line2 = iDimensionPartsParam.GetLine2()	Получить свойство (**)
iDimensionPartsParam.SetLine2(line2)	Установить свойство (**)

shelf – Указатель на выносную полку

Интерфейс...

Тип данных:long.

Синтаксис Automation:

shelf = iDimensionPartsParam.shelf	Получить свойство (*)
iDimensionPartsParam.shelf = shelf	Установить свойство (*)
shelf = iDimensionPartsParam.GetShelf()	Получить свойство (**)
iDimensionPartsParam.SetShelf(shelf)	Установить свойство (**)

ksDimensionPartsParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры объектов размера.

Размерная надпись (Интерфейс ksDimTextParam)

Интерфейс параметров размерной надписи.

Аналог данных параметров при использовании API экспортных функций - DimText.

Примечание:

1. Строки в массив ложатся в следующей последовательности:

   Префикс. Если он добавлен, в bitFlag взводится признак _PREFIX,

   Номинальное значение зависит от флагов _AUTONOMINAL и _NOMINALOFF.

   Номинальное значение добавляется, если _AUTONOMINAL и _NOMINALOFF выключены.

   Квалитет _TOLERANCE.

   Верхнее и нижнее отклонения _DEVIATION и _DEVIATION_INFORM.

   Единицы измерения _UNIT.

   Текст после _SUFFIX.

   Остальные строки это текст после.
2. Если флаг TOLERANCE включен, то значения отклонений не устанавливаются независимо от наличия их в массиве. Они воспринимаются как информационные.
3. Перед вызовом GetObjParam нужно установить stringFlag в 0, если нужно получить массив строк символов CHAR_STR_ARR, или в 1, если нужно получить массив текстов TEXT_LINE_ARR. Для того, чтобы добавить верхнее и нижнее отклонение, нужно рассчитать индекс с учетом включенных или выключенных флагов _PREFIX, _AUTONOMINAL, _NOMINALOFF, и TOLERANCE. Также нужно проверить, включен уже флаг _DEVIATION или нет. Если включен TOLERANCE, данный элемент нужно будет удалить.
4. Динамический массив TEXT_LINE_ARR содержит структуру TextLineParam.

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksDimTextParam – свойства

bitFlag – Признак, определяющий ручное задание размерной надписи или набор битовых полей, задающих признаки размерной надписи

Интерфейс...

Тип данных:long.

Значения свойства:

0 - ручное задание размерной надписи,
>0 - набор битовых полей, задающих признаки размерной надписи.

Синтаксис Automation:

```
bitFlag = iDimText.bitFlag                Получить свойство (* )
iDimText.bitFlag = bitFlag                Установить свойство (* )
bitFlag = iDimText.GetBitFlag()           Получить свойство (**)
iDimText.SetBitFlag( bitFlag )            Установить свойство (**)
```

sign – Тип условного значка перед номиналом размера

Интерфейс...

Тип данных:long.

Значения свойства:

```
0          - нет значка,
1          - диаметр,
2          - квадрат,
3          - радиус,
>3         - номер значка из шрифта "Symbol type A",
4          - M - метрическая резьба,
210        - символ сферы.
```

Синтаксис Automation:

```
sign = iDimText.sign              Получить свойство (* )
iDimText.sign = sign              Установить свойство (* )
sign = iDimText.GetSign()         Получить свойство (**)
iDimText.SetSign( sign )          Установить свойство (**)
```

stringFlag – Тип используемого массива строк

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE	- тип CHAR_STR_ARR - динамический массив строк символов,
TRUE	- тип TEXT_LINE_ARR - динамический массив строк текста.

Синтаксис Automation:

stringFlag = iDimText.stringFlag	Получить свойство (*)
iDimText.stringFlag = stringFlag	Установить свойство (*)
stringFlag = iDimText.GetStringFlag()	Получить свойство (**)
iDimText.SetStringFlag(stringFlag)	Установить свойство (**)

Примечание:

Для функции ksDocument2D::ksGetObjParam нужно определить свойство stringFlag.

Смотрите также:

- ksDimTextParam::SetTextArr
- ksDimTextParam::GetTextArr

style – Стиль текста размера

Интерфейс...

Тип данных:long.

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Синтаксис Automation:

style = iDimText.style	Получить свойство (*)
iDimText.style = style	Установить свойство (*)
style = iDimText.GetStyle()	Получить свойство (**)
iDimText.SetStyle(style)	Установить свойство (**)

Системные стили текста...

ksDimTextParam – методы

GetBitFlagValue – Получить значение одного из флагов признака bitFlag

Интерфейс...

Синтаксис Automation:

BOOL GetBitFlagValue (long bitFlag);

Входной параметр:

bitFlag - битовый флаг, характеризующий определенный признак размерной надписи.

Возвращаемое значение:

TRUE - данный битовый флаг включен,
FALSE - данный битовый флаг выключен.

GetTextArr – Получить указатель на интерфейс динамического массива строк

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTextArr();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray строк типа CHAR_STR_ARR или TEXT_LINE_ARR.

Смотрите также: ksTextLineParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init(BOOL stringFlag);

Входной параметр:

stringFlag - признак типа создаваемого массива строк:
TRUE - создается динамический массив строк текста типа TEXT_LINE_ARR,
FALSE - создается динамический массив строк текста типа CHAR_STR_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры размерной надписи.

SetBitFlagValue – Установить значение одного из флагов признака bitFlag

Интерфейс...

Синтаксис Automation:

BOOL SetBitFlagValue (long bitFlag,

BOOL state);

Входные параметры:

bitFlag	- битовый флаг, характеризующий определенный признак размерной надписи,
state	- состояние указанного битового флага: TRUE - включен, FALSE - выключен.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTextArr – Установить динамический массив параметров строк

Интерфейс...

Синтаксис Automation:

BOOL SetTextArr (LPDISPATCH pText);

Входной параметр:

pText	- указатель на интерфейс динамического массива ksDynamicArray строк типа CHAR_STR_ARR или TEXT_LINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также: ksTextLineParam

Квалитеты

Квалитет (Интерфейс ksQualityContensParam)

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Интерфейс параметров квалитета.

Аналог данных параметров при использовании API экспортных функций - QualityContensParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruc.

Смотрите также: KompasObject

ksQualityContensParam – свойства

kindQuality – Тип квалитета

Интерфейс...

Тип данных:short.

Значения свойства:

1	- предпочтительный,
2	- основной,
3	- дополнительный.

Синтаксис Automation:

kindQuality = iQualityContensParam.kindQuality	Получить свойство (*)
iQualityContensParam.kindQuality = kindQuality	Установить свойство (*)
kindQuality = iQualityContensParam.GetKindQuality()	Получить свойство (**)
iQualityContensParam.SetKindQuality(kindQuality)	Установить свойство (**)

name – Поле допуска

Интерфейс...

Тип данных:строка.

Синтаксис Automation:

name = iQualityContensParam.name	Получить свойство (*)
iQualityContensParam.name = name	Установить свойство (*)
name = iQualityContensParam.GetName()	Получить свойство (**)
iQualityContensParam.SetName(name)	Установить свойство (**)

systemQuality – Система квалитета

Интерфейс...

Тип данных:short.

Значения свойства:

1	- система вала,
2	- система отверстия.

Синтаксис Automation:

systemQuality = iQualityContensParam.systemQuality	Получить свойство (*)
iQualityContensParam.systemQuality = systemQuality	Установить свойство (*)
systemQuality = iQualityContensParam.GetSystemQuality()	Получить свойство (**)
iQualityContensParam.SetSystemQuality(systemQuality)	Установить свойство (**)

ksQualityContensParam – методы

GetpQualityItems – Получить указатель на интерфейс динамического массива структур параметров интервала квалитета типа QUALITYITEM_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpQualityItems();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray структур параметров интервала квалитета типа QUALITYITEM_ARR.

Смотрите также: ksQualityItemParam

Init– Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры квалитета.

SetpQualityItems – Установить указатель на интерфейс динамического массива структур параметров интервала квалитета типа QUALITYITEM_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpQualityItems (LPDISPATCH pGab);

Входной параметр:

pGab - указатель на интерфейс динамического массива ksDynamicArray структур параметров интервала квалитета типа QUALITYITEM_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Смотрите также: ksQualityItemParam

Интервал квалитета (Интерфейс ksQualityItemParam)

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Интерфейс параметров интервала квалитета.

Аналог данных параметров при использовании API экспортных функций - QualityItemParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksQualityItemParam – свойства

high – Верхнее отклонение

Интерфейс...

Тип данных:double.

Синтаксис Automation:

high = iQualityItemParam.high	Получить свойство (*)
iQualityItemParam.high = high	Установить свойство (*)
high = iQualityItemParam.GetHigh()	Получить свойство (**)
iQualityItemParam.SetHigh(high)	Установить свойство (**)

low – Нижнее отклонение

Интерфейс...

Тип данных:double.

Синтаксис Automation:

low = iQualityItemParam.low	Получить свойство (*)
iQualityItemParam.low = low	Установить свойство (*)
low = iQualityItemParam.GetLow()	Получить свойство (**)
iQualityItemParam.SetLow(low)	Установить свойство (**)

maxLimit – Максимальное значение в интервале размеров

Интерфейс...

Тип данных:short.

Синтаксис Automation:

maxLimit = iQualityItemParam.maxLimit	Получить свойство (*)
iQualityItemParam.maxLimit = maxLimit	Установить свойство (*)
maxLimit = iQualityItemParam.GetMaxLimit()	Получить свойство (**)
iQualityItemParam.SetMaxLimit(maxLimit)	Установить свойство (**)

minLimit – Минимальное значение в интервале размеров

Интерфейс...

Тип данных:short.

Синтаксис Automation:

minLimit = iQualityItemParam.minLimit	Получить свойство (*)
iQualityItemParam.minLimit = minLimit	Установить свойство (*)
minLimit = iQualityItemParam.GetMinLimit()	Получить свойство (**)
iQualityItemParam.SetMinLimit(minLimit)	Установить свойство (**)

ksQualityItemParamметоды

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры записи об одном интервале для квалитета.

Линейные и угловые размеры

Отрисовка линейных и угловых размеров (Интерфейс ksDimDrawingParam)

Справка системы КОМПАС: линейный размер

KOMPAS.chm::/CM_DIML.htm

Справка системы КОМПАС:угловой размер

KOMPAS.chm::/CM_DIMA.htm

Интерфейс параметров отрисовки линейного и углового размеров.

Аналог данных параметров при использовании API экспортных функций - DimDrawing.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksDimDrawingParam – свойства

ang – Угол наклона "ножки" выносной полки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iDimDrawing.ang	Получить свойство (*)
iDimDrawing.ang = ang	Установить свойство (*)
ang = iDimDrawing.GetAng()	Получить свойство (**)
iDimDrawing.SetAng(ang)	Установить свойство (**)

lenght – Длина "ножки" выносной полки

Интерфейс...

Тип данных:long.

Синтаксис Automation:

lenght = iDimDrawing.lenght	Получить свойство (*)
iDimDrawing.lenght = lenght	Установить свойство (*)

lenght = iDimDrawing.GetLenght() Получить свойство (**)
iDimDrawing.GetLenght(lenght) Установить свойство (**)

Примечание:

Данное свойство - аннотационное (не зависит от масштаба, измеряется "по бумаге").

pl1 – Признак отрисовки первой выносной линии

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE - первая выносная линия отрисовывается,
TRUE - первая выносная линия не отрисовывается.

Синтаксис Automation:

pl1 = iDimDrawing.pl1 Получить свойство (*)
iDimDrawing.pl1 = pl1 Установить свойство (*)
pl1 = iDimDrawing.GetPl1() Получить свойство (**)
iDimDrawing.SetPl1(pl1) Установить свойство (**)

pl2 – Признак отрисовки второй выносной линии

Интерфейс...

Тип данных:BOOL.

Значения свойства:

FALSE - вторая выносная линия отрисовывается,
TRUE - вторая выносная линия не отрисовывается.

Синтаксис Automation:

pl2 = iDimDrawing.pl2 Получить свойство (*)
iDimDrawing.pl2 = pl2 Установить свойство (*)
pl2 = iDimDrawing.GetPl2() Получить свойство (**)
iDimDrawing.SetPl2(pl2) Установить свойство (**)

pt1 – Тип стрелки у первой выносной линии размера

Интерфейс...

Тип данных:short.

Значения свойства:

0 - стрелки нет,

1	- стрелка внутри,
2	- стрелка снаружи,
3	- засечка,
4	- точка.

Синтаксис Automation:

pt1 = iDimDrawing.pt1	Получить свойство (*)
iDimDrawing.pt1 = pt1	Установить свойство (*)
pt1 = iDimDrawing.GetPt1()	Получить свойство (**)
iDimDrawing.SetPt1(pt1)	Установить свойство (**)

pt2 – Тип стрелки у второй выносной линии размера

Интерфейс...

Тип данных:short.

Значения свойства:

Типы отрисовки стрелок...

Синтаксис Automation:

pt2 = iDimDrawing.pt2	Получить свойство (*)
iDimDrawing.pt2 = pt2	Установить свойство (*)
pt2 = iDimDrawing.GetPt2()	Получить свойство (**)
iDimDrawing.SetPt2(pt2)	Установить свойство (**)

shelfDir – Направление выносной полки

Интерфейс...

Тип данных:long.

Значения свойства:

0	- нет полки,
-1	- полка влево,
1	- полка вправо,
2	- полка вверх,
3	- полка вниз.

Синтаксис Automation:

shelfDir = iDimDrawing.shelfDir	Получить свойство (*)
iDimDrawing.shelfDir = shelfDir	Установить свойство (*)
shelfDir = iDimDrawing.GetShelfDir()	Получить свойство (**)
iDimDrawing.SetShelfDir(shelfDir)	Установить свойство (**)

textPos – Положение текста размерной надписи

Интерфейс...

Тип данных:long.

Значения свойства:

0 - автоматическое размещение текста,
> 0 - размещение текста на указанное расстояние в направлении от первой точки ко второй,
< 0 - размещение текста на указанное расстояние в направлении от второй точки к первой.

Синтаксис Automation:

textPos = iDimDrawing.textPos	Получить свойство (*)
iDimDrawing.textPos = textPos	Установить свойство (*)
textPos = iDimDrawing.GetTextPos()	Получить свойство (**)
iDimDrawing.SetTextPos(textPos)	Установить свойство (**)

Примечание:

Значение данного свойства задается в миллиметрах для линейных размеров и в градусах - для угловых. Этот параметр - аннотационный (не зависит от масштаба, измеряется "по бумаге").

textBase – Параметр отрисовки текста

Интерфейс...

Тип данных:short.

Значения свойства:

0 - в центре,
1 - на расстояние (или угол) textPos относительно первой точки,
2 - на расстояние (или угол) textPos относительно второй точки,
3 - общая размерная линия.

Синтаксис Automation:

textBase = iDimDrawing.textBase	Получить свойство (*)
iDimDrawing.textBase = textBase	Установить свойство (*)
textBase = iDimDrawing.GetTextBase()	Получить свойство (**)
iDimDrawing.SetTextBase(textBase)	Установить свойство (**)

ksDimDrawingParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрисовки линейного и углового размеров.

Отрисовка линейных и угловых размеров с обрывом (Интерфейс ksBreakDimDrawing)

Справка системы КОМПАС: линейный размер

KOMPAS.chm::/CM_CUT_DIML.htm

Справка системы КОМПАС: угловой размер

KOMPAS.chm::/CM_CUT_DIMA.htm

Интерфейс параметров отрисовки линейного и углового размера с обрывом.

Аналог данных параметров при использовании API экспортных функций - BreakDimDrawing.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksBreakDimDrawing – свойства

angle – Угол наклона "ножки" полки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

angle = iBreakDimDrawing.angle	Получить свойство (*)
iBreakDimDrawing.angle = angle	Установить свойство (*)
angle = iBreakDimDrawing.GetAngle()	Получить свойство (**)
iBreakDimDrawing.SetAngle(angle)	Установить свойство (**)

length– Длина “ножки” полки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

length = iBreakDimDrawing.length	Получить свойство (*)
iBreakDimDrawing.length = length	Установить свойство (*)
length = iBreakDimDrawing.GetLength()	Получить свойство (**)
iBreakDimDrawing.SetLength(length)	Установить свойство (**)

Примечание :

Свойство является аннотационным (не зависит от масштаба, показывает длину "на бумаге").

pl – Признак отрисовки выносной линии

Интерфейс...

Тип данных:BOOL.

Значения свойства:

TRUE	- отрисовка выносной линии выключена,
FALSE	- отрисовка выносной линии включена.

Синтаксис Automation:

pl = iBreakDimDrawing.pl	Получить свойство (*)
iBreakDimDrawing.pl = pl	Установить свойство (*)
pl = iBreakDimDrawing.GetPl()	Получить свойство (**)
iBreakDimDrawing.SetPl(pl)	Установить свойство (**)

pt – Тип стрелки у первой выносной линии

Интерфейс...

Тип данных:short.

Значения свойства:

Типы отрисовки стрелок...

Синтаксис Automation:

pt = iBreakDimDrawing.pt	Получить свойство (*)
iBreakDimDrawing.pt = pt	Установить свойство (*)
pt = iBreakDimDrawing.GetPt()	Получить свойство (**)
iBreakDimDrawing.SetPt(pt)	Установить свойство (**)

shelfDir – Признак отрисовки полки

Интерфейс...

Тип данных:long.

Значения свойства:

0	- нет полки,
-1	- полка влево,
1	- полка вправо,
2	- полка вверх,
3	- пока вниз.

Синтаксис Automation:

shelfDir = iBreakDimDrawing.shelfDir	Получить свойство (*)
iBreakDimDrawing.shelfDir = shelfDir	Установить свойство (*)
shelfDir = iBreakDimDrawing.GetShelfDir()	Получить свойство (**)
iBreakDimDrawing.SetShelfDir(shelfDir)	Установить свойство (**)

textPos – Положение текста

Интерфейс...

Тип данных:long.

Значения свойства:

0	- автоматическое размещение текста,
> 0	- значение расстояния от выносной линии до текста в направлении от первой точки ко второй.

Синтаксис Automation:

textPos = iBreakDimDrawing.textPos	Получить свойство (*)
iBreakDimDrawing.textPos = textPos	Установить свойство (*)
textPos = iBreakDimDrawing.GetTextPos()	Получить свойство (**)
iBreakDimDrawing.SetTextPos(textPos)	Установить свойство (**)

Примечание:

Свойство является аннотационным (не зависит от масштаба, показывает расстояние "на бумаге").

ksBreakDimDrawing – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрисовки линейного и углового размера с обрывом.

Линейные размеры

Параметры размера (Интерфейс ksLDimParam)

Справка системы КОМПАС...

KOMPAS.chm::/191_Glava24_Linejnye_razmery.htm

Интерфейс параметров линейного размера.

Аналог данных параметров при использовании API экспортных функций - LDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLDimParam – методы

GetDPar – Получить указатель на интерфейс параметров отрисовки линейного размера ksDimDrawingParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров отрисовки линейного размера ksDimDrawingParam.

GetSPar – Получить указатель на интерфейс параметров привязки линейного размера ksLDimSourceParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки линейного размера ksLDimSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи ksDimTextParam.

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить параметры изображения размера

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров отрисовки линейного размера ksDimDrawingParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetSPar – Установить параметры привязки линейного размера

Интерфейс...

Синтаксис Automation:

BOOL SetSPar (LPDISPATCH sPar);

Входной параметр:

sPar	- указатель на интерфейс параметров привязки линейного размера ksLDimSourceParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTPar – Установить размерную надпись

Интерфейс...

Синтаксис Automation:

BOOL SetTPar(LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Привязка размера (Интерфейс ksLDimSourceParam)

Справка системы КОМПАС..

KOMPAS.chm::/191_Glava24_Linejnye_razmery.htm

Интерфейс параметров привязки линейного размера.

Аналог данных параметров при использовании API экспортных функций - LDimSource.

Примечания:

1. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.
2. Вектор, определяющий положение размерной линии - аннотационный (не зависит от масштаба).

   Смотрите также: KompasObject

ksLDimSourceParam – свойства

basePoint – Признак, указывающий, от какой точки откладывать вектор по dx,dy

Интерфейс...

Тип данных:short.

Значения свойства:

1	- от первой точки,
2	- от второй точки.

Синтаксис Automation:

```
basePoint = iLDimSource.basePoint                Получить свойство (* )
iLDimSource.basePoint = basePoint                Установить свойство (* )
basePoint = iLDimSource.GetBasePoint()           Получить свойство (**)
iLDimSource.SetBasePoint( basePoint )            Установить свойство (**)
```

dx, dy– Вектор, определяющий положение размерной линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
dx = iLDimSource.dx              Получить свойство (* )
iLDimSource.dx = dx              Установить свойство (* )
dx = iLDimSource.GetDx()         Получить свойство (**)
iLDimSource.SetDx( dx )          Установить свойство (**)
```

ps – Признак ориентации размерной линии

Интерфейс...

Тип данных:short.

Значения свойства:

```
0        - горизонтально,
1        - вертикально,
2        - параллельно отрезку,
3        - по dx, dy,
4        - параллельно отрезку с выносными линиями по dx, dy.
```

Синтаксис Automation:

```
ps = iLDimSource.ps              Получить свойство (* )
iLDimSource.ps = ps              Установить свойство (* )
ps = iLDimSource.GetPs()         Получить свойство (**)
iLDimSource.SetPs( ps )          Установить свойство (**)
```

x1, y1 – Координаты первой точки привязки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
x1 = iLDimSource.x1              Получить свойство (* )
iLDimSource.x1 = x1              Установить свойство (* )
x1 = iLDimSource.GetX1()         Получить свойство (**)
```

iLDimSource.SetX1(x1) Установить свойство (**)

x2, y2 – Координаты второй точки привязки

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x2 = iLDimSource.x2 Получить свойство (*)
iLDimSource.x2 = x2 Установить свойство (*)
x2 = iLDimSource.GetX2() Получить свойство (**)
iLDimSource.SetX2(x2) Установить свойство (**)

ksLDimSourceParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры привязки линейного размера.

Параметры размера с обрывом (Интерфейс ksLBreakDimParam)

Интерфейс параметров линейного размера с обрывом.

Аналог данных параметров при использовании API экспортных функций - LBreakDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLBreakDimParam – методы

GetDPar – Получить указатель на интерфейс параметров изображения размера

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров изображения размера ksBreakDimDrawing.

GetSPar – Получить указатель на интерфейс параметров привязки линейного размера

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки линейного размера ksLBreakDimSource.

GetTPar – Получить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить указатель на интерфейс параметров изображения размера

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров изображения размера ksBreakDimDrawing.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetSPar – Установить указатель на интерфейс параметров привязки линейного размера

Интерфейс...

Синтаксис Automation:

BOOL SetSPar(LPDISPATCH sPar);

Входной параметр:

sPar	- указатель на интерфейс параметров привязки линейного размера ksLBreakDimSource.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTPar – Установить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Привязка размера с обрывом (Интерфейс ksLBreakDimSource)

Интерфейс параметров привязки линейного размера с обрывом.

Аналог данных параметров при использовании API экспортных функций - LBreakDimSource.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLBreakDimSource – свойства

x1, y1 – Координаты первой точки привязки

Интерфейс..

Тип данных:double.

Синтаксис Automation:

x1 = iLBreakDimSource.x1	Получить свойство (*)
iLBreakDimSource.x1 = x1	Установить свойство (*)
x1 = iLBreakDimSource.GetX1()	Получить свойство (**)
iLBreakDimSource.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты точки выхода стрелки

Интерфейс..

Тип данных:double.

Синтаксис Automation:

x2 = iLBreakDimSource.x2	Получить свойство (*)
iLBreakDimSource.x2 = x2	Установить свойство (*)
x2 = iLBreakDimSource.GetX2()	Получить свойство (**)
iLBreakDimSource.SetX2(x2)	Установить свойство (**)

x3, y3 – Координаты точки на размерной линии

Интерфейс..

Тип данных:double.

Синтаксис Automation:

x3 = iLBreakDimSource.x3	Получить свойство (*)
iLBreakDimSource.x3 = x3	Установить свойство (*)
x3 = iLBreakDimSource.GetX3()	Получить свойство (**)
iLBreakDimSource.SetX3(x3)	Установить свойство (**)

ksLBreakDimSource – методы

Init – Инициализировать параметры

Интерфейс..

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры привязки линейного размера с обрывом.

Угловые размеры

Параметры размера (Интерфейс ksADimParam)

Справка системы КОМПАС...

KOMPAS.chm::/204_Glava26_Uglovye_razmery.htm

Интерфейс параметров углового размера.

Аналог данных параметров при использовании API экспортных функций - ADimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksADimParam – методы

GetDPar – Получить указатель на интерфейс параметров отрисовки углового размера ksDimDrawingParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров отрисовки углового размера ksDimDrawingParam.

GetSPar – Получить указатель на интерфейс параметров привязки углового размера ksADimSourceParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки углового размера ksADimSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи ksDimTextParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить параметры изображения размера

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров отрисовки линейного размера ksDimDrawingParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetSPar – Установить параметры привязки углового размера

Интерфейс...

Синтаксис Automation:

BOOL SetSPar (LPDISPATCH sPar);

Входной параметр:

sPar	- указатель на интерфейс параметров привязки углового размера ksADimSourceParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTPar – Установить размерную надпись

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Привязка размера (Интерфейс ksADimSourceParam)

Справка системы КОМПАС...

KOMPAS.chm::/204_Glava26_Uglovye_razmery.htm

Интерфейс параметров привязки углового размера.

Аналог данных параметров при использовании API экспортных функций - ADimSource.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksADimSourceParam – свойства

ang1,ang2 – Начальный и конечный угол размерной дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang1 = iADimSource.ang1	Получить свойство (*)
iADimSource.ang1 = ang1	Установить свойство (*)
ang1 = iADimSource.GetAng1()	Получить свойство (**)
iADimSource.SetAng1(ang1)	Установить свойство (**)

dir – Направление размерной дуги

Интерфейс...

Тип данных:long.

Значения свойства:

1	- против часовой стрелки,
-1	- по часовой стрелке.

Синтаксис Automation:

dir = iADimSource.dir	Получить свойство (*)
iADimSource.dir = dir	Установить свойство (*)
dir = iADimSource.GetDir()	Получить свойство (**)
iADimSource.SetDir(dir)	Установить свойство (**)

rad – Радиус размерной дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

rad = iADimSource.rad	Получить свойство (*)
iADimSource.rad = rad	Установить свойство (*)
rad = iADimSource.GetRad()	Получить свойство (**)
iADimSource.SetRad(rad)	Установить свойство (**)

Примечание:

Данное свойство - аннотационное (не зависит от масштаба, измеряется "по бумаге").

xc, yc – Координаты центра размерной дуги

Интерфейс...

Тип данных:double.

Синтаксис Automation:

xc = iADimSource.xc	Получить свойство (*)
iADimSource.xc = xc	Установить свойство (*)

xc = iADimSource.GetXc() Получить свойство (**)
iADimSource.SetXc(xc) Установить свойство (**)

x1, y1 – Координаты точки выхода первой выносной линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x1 = iADimSource.x1 Получить свойство (*)
iADimSource.x1 = x1 Установить свойство (*)
x1 = iADimSource.GetX1() Получить свойство (**)
iADimSource.SetX1(x1) Установить свойство (**)

x2, y2 – Координаты точки выхода второй выносной линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x2 = iADimSource.x2 Получить свойство (*)
iADimSource.x2 = x2 Установить свойство (*)
x2 = iADimSource.GetX2() Получить свойство (**)
iADimSource.SetX2(x2) Установить свойство (**)

ksADimSourceParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры привязки углового размера.

Параметры размера с обрывом (Интерфейс ksABreakDimParam)

Интерфейс параметров углового размера с обрывом.

Аналог данных параметров при использовании API экспортных функций - ABreakDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksABreakDimParam – методы

GetDPar – Получить указатель на интерфейс параметров изображения размера

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров изображения размера ksBreakDimDrawing.

GetSPar – Получить указатель на интерфейс параметров привязки углового размера

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки углового размера ksADimSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить указатель на интерфейс параметров изображения размера

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров изображения размера ksBreakDimDrawing.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetSPar – Установить указатель на интерфейс параметров привязки углового размера

Интерфейс...

Синтаксис Automation:

BOOL SetSPar (LPDISPATCH sPar);

Входной параметр:

sPar	- указатель на интерфейс параметров привязки углового размера ksADimSourceParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTPar – Установить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Размеры высоты

Параметры размера (Интерфейс ksOrdinatedDimParam)

Интерфейс параметров размера высоты.

Аналог данных параметров при использовании API экспортных функций - OrdinatedDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksOrdinatedDimParam – методы

GetDPar – Получить указатель на интерфейс параметров изображения размера высоты

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров изображения размера высоты ksOrdinatedDrawingParam.

GetSPar – Получить указатель на интерфейс параметров привязки размера высоты

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки размера высоты ksOrdinatedSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрисовки размера высоты.

SetDPar – Установить указатель на интерфейс параметров изображения размера высоты

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar - указатель на интерфейс параметров изображения размера высоты ksOrdinatedDrawingParam.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

SetTPar – Установить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Привязка размера (Интерфейс ksOrdinatedSourceParam)

Интерфейс параметров привязки размера высоты.

Аналог данных параметров при использовании API экспортных функций - OrdinatedSource.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksOrdinatedSourceParam – свойства

x0, y0 – Координаты точки, задающей нулевой уровень размера

Интерфейс...

Тип данных:double.

Синтаксис Automation:

x0 = iOrdinatedSourceParam.x0	Получить свойство (*)
iOrdinatedSourceParam.x0 = x0	Установить свойство (*)
x0 = iOrdinatedSourceParam.GetX0()	Получить свойство (**)
iOrdinatedSourceParam.SetX0(x0)	Установить свойство (**)

x1, y1 – Координаты точки, задающей образмериваемый уровень

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
x1 = iOrdinatedSourceParam.x1                Получить свойство (* )
iOrdinatedSourceParam.x1 = x1                Установить свойство (* )
x1 = iOrdinatedSourceParam.GetX1()           Получить свойство (**)
iOrdinatedSourceParam.SetX1(x1)              Установить свойство (**)
```

x2, y2 – Координаты точки, задающей положение размерной надписи

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
x2 = iOrdinatedSourceParam.x2                Получить свойство (* )
iOrdinatedSourceParam.x2 = x2                Установить свойство (* )
x2 = iOrdinatedSourceParam.GetX2()           Получить свойство (**)
iOrdinatedSourceParam.SetX2(x2)              Установить свойство (**)
```

Отрисовка размера (Интерфейс ksOrdinatedDrawingParam)

Интерфейс параметров отрисовки размера высоты.

Аналог данных параметров при использовании API экспортных функций - OrdinatedDrawing.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksOrdinatedDrawingParam – свойства

type – Тип размера высоты

Интерфейс...

Тип данных:long.

Типы размеров высоты...

Синтаксис Automation:

type = iOrdinatedDrawingParam.type	Получить свойство (*)
iOrdinatedDrawingParam.type = type	Установить свойство (*)
type = iOrdinatedDrawingParam.GetType()	Получить свойство (**)
iOrdinatedDrawingParam.SetType(type)	Установить свойство (**)

Радиальные и диаметральные размеры

Параметры размера (Интерфейс ksRDimParam)

Справка системы КОМПАС: диаметральный размер

KOMPAS.chm::/CM_DIMD.htm

Справка системы КОМПАС: простой радиальный размер

KOMPAS.chm::/CM_DIMR.htm

Интерфейс параметров диаметрального и обычного радиального размера.

Аналог данных параметров при использовании API экспортных функций - RDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRDimParam – методы

GetDPar – Получить указатель на интерфейс параметров отрисовки диаметрального и радиального размеров

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров отрисовки диаметрального и радиального размеров ksRDimDrawingParam.

GetSPar – Получить указатель на интерфейс параметров привязки диаметрального и радиального размеров

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки диаметрального и радиального размеров ksRDimSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить указатель на интерфейс параметров отрисовки диаметрального и радиального размеров

Интерфейс...

Синтаксис Automation:

BOOL SetDPar(LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров отрисовки диаметрального и радиального размеров ksRDimDrawingParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetSPar – Установить указатель на интерфейс параметров привязки диаметрального и радиального размеров

Интерфейс...

Синтаксис Automation:

BOOL SetSPar (LPDISPATCH sPar);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Входной параметр:

sPar	- указатель на интерфейс параметров привязки диаметрального и радиального размеров ksRDimSourceParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetTPar – Установить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar	- указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Привязка размера (Интерфейс ksRDimSourceParam)

Справка системы КОМПАС: диаметральный размер

KOMPAS.chm::/CM_DIMD.htm

Справка системы КОМПАС:простой радиальный размер

KOMPAS.chm::/CM_DIMR.htm

Интерфейс параметров привязки диаметрального и радиального размеров.

Аналог данных параметров при использовании API экспортных функций - RDimSource.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRDimSourceParam – свойства

rad – Радиус дуги или окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
rad = iRDimSource.rad                Получить свойство (* )
iRDimSource.rad = rad                Установить свойство (* )
rad = iRDimSource.GetRad()           Получить свойство (**)
iRDimSource.SetRad( rad )            Установить свойство (**)
```

xc, yc – Координаты центра дуги или окружности

Интерфейс...

Тип данных:double.

Синтаксис Automation:

```
xc = iRDimSource.xc                  Получить свойство (* )
iRDimSource.xc = xc                  Установить свойство (* )
xc = iRDimSource.GetXc()             Получить свойство (**)
iRDimSource.SetXc( xc )              Установить свойство (**)
```

ksRDimSourceParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры привязки диаметрального и радиального размеров.

Отрисовка размера (Интерфейс ksRDimDrawingParam)

Справка системы КОМПАС: диаметральный размер

KOMPAS.chm::/CM_DIMD.htm

Справка системы КОМПАС:простой радиальный размер

KOMPAS.chm::/CM_DIMR.htm

Интерфейс параметров отрисовки диаметрального и радиального размеров.

Аналог данных параметров при использовании API экспортных функций - RDimDrawing.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRDimDrawingParam – свойства

ang – Угол наклона размерной линии

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iRDimDrawing.ang	Получить свойство (*)
iRDimDrawing.ang = ang	Установить свойство (*)
ang = iRDimDrawing.GetAng()	Получить свойство (**)
iRDimDrawing.SetAng(ang)	Установить свойство (**)

pt1 – Тип стрелки у первой выносной линии

Интерфейс...

Тип данных:short.

Типы отрисовки стрелок...

Синтаксис Automation:

pt1 = iRDimDrawing.pt1	Получить свойство (*)
iRDimDrawing.pt1 = pt1	Установить свойство (*)
pt1 = iRDimDrawing.GetPt1()	Получить свойство (**)
iRDimDrawing.SetPt1(pt1)	Установить свойство (**)

pt2 – Тип стрелки у второй выносной линии (для диаметрального размера); признак построения (для радиального размера)

Интерфейс...

Тип данных:short.

Типы отрисовки стрелок...

Значения свойства для радиального размера:

0	- размер от центра,
1	- размер не от центра.

Синтаксис Automation:

pt2 = iRDimDrawing.pt2	Получить свойство (*)
iRDimDrawing.pt2 = pt2	Установить свойство (*)
pt2 = iRDimDrawing.GetPt2()	Получить свойство (**)
iRDimDrawing.SetPt2(pt2)	Установить свойство (**)

shelfDir – Направление выносной полки

Интерфейс...

Тип данных:long.

Значения свойства:

0	- нет полки,
-1	- полка влево,
1	- полка вправо,
2	- полка вверх,
3	- полка вниз.

Синтаксис Automation:

shelfDir = iRDimDrawing.shelfDir	Получить свойство (*)
iRDimDrawing.shelfDir = shelfDir	Установить свойство (*)
shelfDir = iRDimDrawing.GetShelfDir()	Получить свойство (**)
iRDimDrawing.SetShelfDir(shelfDir)	Установить свойство (**)

textPos – Параметр отрисовки текста или длина "ножки"

Интерфейс...

Тип данных:long.

Синтаксис Automation:

textPos = iRDimDrawing.textPos	Получить свойство (*)
iRDimDrawing.textPos = textPos	Установить свойство (*)

textPos = iRDimDrawing.GetTextPos() Получить свойство (**)
iRDimDrawing.GetTextPos(textPos) Установить свойство (**)

Синтаксис Automation:

Данный параметр - аннотационный (не зависит от масштаба, измеряется "по бумаге").

ksRDimDrawingParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрисовки диаметрального и радиального размеров.

Параметры размера с изломом (Интерфейс ksRBreakDimParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_DIMR_WITH_BREAK.htm

Интерфейс параметров радиального размера с изломом.

Аналог данных параметров при использовании API экспортных функций - RBreakDimParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRBreakDimParam – методы

GetDPar – Получить указатель на интерфейс параметров отрисовки радиального размера с изломом

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDPar();

Возвращаемое значение:

- указатель на интерфейс параметров отрисовки радиального размера с изломом ksRBreakDrawingParam.

GetSPar – Получить указатель на интерфейс параметров привязки диаметрального и радиального размеров

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSPar();

Возвращаемое значение:

- указатель на интерфейс параметров привязки диаметрального и радиального размеров ksRDimSourceParam.

GetTPar – Получить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTPar();

Возвращаемое значение:

- указатель на интерфейс параметров размерной надписи ksDimTextParam.

SetDPar – Установить указатель на интерфейс параметров отрисовки радиального размера с изломом

Интерфейс...

Синтаксис Automation:

BOOL SetDPar (LPDISPATCH dPar);

Входной параметр:

dPar	- указатель на интерфейс параметров отрисовки радиального размера с изломом ksRBreakDrawingParam.

Возвращаемое значение:

TRUE - в случае удачного завершения.

SetSPar – Установить параметры привязки линейного размера

Интерфейс...

Синтаксис Automation:

BOOL SetSPar (LPDISPATCH sPar);

Входной параметр:

sPar - указатель на интерфейс параметров привязки диаметрального и радиального размеров ksRDimSourceParam.

Возвращаемое значение:

TRUE - в случае удачного завершения.

SetTPar – Установить указатель на интерфейс параметров размерной надписи

Интерфейс...

Синтаксис Automation:

BOOL SetTPar (LPDISPATCH tPar);

Входной параметр:

tPar - указатель на интерфейс параметров размерной надписи ksDimTextParam.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Отрисовка размера с изломом (Интерфейс ksRBreakDrawingParam)

Справка системы КОМПАС...

`KOMPAS.chm::/CM_DIMR_WITH_BREAK.htm`

Интерфейс параметров отрисовки радиального размера с изломом.

Аналог данных параметров при использовании API экспортных функций - RBreakDrawing.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRBreakDrawingParam – свойства

ang – Угол наклона размерной линии в градусах

Интерфейс...

Тип данных:double.

Синтаксис Automation:

ang = iRBreakDrawing.ang	Получить свойство (*)
iRBreakDrawing.ang = ang	Установить свойство (*)
ang = iRBreakDrawing.GetAng()	Получить свойство (**)
iRBreakDrawing.SetAng(ang)	Установить свойство (**)

pb – Длина излома

Интерфейс...

Тип данных:long.

Синтаксис Automation:

pb = iRBreakDrawing.pb	Получить свойство (*)
iRBreakDrawing.pb = pb	Установить свойство (*)
pb = iRBreakDrawing.GetAng()	Получить свойство (**)
iRBreakDrawing.SetAng(pb)	Установить свойство (**)

Примечание:

Данный параметр - аннотационный (он не зависит от масштаба, измеряется "по бумаге").

pt – Тип стрелки

Интерфейс...

Тип данных:short.

Типы отрисовки стрелок..

Синтаксис Automation:

pt = iRBreakDrawing.pt	Получить свойство (*)
iRBreakDrawing.pt = pt	Установить свойство (*)

pt = iRBreakDrawing.GetPt()	Получить свойство (**)
iRBreakDrawing.SetPt(pt)	Установить свойство (**)

ksRBreakDrawingParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры отрисовки радиального размера с изломом.

Интерфейсы параметров обозначений

Штриховка (Интерфейс ksHatchParam)

Интерфейс параметров штриховки.

Аналог данных параметров при использовании API экспортных функций - HatchParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksHatchParam – свойства

ang – Угол штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

ang = iHatchParam.ang	Получить свойство (*)
iHatchParam.ang = ang	Установить свойство (*)
ang = iHatchParam.GetAngle()	Получить свойство (**)
iHatchParam.SetAngle(ang)	Установить свойство (**)

boundaries – Указатель на временную группу границ штриховки

Интерфейс...

Тип данных: long

Синтаксис Automation:

boundaries = iHatchParam.boundaries	Получить свойство (*)
iHatchParam.boundaries = boundaries	Установить свойство (*)
boundaries = iHatchParam.GetBoundaries()	Получить свойство (**)
iHatchParam.SetBoundaries(boundaries)	Установить свойство (**)

Примечание:

Свойство boundaries используется для функций GetObjParam и ksHatch.

color – Цвет штриховки

Интерфейс...

Тип данных: long

Синтаксис Automation:

color = iHatchParam.color — Получить свойство (*)
iHatchParam.color = color — Установить свойство (*)
color = iHatchParam.GetColor() — Получить свойство (**)
iHatchParam.SetColor(color) — Установить свойство (**)

sheeting – Тип угла штриховки

Интерфейс...

Тип данных: short

Значения свойства:

0 - угол штриховки относительно ее границ сохраняется при повороте границ (используется при изображении накатки на деталях),
1 - обычная штриховка (угол штриховки - постоянный).

Синтаксис Automation:

sheeting = iHatchParam.sheeting — Получить свойство (*)
iHatchParam.sheeting = sheeting — Установить свойство (*)
sheeting = iHatchParam.GetSheeting() — Получить свойство (**)
iHatchParam.SetSheeting(sheeting) — Установить свойство (**)

step – Шаг штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

step = iHatchParam.step — Получить свойство (*)
iHatchParam.step = step — Установить свойство (*)
step = iHatchParam.GetStep() — Получить свойство (**)
iHatchParam.SetStep(step) — Установить свойство (**)

style – Стиль штриховки

Интерфейс...

Тип данных: long

Синтаксис Automation:

style = iHatchParam.style — Получить свойство (*)
iHatchParam.style = style — Установить свойство (*)
style = iHatchParam.GetStyle() — Получить свойство (**)
iHatchParam.SetStyle(style) — Установить свойство (**)

Системные стили штриховок...

width – Ширина полосы штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

width = iHatchParam.width	Получить свойство (*)
iHatchParam.width = width	Установить свойство (*)
width = iHatchParam.GetWidth()	Получить свойство (**)
iHatchParam.SetWidth(width)	Установить свойство (**)

Примечание:

width = 0 - штриховать всю область.

x, y – Координаты базовой точки штриховки

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iHatchParam.x	Получить свойство (*)
iHatchParam.x = x	Установить свойство (*)
x = iHatchParam.GetX()	Получить свойство (**)
iHatchParam.SetX(x)	Установить свойство (**)

ksHatchParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры штриховки.

Обозначение центра (Интерфейс ksCentreParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CENTRE_MARKER.htm

Интерфейс параметров объекта "обозначение центра".

Аналог данных параметров при использовании API экспортных функций - CentreParam.

Примечание:

Индексация полуосей обозначения центра следующая:

- Xp- по оси абсцисс системы координат обозначения центра,
- Xm- против оси абсцисс системы координат обозначения центра,
- Yp- по оси ординат системы координат обозначения центра,
- Ym- против оси ординат системы координат обозначения центра.

Система координат обозначения центра - это система, оси которой совпадают с осями обозначения, а сама она повернута относительно текущей системы координат на угол angle.

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCentreParam – свойства

angle – Угол наклона обозначения центра

Интерфейс...

Тип данных: double

Синтаксис Automation:

angle = iCentreParam.angle	Получить свойство (*)
iCentreParam.angle = angle	Установить свойство (*)
angle = iCentreParam.GetAngle()	Получить свойство (**)
iCentreParam.SetAngle(angle)	Установить свойство (**)

Примечание:

Если задана базовая кривая, этот параметр не используется.

baseCurve – Указатель на базовую геометрическую кривую (например, окружность)

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
baseCurve = iCentreParam.baseCurve                  Получить свойство (* )
iCentreParam.baseCurve = baseCurve                  Установить свойство (* )
baseCurve = iCentreParam.GetBaseCurve()             Получить свойство (**)
iCentreParam.SetBaseCurve( baseCurve )              Установить свойство (**)
```

Примечание:

Свойство равно 0, если создается отдельное обозначение центра.

lenXpTail – Длина изображения полуоси, если оно нестандартное, в положительном направлении оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
lenXpTail = iCentreParam.lenXpTail                  Получить свойство (* )
iCentreParam.lenXpTail = lenXpTail                  Установить свойство (* )
lenXpTail = iCentreParam.GetLenXpTail()             Получить свойство (**)
iCentreParam.SetLenXpTail( lenXpTail )              Установить свойство (**)
```

lenXmTail – Длина изображения полуоси, если оно нестандартное, в отрицательном направлении оси X

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
lenXmTail = iCentreParam.lenXmTail                  Получить свойство (* )
iCentreParam.lenXmTail = lenXmTail                  Установить свойство (* )
lenXmTail = iCentreParam.GetLenXmTail()             Получить свойство (**)
iCentreParam.SetLenXmTail( lenXmTail )              Установить свойство (**)
```

lenYpTail – Длина изображения полуоси, если оно нестандартное, в положительном направлении оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
lenYpTail = iCentreParam.lenYpTail                  Получить свойство (* )
iCentreParam.lenYpTail = lenYpTail                  Установить свойство (* )
lenYpTail = iCentreParam.GetLenYpTail()             Получить свойство (**)
iCentreParam.SetLenYpTail( lenYpTail )              Установить свойство (**)
```

lenYmTail – Длина изображения полуоси, если оно нестандартное, в отрицательном направлении оси Y

Интерфейс...

Тип данных: double

Синтаксис Automation:

lenYmTail = iCentreParam.lenYmTail	Получить свойство (*)
iCentreParam.lenYmTail = lenYmTail	Установить свойство (*)
lenYmTail = iCentreParam.GetLenYmTail()	Получить свойство (**)
iCentreParam.SetLenYmTail(lenYmTail)	Установить свойство (**)

standXpTail – Признак стандартного изображения полуоси в положительном направлении оси X

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

standXpTail = iCentreParam.standXpTail	Получить свойство (*)
iCentreParam.standXpTail = standXpTail	Установить свойство (*)
standXpTail = iCentreParam.GetStandXpTail()	Получить свойство (**)
iCentreParam.SetStandXpTail(standXpTail)	Установить свойство (**)

standXmTail – Признак стандартного изображения полуоси в отрицательном направлении оси X

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

standXmTail = iCentreParam.standXmTail	Получить свойство (*)
iCentreParam.standXmTail = standXmTail	Установить свойство (*)
standXmTail = iCentreParam.GetStandXmTail()	Получить свойство (**)
iCentreParam.SetStandXmTail(standXmTail)	Установить свойство (**)

standYpTail – Признак стандартного изображения полуоси в положительном направлении оси Y

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

standYpTail = iCentreParam.standYpTail	Получить свойство (*)

```
iCentreParam.standYpTail = standYpTail                  Установить свойство (* )
standYpTail = iCentreParam.GetStandYpTail()             Получить свойство (**)
iCentreParam.SetStandYpTail( standYpTail )              Установить свойство (**)
```

standYmTail – Признак стандартного изображения полуоси в отрицательном направлении оси Y

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
standYmTail = iCentreParam.standYmTail                  Получить свойство (* )
iCentreParam.standYmTail = standYmTail                  Установить свойство (* )
standYmTail = iCentreParam.GetStandYmTail()             Получить свойство (**)
iCentreParam.SetStandYmTail( standYmTail )              Установить свойство (**)
```

type – Тип обозначения центра

Интерфейс...

Тип данных: short

Значения свойства:

```
0                       - маленький "крестик",
1                       - одна ось,
2                       - две оси.
```

Синтаксис Automation:

```
type = iCentreParam.type                Получить свойство (* )
iCentreParam.type = type                Установить свойство (* )
type = iCentreParam.GetType()           Получить свойство (**)
iCentreParam.SetType( type )            Установить свойство (**)
```

x, y – Координаты точки привязки

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
x = iCentreParam.x              Получить свойство (* )
iCentreParam.x = x              Установить свойство (* )
x = iCentreParam.GetX()         Получить свойство (**)
iCentreParam.SetX( x )          Установить свойство (**)
```

Примечание:

Если задана базовая кривая, эти параметры не используются.

ksCentreParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры объекта "обозначение центра".

Обозначение базы (Интерфейс ksBaseParam)

Интерфейс параметров обозначения базы.

Аналог данных параметров при использовании API экспортных функций - BaseParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksBaseParam – свойства

str – Текст в обозначении базы

Интерфейс...

Тип данных: строка

Синтаксис Automation:

str = iBaseParam.str	Получить свойство (*)
iBaseParam.str = str	Установить свойство (*)
str = iBaseParam.GetStr()	Получить свойство (**)
iBaseParam.SetStr(str)	Установить свойство (**)

Примечание:

Свойство используется при type = FALSE.

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Синтаксис Automation:

style = iBaseParam.style	Получить свойство (*)
iBaseParam.style = style	Установить свойство (*)
style = iBaseParam.GetStyle()	Получить свойство (**)
iBaseParam.SetStyle(style)	Установить свойство (**)

Системные стили текста...

type – Способ задания надписи в обозначении базы

Интерфейс...

Тип данных: BOOL

Значения свойства:

FALSE	- текст в виде строки (текст определяется свойством ksBaseParam::str),
TRUE	- динамический массив компонент текстов (текст определяется массивом pTextItem (смотрите методы ksBaseParam::GetPTextItem и ksBaseParam::SetPTextItem).

Синтаксис Automation:

type = iBaseParam.type	Получить свойство (*)
iBaseParam.type = type	Установить свойство (*)
type = iBaseParam.GetType()	Получить свойство (**)
iBaseParam.SetType(type)	Установить свойство (**)

x1, y1 – Координаты базовой точки (начальная точка "опоры")

Интерфейс...

Тип данных: double

Синтаксис Automation:

x1 = iBaseParam.x1	Получить свойство (*)
iBaseParam.x1 = x1	Установить свойство (*)
x1 = iBaseParam.GetX1()	Получить свойство (**)
iBaseParam.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты конечной точки "опоры"

Интерфейс...

Тип данных: double

Синтаксис Automation:

x2 = iBaseParam.x2	Получить свойство (*)
iBaseParam.x2 = x2	Установить свойство (*)
x2 = iBaseParam.GetX2()	Получить свойство (**)
iBaseParam.SetX2(x2)	Установить свойство (**)

ksBaseParam – методы

GetPTextItem – Получить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPTextItem();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_ITEM_ARR.

Примечание:

Массив используется при ksBaseParam::type = TRUE.

Смотрите также: StructTextItemParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE

Примечание:

Метод обнуляет все параметры объекта "обозначение базы".

SetPTextItem – Установить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPTextItem (LPDISPATCH pTextItem);

Входной параметр:

pTextItem	- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_ITEM_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Массив используется при ksBaseParam::type = TRUE.

Смотрите также: TextItemParam

Обозначение шероховатости

Обозначение шероховатости (Интерфейс ksRoughPar)

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Интерфейс параметров обозначения шероховатости.

Аналог данных параметров при использовании API экспортных функций - RoughPar.

Примечания:

1. Текст над знаком содержит параметры шероховатости по ГОСТ 2789-73.
2. Текст над полкой содержит вид обработки и дополнительные указания по ГОСТ 2789-73.
3. Первый текст под полкой содержит базовую длину по ГОСТ 2789-73.
4. Второй текст под полкой содержит условное обозначение направления неровностей по ГОСТ 2789-73.
5. Если cText0 = 0 или cText1 = 0, или cText2 = 0, или cText3 = 0, то соответствующий текст в обозначении шероховатости отсутствует.
6. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

   Смотрите также: KompasObject

ksRoughPar – свойства

ang – Угол наклона оси знака шероховатости к оси OX

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ang = iRoughPar.ang              Получить свойство (* )
iRoughPar.ang = ang              Установить свойство (* )
ang = iRoughPar.GetAng()         Получить свойство (**)
iRoughPar.SetAng( ang )          Установить свойство (**)
```

around – Признак наличия обозначения контура

Интерфейс...

Тип данных: short

Значения свойства:

```
0          - обычный знак шероховатости,
1          - шероховатость "по контуру".
```

Синтаксис Automation:

```
around = iRoughPar.around              Получить свойство (* )
iRoughPar.around = around              Установить свойство (* )
around = iRoughPar.GetAround()         Получить свойство (**)
iRoughPar.SetAround(around )           Установить свойство (**)
```

cText0 – Количество строк в тексте над знаком шероховатости

Интерфейс...

Тип данных: short

Синтаксис Automation:

```
cText0 = iRoughPar.cText0              Получить свойство (* )
iRoughPar.cText0 = cText0              Установить свойство (* )
cText0 = iRoughPar.GetCText0()         Получить свойство (**)
iRoughPar.SetCText0( cText0 )          Установить свойство (**)
```

cText1 – Количество строк в тексте над полкой знака шероховатости

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText1 = iRoughPar.cText1	Получить свойство (*)
iRoughPar.cText1 = cText1	Установить свойство (*)
cText1 = iRoughPar.GetcText1()	Получить свойство (**)
iRoughPar.SetcText1(cText1)	Установить свойство (**)

cText2 – Количество строк в первом тексте под полкой (не более двух)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText2 = iRoughPar.cText2	Получить свойство (*)
iRoughPar.cText2 = cText2	Установить свойство (*)
cText2 = iRoughPar.GetcText2()	Получить свойство (**)
iRoughPar.SetcText2(cText2)	Установить свойство (**)

cText3 – Количество строк во втором тексте под полкой (не более одной)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText3 = iRoughPar.cText3	Получить свойство (*)
iRoughPar.cText2 = cText3	Установить свойство (*)
cText3 = iRoughPar.GetcText3()	Получить свойство (**)
iRoughPar.SetcText3(cText3)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля, номер пользовательского стиля, INDICATIN_TEXT_LINE_ARR	- в этом случае текст задается массивом строк, и стиль текста может быть указан индивидуально для каждой строки.

Системные стили текстов...

Синтаксис Automation:

style = iRoughPar.style	Получить свойство (*)
iRoughPar.style = style	Установить свойство (*)
style = iRoughPar.GetStyle()	Получить свойство (**)
iRoughPar.SetStyle(style)	Установить свойство (**)

Примечание:

Если style = INDICATIN_TEXT_LINE_ARR, то динамический массив pText - типа TEXT_LINE_ARR. В остальных случаях - типа CHAR_STR_ARR.

type – Тип знака шероховатости

Интерфейс...

Тип данных: short

Значения свойства:

0	- вид обработки не устанавливается,
1	- обработка удалением слоя материала,
2	- обработка без удаления слоя материала.

Синтаксис Automation:

type = iRoughPar.type	Получить свойство (*)
iRoughPar.type = type	Установить свойство (*)
type = iRoughPar.GetType()	Получить свойство (**)
iRoughPar.SetType(type)	Установить свойство (**)

x, y – Координаты точки привязки знака шероховатости

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iRoughPar.x	Получить свойство (*)
iRoughPar.x = x	Установить свойство (*)
x = iRoughPar.GetX()	Получить свойство (**)
iRoughPar.SetX(x)	Установить свойство (**)

ksRoughPar – методы

GetpText – Получить указатель на интерфейс динамического массива текста обозначения шероховатости ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpText();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray.

Примечания:

1. Если свойство ksRoughPar::style = INDICATIN_TEXT_LINE_ARR, то динамический массив - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста шероховатости).
2. Различают четыре разных текста для шероховатости, их строки лежат в следующей последовательности:

- текст над знаком (параметры шероховатости по ГОСТ 2789-73). Если свойство cText0 = 0, то этот текст отсутствует;
- текст над полкой (вид обработки и дополнительные указания). Если свойство cText1 = 0, то этот текст отсутствует;
- текст под полкой (базовая длина по ГОСТ 2789-73). Если свойство cText2 = 0, то этот текст отсутствует;
- текст под полкой (условное обозначение направления неровностей). Если свойство cText3 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Init – Инициализировать параметры

Интерфейс...

Замечание:

Данный метод устарел. Рекомендуется использовать вместо него метод InitEx.

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pText типа CHAR_STR_ARR.

InitEx – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL InitEx (int style);

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Если style = INDICATIN_TEXT_LINE_ARR, то создается динамический массив pText типа TEXT_LINE_ARR, в противном случае создается динамический массив pText типа CHAR_STR_ARR.

SetpText – Установить указатель на интерфейс динамического массива текста обозначения шероховатости ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetpText (LPDISPATCH pText);

Входной параметр:

pText	- указатель на интерфейс динамического массива ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

1. Если свойство ksRoughPar::style = INDICATIN_TEXT_LINE_ARR, то динамический массив ksDynamicArray - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста шероховатости).
2. Различают четыре разных текста для шероховатости, их строки лежат в следующей последовательности:

- текст над знаком (параметры шероховатости по ГОСТ 2789-73). Если свойство cText0 = 0, то этот текст отсутствует;
- текст над полкой (вид обработки и дополнительные указания). Если свойство cText1 = 0, то этот текст отсутствует;

- текст под полкой (базовая длина по ГОСТ 2789-73). Если свойство cText2 = 0, то этот текст отсутствует;
- текст под полкой (усл.обозначение направления неровностей). Если свойство cText3 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Обозначение шероховатости с выносной полкой (Интерфейс ksRoughParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Интерфейс параметров обозначения шероховатости с выносной полкой.

Аналог данных параметров при использовании API экспортных функций - RoughParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksRoughParam – методы

GetrPar – Получить указатель на интерфейс параметров шероховатости ksRoughPar

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetrPar();

Возвращаемое значение:

- указатель на интерфейс параметров знака шероховатости ksRoughPar.

GetshPar – Получить указатель на интерфейс параметров выносной полки ksShelfPar

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetshPar();

Возвращаемое значение:

- указатель на интерфейс параметров выносной полки ksShelfPar.

SetrPar – Установить указатель на интерфейс параметров шероховатости

Интерфейс...

Синтаксис Automation:

BOOL SetrPar (LPDISPATCH rPar);

Входной параметр:

rPar	- указатель на интерфейс параметров знака шероховатости ksRoughPar.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

SetshPar – Установить указатель на интерфейс параметров выносной полки ksShelfPar

Интерфейс...

Синтаксис Automation:

BOOL SetshPar (LPDISPATCH shPar);

Входной параметр:

shPar	- указатель на интерфейс параметров выносной полки ksShelfPar.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Выносная полка (Интерфейс ksShelfPar)

Справка системы КОМПАС...

`KOMPAS.chm::/223_28_3_2_Nastrojka_otrisovki_.htm`

Интерфейс параметров выносной полки.

Аналог данных параметров при использовании API экспортных функций - ShelfPar.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksShelfPar – свойства

ang – Угол наклона размерной линии диаметрального и радиального размеров

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ang = iShelfPar.ang                 Получить свойство (* )
iShelfPar.ang = ang                 Установить свойство (* )
ang = iShelfPar.GetAng()            Получить свойство (**)
        iShelfPar.SetAng( ang       Установить
        )                           свойство (**)
```

length – Длина "ножки" выносной линии размера

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
length = iShelfPar.length           Получить свойство (* )
iShelfPar.length = length           Установить свойство (* )
length = iShelfPar.GetLength()      Получить свойство (**)
iShelfPar.SetLength( length )       Установить свойство (**)
```

psh – Направление полки (относительно оси OX)

Интерфейс...

Тип данных: long

Значения свойства:

0	- нет полки,
-1	- полка влево,
1	- полка вправо,
2	- полка вверх,
3	- полка вниз.

Синтаксис Automation:

psh = iShelfPar.psh	Получить свойство (*)
iShelfPar.psh = psh	Установить свойство (*)
psh = iShelfPar.GetPsh()	Получить свойство (**)
iShelfPar.SetPsh(psh)	Установить свойство (**)

ksShelfPar – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Линия – выноска

Линия – выноска (Интерфейс ksLeaderParam)

Интерфейс параметров линии-выноски.

Аналог данных параметров при использовании API экспортных функций - LeaderParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLeaderParam – свойства

around – Признак наличия на линии-выноске знака обработки по контуру

Интерфейс...

Тип данных: short

Значения свойства:

0	- знак отсутствует,
1	- знак присутствует.

Синтаксис Automation:

around = iLeaderParam.around	Получить свойство (*)
iLeaderParam.around = around	Установить свойство (*)
around = iLeaderParam.GetAround()	Получить свойство (**)
iLeaderParam.SetAround(around)	Установить свойство (**)

arrowType – Тип указателя линии-выноски

Интерфейс...

Тип данных: short

Значения свойства:

0	- указатель отсутствует,
1	- точка,
2	- стрелка,
3	- верхняя половина стрелки,
4	- нижняя половина стрелки.

Синтаксис Automation:

arrowType = iLeaderParam.arrowType	Получить свойство (*)
iLeaderParam.arrowType = arrowType	Установить свойство (*)
arrowType = iLeaderParam.GetArrowType()	Получить свойство (**)
iLeaderParam.SetArrowType(arrowType)	Установить свойство (**)

dirX – Направление полки линии-выноски

Интерфейс...

Тип данных: long

Значения свойства:

0	- нет полки,
-1	- полка влево,
1	- полка вправо,
2	- полка вверх,
3	- полка вниз.

Синтаксис Automation:

dirX = iLeaderParam.dirX	Получить свойство (*)
iLeaderParam.dirX = dirX	Установить свойство (*)
dirX = iLeaderParam.GetDirX()	Получить свойство (**)
iLeaderParam.SetDirX(dirX)	Установить свойство (**)

cText0 – Количество строк текста над полкой линии-выноски

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText0 = iLeaderParam.cText0	Получить свойство (*)
iLeaderParam.cText0 = cText0	Установить свойство (*)
cText0 = iLeaderParam.GetCText0()	Получить свойство (**)
iLeaderParam.SetCText0(cText0)	Установить свойство (**)

cText1 – Количество строк текста под полкой

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText1 = iLeaderParam.cText1	Получить свойство (*)
iLeaderParam.cText1 = cText1	Установить свойство (*)
cText1 = iLeaderParam.GetcText1()	Получить свойство (**)
iLeaderParam.SetcText1(cText1)	Установить свойство (**)

cText2 – Количество строк текста над "ножкой" линии-выноски (не более одной строки)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText2 = iLeaderParam.cText2	Получить свойство (*)
iLeaderParam.cText2 = cText2	Установить свойство (*)
cText2 = iLeaderParam.GetcText2()	Получить свойство (**)
iLeaderParam.SetcText2(cText2)	Установить свойство (**)

cText3 – Количество строк текста под "ножкой" линии-выноски (не более одной строки)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText3 = iLeaderParam.cText3	Получить свойство (*)
iLeaderParam.cText3 = cText3	Установить свойство (*)
cText3 = iLeaderParam.GetcText3()	Получить свойство (**)
iLeaderParam.SetcText3(cText3)	Установить свойство (**)

signType – Тип знака на "ножке" линии-выноски

Интерфейс...

Тип данных: short

Значения свойства:

0	- знак отсутствует,
1	- знак склеивания,
2	- знак пайки,
3	- знак сшивания,
4	- знак соединения внахлестку металлическими скобами,
5	- знак углового соединения металлическими скобами,
6	- знак монтажного шва.

Синтаксис Automation:

signType = iLeaderParam.signType	Получить свойство (*)
iLeaderParam.signType = signType	Установить свойство (*)
signType = iLeaderParam.GetSignType()	Получить свойство (**)
iLeaderParam.SetSignType(signType)	Установить свойство (**)

x, y – Координаты базовой точки линии-выноски

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iLeaderParam.x	Получить свойство (*)
iLeaderParam.x = x	Установить свойство (*)
x = iLeaderParam.GetX()	Получить свойство (**)
iLeaderParam.SetX(x)	Установить свойство (**)

Примечание:

Базовая точка линии-выноски - начало полки, точка выхода из нее "ножки".

ksLeaderParam – методы

GetpPolyline – Получить указатель на интерфейс динамического массива ответвлений линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpPolyline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива текста линии выноски ksDynamicArray типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Примечание:

Различают четыре разных текста для линии-выноски, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текст под полкой. Если свойство cText1 = 0, то этот текст отсутствует;
- текст над ножкой. Если свойство cText2 = 0, то этот текст отсутствует;
- текст под ножкой. Если свойство cText3 = 0, то этот текст отсутствует.

Смотрите также: ksTextLineParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pTextline типа TEXT_LINE_ARR.
3. Создается динамический массив pPolytline типа POLYLINE_ARR.

SetpPolyline – Установить указатель на интерфейс динамического массива ответвлений линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpPolyline (LPDISPATCH pPolyline);

Входной параметр:

pPolyline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива текста линии выноски ksDynamicArray типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Различают четыре разных текста для линии-выноски, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текст под полкой. Если свойство cText1 = 0, то этот текст отсутствует;
- текст над ножкой. Если свойство cText2 = 0, то этот текст отсутствует;
- текст под ножкой. Если свойство cText3 = 0, то этот текст отсутствует.

Смотрите также: ksTextLineParam

Линия – выноска для обозначения позиции (Интерфейс ksPosLeaderParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_POSITIONLEADER.htm

Интерфейс параметров линии-выноски для обозначения позиции.

Аналог данных параметров при использовании API экспортных функций - PosLeaderParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksPosLeaderParam – свойства

arrowType – Тип указателя линии-выноски

Интерфейс...

Тип данных: short

Значения свойства:

0 - указатель отсутствует,
1 - точка,
2 - стрелка.

Синтаксис Automation:

arrowType = iPosLeaderParam.arrowType Получить свойство (*)
iPosLeaderParam.arrowType = arrowType Установить свойство (*)
arrowType = iPosLeaderParam.GetArrowType() Получить свойство (**)

iPosLeaderParam.SetArrowType(arrowType) Установить свойство (**)

dirX – Направление полки линии-выноски относительно оси OX

Интерфейс...

Тип данных: long

Значения свойства:

1 - вправо (по оси OX),
-1 - влево (против оси OX).

Синтаксис Automation:

dirX = iPosLeaderParam.dirX Получить свойство (*)
iPosLeaderParam.dirX = dirX Установить свойство (*)
dirX = iPosLeaderParam.GetDirX() Получить свойство (**)
iPosLeaderParam.SetDirX(dirX) Установить свойство (**)

dirY – Направление полки линии-выноски относительно оси OY

Интерфейс...

Тип данных: long

Значения свойства:

1 - вверх (по оси OY),
-1 - вниз (против оси OY).

Синтаксис Automation:

dirY = iPosLeaderParam.dirY Получить свойство (*)
iPosLeaderParam.dirY = dirY Установить свойство (*)
dirY = iPosLeaderParam.GetDirY() Получить свойство (**)
iPosLeaderParam.SetDirY(dirY) Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля,

INDICATIN_TEXT_LINE_ARR	- в этом случае текст задается массивом строк, и стиль текста может быть указан индивидуально для каждой строки.

Системные стили текстов...

Синтаксис Automation:

style = iPosLeaderParam.style	Получить свойство (*)
iPosLeaderParam.style = style	Установить свойство (*)
style = iPosLeaderParam.GetStyle()	Получить свойство (**)
iPosLeaderParam.SetStyle(style)	Установить свойство (**)

Примечание:

Если style = INDICATIN_TEXT_LINE_ARR, то динамический массив pText - типа TEXT_LINE_ARR. В остальных случаях - типа CHAR_STR_ARR.

x, y – Координаты базовой точки линии-выноски обозначения позиции

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iPosLeaderParam.x	Получить свойство (*)
iPosLeaderParam.x = x	Установить свойство (*)
x = iPosLeaderParam.GetX()	Получить свойство (**)
iPosLeaderParam.SetX(x)	Установить свойство (**)

Примечание:

Базовая точка линии-выноски - начало первой полки, точка выхода из нее "ножки".

ksPosLeaderParam – методы

GetpPolyline – Получить указатель на интерфейс динамического массива ответвлений позиционной линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpPolyline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива текста обозначения позиции ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Примечания:

1. Если свойство ksPosLeaderParam::style = INDICATIN_TEXT_LINE_ARR, то динамический массив - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк обозначения позиции).
2. Каждая строка соответствует одному номеру позиции.

   Смотрите также:

   ksTextLineParam

   ksChar255

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pText типа CHAR_STR_ARR.
3. Создается динамический массив pPolyline типа POLYLINE_ARR.

SetpPolyline – Установить указатель на интерфейс динамического массива ответвлений позиционной линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpPolyline (LPDISPATCH pPolyline);

Входной параметр:

pPolyline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива текста обозначения позиции ksDynamicArray в каждой строке лежит одна позиция

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Если свойство ksPosLeaderParam::style = INDICATIN_TEXT_LINE_ARR, то динамический массив - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк обозначения позиции).
2. Каждая строка соответствует одному номеру позиции.

   Смотрите также:

   ksTextLineParam

   ksChar255

Линия – выноска для обозначения параметров клеймения (Интерфейс ksBrandLeaderParam)

Интерфейс параметров линии-выноски для обозначения клеймения.

Аналог данных параметров при использовании API экспортных функций - BrandLeaderParam.

Примечания:

1. Если cText0 = 0 или cText1 = 0 или cText2 = 0, то соответствующий текст на линии-выноске отсутствует.
2. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

   Смотрите также: KompasObject

ksBrandLeaderParam – свойства

arrowType – Тип знака на "ножке" линии-выноски обозначения клеймения

Интерфейс..

Тип данных: short

Значения свойства:

0	- знак отсутствует,
1	- точка,
2	- стрелка.

Синтаксис Automation:

arrowType = iBrandLeaderParam.arrowType	Получить свойство (*)
iBrandLeaderParam.arrowType = arrowType	Установить свойство (*)
arrowType = iBrandLeaderParam.GetArrowType()	Получить свойство (**)
iBrandLeaderParam.SetArrowType(arrowType)	Установить свойство (**)

cText0 – Количество строк текста в знаке клеймения (не более одной строки)

Интерфейс..

Тип данных: short

Синтаксис Automation:

cText0 = iBrandLeaderParam.cText0	Получить свойство (*)
iBrandLeaderParam.cText0 = cText0	Установить свойство (*)
cText0 = iBrandLeaderParam.GetCText0()	Получить свойство (**)
iBrandLeaderParam.SetCText0(cText0)	Установить свойство (**)

cText1 – Количество строк текста над "ножкой" (не более одной строки)

Интерфейс..

Тип данных: short

Синтаксис Automation:

cText1 = iBrandLeaderParam.cText1	Получить свойство (*)
iBrandLeaderParam.cText1 = cText1	Установить свойство (*)
cText1 = iBrandLeaderParam.GetCText1()	Получить свойство (**)
iBrandLeaderParam.SetCText1(cText1)	Установить свойство (**)

cText2 – Количество строк текста под "ножкой" (не более одной строки)

Интерфейс..

Тип данных: short

Синтаксис Automation:

cText2 = iBrandLeaderParam.cText2	Получить свойство (*)
iBrandLeaderParam.cText2 = cText2	Установить свойство (*)
cText2 = iBrandLeaderParam.GetCText2()	Получить свойство (**)
iBrandLeaderParam.SetCText2(cText2)	Установить свойство (**)

dirX – Направление полки относительно оси OX

Интерфейс..

Тип данных: long

Значения свойства:

1	- вправо (по оси OX),

-1 - влево (против оси OX).

Синтаксис Automation:

dirX = iBrandLeaderParam.dirX	Получить свойство (*)
iBrandLeaderParam.dirX = dirX	Установить свойство (*)
dirX = iBrandLeaderParam.GetDirX()	Получить свойство (**)
iBrandLeaderParam.SetDirX(dirX)	Установить свойство (**)

style1 – Стиль текста в знаке клеймения

Интерфейс..

Тип данных: long

Значения свойства:

номер системного стиля, номер пользовательского стиля, INDICATIN_TEXT_LINE_ARR	- в этом случае текст задается массивом строк и стиль текста может быть указан индивидуально для каждой строки.

Системные стили текстов...

Синтаксис Automation:

style1 = iBrandLeaderParam.style1	Получить свойство (*)
iBrandLeaderParam.style1 = style1	Установить свойство (*)
style1 = iBrandLeaderParam.GetStyle1()	Получить свойство (**)
iBrandLeaderParam.SetStyle1(style1)	Установить свойство (**)

Примечание:

Если style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив текстов - типа TEXT_LINE_ARR. В остальных случаях - типа CHAR_STR_ARR.

style2 – Стиль текстов у "ножки" знака клеймения

Интерфейс..

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Системные стили текстов...

Синтаксис Automation:

style2 = iBrandLeaderParam.style2	Получить свойство (*)
iBrandLeaderParam.style1 = style2	Установить свойство (*)

style2 = iBrandLeaderParam.GetStyle2() Получить свойство (**)
iBrandLeaderParam.SetStyle2(style2) Установить свойство (**)

Примечание:

Используется, если style1 не равен INDICATIN_TEXT_LINE_ARR.

x, y – Координаты базовой точки обозначения клеймения

Интерфейс..

Тип данных: double

Синтаксис Automation:

x = iBrandLeaderParam.x Получить свойство (*)
iBrandLeaderParam.x = x Установить свойство (*)
x = iBrandLeaderParam.GetX() Получить свойство (**)
iBrandLeaderParam.SetX(x) Установить свойство (**)

Примечание:

Базовая точка обозначения клеймения - точка выхода из знака "ножки".

ksBrandLeaderParam – методы

GetpPolyline – Получить указатель на интерфейс динамического массива ответвлений линии-выноски обозначения клеймения ksDynamicArray типа POLYLINE_ARR

Интерфейс..

Получить указатель на интерфейс динамического массива ответвлений линии-выноски обозначения клеймения ksDynamicArray типа POLYLINE_ARR.

Синтаксис Automation:

LPDISPATCH GetpPolyline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_AR.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива текста обозначения клеймения ksDynamicArray

Интерфейс..

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray.

Примечания:

1. Если свойство ksBrandLeaderParam::style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста клеймения).
2. Различают три разных текста для обозначения клеймения, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текста над "ножкой". Если свойство cText1 = 0, то этот текст отсутствует;
- текст под "ножкой". Если свойство cText2 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Init – Инициализировать параметры

Интерфейс..

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pText типа CHAR_STR_ARR.
3. Создается динамический массив pPolyline типа POLYLINE_ARR.

SetpPolyline – Установить указатель на интерфейс динамического массива ответвлений линии-выноски обозначения клеймения ksDynamicArray типа POLYLINE_ARR

Интерфейс..

Синтаксис Automation:

BOOL SetpPolyline (LPDISPATCH pPolyline);

Входной параметр:

pPolyline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива обозначения клеймения ksDynamicArray

Интерфейс..

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline	- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_AR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Если свойство ksBrandLeaderParam::style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив ksDynamicArray - типа TEXT_LINE_ARR (динамический массив строк текста),

в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста клеймения).

2. Различают три разных текста для обозначения клеймения, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текста над "ножкой". Если свойство cText1 = 0, то этот текст отсутствует;
- текст под "ножкой". Если свойство свойство cText2 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Линия – выноска для обозначения маркировки (Интерфейс ksMarkerLeaderParam)

Интерфейс параметров линии-выноски для обозначения маркировки.

Аналог данных параметров при использовании API экспортных функций - MarkerLeaderParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksMarkerLeaderParam – свойства

arrowType – Тип указателя линии-выноски

Интерфейс...

Тип данных: short

Значения свойства:

0	- указатель отсутствует,
1	- точка,
2	- стрелка.

Синтаксис Automation:

arrowType = iMarkerLeaderParam.arrowType	Получить свойство (*)
iMarkerLeaderParam.arrowType = arrowType	Установить свойство (*)
arrowType = iMarkerLeaderParam.GetArrowType()	Получить свойство (**)
iMarkerLeaderParam.SetArrowType(arrowType)	Установить свойство (**)

cText0 – Количество строк текста в знаке маркировки (не более одной строки)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText0 = iMarkerLeaderParam.cText0	Получить свойство (*)
iMarkerLeaderParam.cText0 = cText0	Установить свойство (*)
cText0 = iMarkerLeaderParam.GetCText0()	Получить свойство (**)
iMarkerLeaderParam.SetCText0(cText0)	Установить свойство (**)

cText1 – Количество строк текста над "ножкой" линии-выноски обозначения маркировки (не более одной строки)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText1 = iMarkerLeaderParam.cText1	Получить свойство (*)
iMarkerLeaderParam.cText1 = cText1	Установить свойство (*)
cText1 = iMarkerLeaderParam.GetCText1()	Получить свойство (**)
iMarkerLeaderParam.SetCText1(cText1)	Установить свойство (**)

cText2 – Количество строк текста под "ножкой" линии-выноски обозначения маркировки (не более одной строки)

Интерфейс...

Тип данных: short

Синтаксис Automation:

cText2 = iMarkerLeaderParam.cText2	Получить свойство (*)
iMarkerLeaderParam.cText2 = cText2	Установить свойство (*)
cText2 = iMarkerLeaderParam.GetCText2()	Получить свойство (**)
iMarkerLeaderParam.SetCText2(cText2)	Установить свойство (**)

style1 – Стиль текста в знаке маркировки

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,

номер пользовательского стиля,
INDICATIN_TEXT_LINE_ARR - в этом случае текст задается массивом строк и стиль текста может быть указан индивидуально для каждой строки.

Системные стили текстов...

Синтаксис Automation:

style1 = iMarkerLeaderParam.style1	Получить свойство (*)
iMarkerLeaderParam.style1 = style1	Установить свойство (*)
style1 = iMarkerLeaderParam.GetStyle1()	Получить свойство (**)
iMarkerLeaderParam.SetStyle1(style1)	Установить свойство (**)

Примечание:

Если style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив текстов - типа TEXT_LINE_ARR. В остальных случаях - типа CHAR_STR_ARR.

style2 – Стиль текстов у "ножки" линии-выноски обозначения маркировки

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Системные стили текстов...

Синтаксис Automation:

style2 = iMarkerLeaderParam.style2	Получить свойство (*)
iMarkerLeaderParam.style2 = style2	Установить свойство (*)
style2 = iMarkerLeaderParam.GetStyle2()	Получить свойство (**)
iMarkerLeaderParam.SetStyle2(style2)	Установить свойство (**)

Примечание:

Используется, если style1 не равен INDICATIN_TEXT_LINE_ARR.

x, y – Координаты базовой точки линии-выноски обозначения маркировки

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iMarkerLeaderParam.x	Получить свойство (*)

iMarkerLeaderParam.x = x	Установить свойство (*)
x = iMarkerLeaderParam.GetX()	Получить свойство (**)
iMarkerLeaderParam.SetX(x)	Установить свойство (**)

Примечание:

Базовая точка линии-выноски - точка выхода "ножки" из знака маркировки.

ksMarkerLeaderParam – методы

GetpPolyline – Получить указатель на интерфейс динамического массива ответвлений линии–выноски обозначения маркировки ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpPolyline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также

ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива текста обозначения маркирования ksDynamicArray

Интерфейс...

Получить указатель на интерфейс динамического массива текста обозначения маркирования ksDynamicArray.

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray.

Примечания:

1. Если свойство ksMarkerLeaderParam::style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив ksDynamicArray - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста маркировки).
2. Различают три разных текста для обозначения маркировки, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текста над "ножкой". Если свойство cText1 = 0, то этот текст отсутствует;
- текст под "ножкой". Если свойство cText2 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Init- Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pText типа CHAR_STR_ARR.
3. Создается динамический массив pPolyline типа POLYLINE_ARR.

SetpPolyline – Установить указатель на интерфейс динамического массива ответвлений линии-выноски обозначения маркировки ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpPolyline (LPDISPATCH pPolyline);

Входной параметр:

pPolyline - указатель на интерфейс динамического массива ksDynamicArray типа POLYLINE_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также

ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива текста обозначения маркировки ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline - указатель на интерфейс динамического массива ksDynamicArray.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Если свойство ksMarkerLeaderParam::style1 = INDICATIN_TEXT_LINE_ARR, то динамический массив ksDynamicArray - типа TEXT_LINE_ARR (динамический массив строк текста), в остальных случаях - типа CHAR_STR_ARR (динамический массив строк символов текста маркировки).
2. Различают три разных текста для обозначения маркировки, их строки лежат в следующей последовательности:

- текст над полкой. Если свойство cText0 = 0, то этот текст отсутствует;
- текста над "ножкой". Если свойство cText1 = 0, то этот текст отсутствует;
- текст под "ножкой". Если свойство cText2 = 0, то этот текст отсутствует.

Смотрите также:

ksTextLineParam

ksChar255

Линия разреза/сечения (Интерфейс ksCutLineParam)

Интерфейс параметров линии разреза/сечения.

Аналог данных параметров при использовании API экспортных функций - CutLineParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksCutLineParam – свойства

right – Положение стрелок относительно линии разреза/сечения

Интерфейс...

Тип данных:short.

Значения свойства:

0 - слева,
1 - справа.

Синтаксис Automation:

right = iCutLineParam.right	Получить свойство (*)
iCutLineParam.right = right	Установить свойство (*)
right = iCutLineParam.GetRight()	Получить свойство (**)
iCutLineParam.SetRight(right)	Установить свойство (**)

str – Текст на линии разреза/сечения

Интерфейс...

Тип данных: строка

Синтаксис Automation:

str = iCutLineParam.str	Получить свойство (*)
iCutLineParam.str = str	Установить свойство (*)
str = iCutLineParam.GetStr()	Получить свойство (**)
iCutLineParam.SetStr(str)	Установить свойство (**)

Примечание:

Свойство используется при ksCutLineParam::type = 0 (т.е. при способе задания надписи на линии в виде строки).

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Системные стили текстов...

Синтаксис Automation:

style = iCutLineParam.style	Получить свойство (*)
iCutLineParam.style = style	Установить свойство (*)
style = iCutLineParam.GetStyle()	Получить свойство (**)
iCutLineParam.SetStyle(style)	Установить свойство (**)

type – Способ задания надписи на линии разреза/сечения

Интерфейс...

Тип данных: short

Значения свойства:

0	- текст в виде строки (используется значение свойства ksCutLineParam::str),
1	- динамический массив компонент текста (используется массив pTextline).

Синтаксис Automation:

type = iCutLineParam.type	Получить свойство (*)
iCutLineParam.type = type	Установить свойство (*)
type = iCutLineParam.GetType()	Получить свойство (**)
iCutLineParam.SetType(type)	Установить свойство (**)

Смотрите также:

ksCutLineParam::SetpTextline,

ksCutLineParam::GetpTextline

x1, y1 – Координаты надписи у первого участка линии разреза/сечения

Интерфейс...

Тип данных: double

Синтаксис Automation:

x1 = iCutLineParam.x1	Получить свойство (*)
iCutLineParam.x1 = x1	Установить свойство (*)
x1 = iCutLineParam.GetX1()	Получить свойство (**)
iCutLineParam.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты надписи у второго участка линии разреза/сечения

Интерфейс...

Тип данных: double

Синтаксис Automation:

x2 = iCutLineParam.x2	Получить свойство (*)
iCutLineParam.x2 = x2	Установить свойство (*)
x2 = iCutLineParam.GetX2()	Получить свойство (**)
iCutLineParam.SetX2(x2)	Установить свойство (**)

ksCutLineParam – методы

GetpMathPoint – Получить указатель на интерфейс динамического массива точек ломаной линии ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpMathPoint();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа POINT_ARR.

Примечание:

Массив точек ломаной состоит из начальной точки, точек излома и конечной точки.

Смотрите также:

ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray компонент текста типа TEXT_ITEM_ARR.

Примечание:

Массив используется при ksCutLineParam::type = 1 (т.е. при способе задания надписи на линии).

Смотрите также: StructTextItem

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pTextItem типа TEXT_ITEM_ARR.
3. Создается динамический массив pMathPoint типа POINT_ARR.

SetpMathPoint – Установить указатель на интерфейс динамического массива точек ломаной линии ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpMathPoint (LPDISPATCH pMathPoint);

Входной параметр:

pMathPoint - указатель на интерфейс динамического массива ksDynamicArray типа POINT_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Массив точек ломаной состоит из начальной точки, точек излома и конечной точки.

Смотрите также: ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline - указатель на интерфейс динамического массива ksDynamicArray компонент текста типа TEXT_ITEM_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Массив используется при ksCutLineParam::type = 1 (т.е. при способе задания надписи на линии).

Смотрите также: TextItemParam

Допуск формы и расположения поверхностей

Интерфейс ksToleranceBranch

Справка системы КОМПАС...

`KOMPAS.chm::/CM_FORMTOLERANCE.htm`

Интерфейс параметров "опоры" допуска формы.

Аналог данных параметров при использовании API экспортных функций - ToleranceBranch .

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksToleranceBranch – свойства

arrowType – Тип "опоры"

Интерфейс...

Тип данных: short

Значения свойства:

0	- "опора" отсутствует,
1	- "опора"- треугольник,
2	- "опора"- стрелка.

Синтаксис Automation:

arrowType = iToleranceBranch.arrowType	Получить свойство (*)
iToleranceBranch.arrowType = arrowType	Установить свойство (*)
arrowType = iToleranceBranch.GetArrowType()	Получить свойство (**)
iToleranceBranch.SetArrowType(arrowType)	Установить свойство (**)

tCorner – Точка выхода "опоры" из таблицы допуска формы

Интерфейс...

Тип данных: short

Значения свойства

Синтаксис Automation:

tCorner = iToleranceBranch.tCorner	Получить свойство (*)
iToleranceBranch.tCorner = tCorner	Установить свойство (*)
tCorner = iToleranceBranch.GetTCorner()	Получить свойство (**)
iToleranceBranch.SetTCorner(tCorner)	Установить свойство (**)

ksToleranceBranch – методы

GetpMathPoint – Получить указатель на интерфейс динамического массив точек "опоры" ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpMathPoint();

Возвращаемое значение:

- указатель на интерфейс динамического массива математических точек ksDynamicArray типа POINT_ARR.

Примечание:

Массив точек опоры состоит из начальной точки, принадлежащей таблице (ее в массив помещать не нужно, т.к. она определена свойством tCorner), точек излома (они могут отсутствовать) и конечной точки "опоры", указывающей на объект.

Смотрите также: ksMathPointParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pMathPoint типа POINT_ARR.

SetpMathPoint – Установить указатель на интерфейс динамического массива точек "опоры" ksDynamicArray типа POINT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpMathPoint(LPDISPATCH pMathPoint);

Входной параметр:

pMathPoint - указатель на интерфейс динамического массива математических точек ksDynamicArray типа POINT_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Массив точек опоры состоит из начальной точки, принадлежащей таблице (ее в массив помещать не нужно, т.к. она определена свойством tCorner), точек излома (они могут отсутствовать) и конечной точки "опоры", указывающей на объект.

Смотрите также: ksMathPointParam

Интерфейс ksToleranceParam

Справка системы КОМПАС...

KOMPAS.chm::/CM_FORMTOLERANCE.htm

Интерфейс параметров обозначения допуска формы и расположения поверхностей.

Аналог данных параметров при использовании API экспортных функций - ToleranceParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksToleranceParam – свойства

tBase – Положение базовой точки на таблице допуска формы

Интерфейс...

Тип данных: short

Значения свойства...

Синтаксис Automation:

tBase = iToleranceParam.tBase	Получить свойство (*)
iToleranceParam.tBase = tBase	Установить свойство (*)
tBase = iToleranceParam.GetTBase()	Получить свойство (**)
iToleranceParam.SetTBase(tBase)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Системные стили текстов...

Синтаксис Automation:

```
style = iToleranceParam.style                Получить свойство (* )
iToleranceParam.style = style                Установить свойство (* )
style = iToleranceParam.GetStyle()           Получить свойство (**)
iToleranceParam.SetStyle( style )            Установить свойство (**)
```

type – Ориентация таблицы допуска формы

Интерфейс...

Тип данных: short

Значения свойства:

```
0          - горизонтально,
1          - вертикально.
```

Синтаксис Automation:

```
type = iToleranceParam.type                  Получить свойство (* )
iToleranceParam.type = type                  Установить свойство (* )
type = iToleranceParam.GetType()             Получить свойство (**)
iToleranceParam.SetType( type )              Установить свойство (**)
```

x, y – Координаты базовой точки таблицы допуска формы

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
x = iToleranceParam.x                        Получить свойство (* )
iToleranceParam.x = x                        Установить свойство (* )
x = iToleranceParam.GetX()                   Получить свойство (**)
iToleranceParam.SetX( x )                    Установить свойство (**)
```

ksToleranceParam – методы

GetBranchArr – Получить указатель на интерфейс динамического массива опор допуска формы

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetBranchArr();

Возвращаемое значение:

- указатель на интерфейс динамического массива опор допуска формы ksDynamicArray типа TOLERANCEBRANCH_ARR.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив branchArr типа TOLERANCEBRANCH_ARR.

SetBranchArr – Установить указатель на интерфейс динамического массива опор допуска формы

Интерфейс...

Синтаксис Automation:

BOOL SetBranchArr (LPDISPATCH pBranchArr);

Входной параметр:

pBranchArr - указатель на интерфейс динамического массива опор допуска формы ksDynamicArray типа TOLERANCEBRANCH_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Стрелка направления взгляда (Интерфейс ksViewPointerParam)

Справка системы КОМПАС...

`KOMPAS.chm::/CM_VIEWPOINTER.htm`

Интерфейс параметров стрелки направления взгляда.

Аналог данных параметров при использовании API экспортных функций - ViewPointerParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksViewPointerParam – свойства

str – Текст на стрелке направления взгляда

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
str = iViewPointerParam.str                 Получить свойство (* )
iViewPointerParam.str = str                 Установить свойство (* )
str = iViewPointerParam.GetStr()            Получить свойство (**)
iViewPointerParam.SetStr(str)               Установить свойство (**)
```

Примечание:

Свойство используется при ksViewPointerParam::type = 0 (т.е. при способе задания надписи на линии в виде строки).

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Системные стили текстов...

Синтаксис Automation:

```
style = iViewPointerParam.style             Получить свойство (* )
iViewPointerParam.style = style             Установить свойство (* )
style = iViewPointerParam.GetStyle()        Получить свойство (**)
iViewPointerParam.SetStyle(style)           Установить свойство (**)
```

type – Способ задания надписи на стрелке направления взгляда

Интерфейс...

Тип данных: short

Значения свойства:

0	- текст в виде строки (используется Значение свойства ksViewPointerParam),
1	- динамический массив компонент текста (используется массив pTextline).

Синтаксис Automation:

type = iViewPointerParam.type	Получить свойство (*)
iViewPointerParam.type = type	Установить свойство (*)
type = iViewPointerParam.GetType()	Получить свойство (**)
iViewPointerParam.SetType(type)	Установить свойство (**)

Смотрите также

ksViewPointerParam::SetpTextline

ksViewPointerParam::GetpTextline

x1, y1 – Координаты вершины ("острия") стрелки направления взгляда

Интерфейс...

Тип данных: double

Синтаксис Automation:

x1 = iViewPointerParam.x1	Получить свойство (*)
iViewPointerParam.x1 = x1	Установить свойство (*)
x1 = iViewPointerParam.GetX1()	Получить свойство (**)
iViewPointerParam.SetX1(x1)	Установить свойство (**)

x2, y2 – Координаты конечной точки стрелки направления взгляда

Интерфейс...

Тип данных: double

Синтаксис Automation:

x2 = iViewPointerParam.x2	Получить свойство (*)
iViewPointerParam.x2 = x2	Установить свойство (*)
x2 = iViewPointerParam.GetX2()	Получить свойство (**)
iViewPointerParam.SetX2(x2)	Установить свойство (**)

xt, yt – Координаты точки привязки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

xt = iViewPointerParam.xt	Получить свойство (*)
iViewPointerParam.xt = xt	Установить свойство (*)
xt = iViewPointerParam.GetXt()	Получить свойство (**)
iViewPointerParam.SetXt(xt)	Установить свойство (**)

ksViewPointerParam – методы

GetpTextline – Получить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray компонент текста типа TEXT_ITEM_ARR.

Примечание:

Массив используется при ksViewPointerParam::type = 1 (т.е. при способе задания надписи на линии).

Смотрите также

ksTextItemParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Свойство ksViewPointerParam::type = 1.
3. Создается динамический массив pTextItem типа TEXT_ITEM_ARR.

SetpTextline – Установить указатель на интерфейс динамического массива компонент текста ksDynamicArray типа TEXT_ITEM_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline	- указатель на интерфейс динамического массива ksDynamicArray компонент текста типа TEXT_ITEM_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Массив используется при ksViewPointerParam::type = 1 (т.е. при способе задания надписи на линии).

Смотрите также: ksTextItemParam

Выносной элемент (Интерфейс ksRemoteElementParam)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_MAIN_REMOTE_VIEW.htm

Интерфейс параметров объекта "выносной элемент".

Аналог данных параметров при использовании API экспортных функций - RemoteElementParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

ksRemoteElementParam – свойства

signType – Тип значка из LtRemoteElmSignType

Интерфейс...

Тип данных: long

Синтаксис Automation:

signType = iRemoteElementParam.signType	Получить свойство (*)

iRemoteElementParam.signType = signType	Установить свойство (*)
signType = iRemoteElementParam.GetSignType()	Получить свойство (**)
iRemoteElementParam.SetSignType(signType)	Установить свойство (**)

height – Высота выносного элемента (для прямоугольника и скругленного прямоугольника)

Интерфейс...

Тип данных: double

Синтаксис Automation:

height = iRemoteElementParam.height	Получить свойство (*)
iRemoteElementParam.height = height	Установить свойство (*)
height = iRemoteElementParam.GetHeight()	Получить свойство (**)
iRemoteElementParam.SetHeight(height)	Установить свойство (**)

radius – Радиус окружности (для окружности)

Интерфейс...

Тип данных: double

Синтаксис Automation:

radius = iRemoteElementParam.radius	Получить свойство (*)
iRemoteElementParam.radius = radius	Установить свойство (*)
radius = iRemoteElementParam.GetRadius()	Получить свойство (**)
iRemoteElementParam.SetRadius(radius)	Установить свойство (**)

shelfX – Координата x начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

shelfX = iRemoteElementParam.shelfX	Получить свойство (*)
iRemoteElementParam.shelfX = shelfX	Установить свойство (*)
shelfX = iRemoteElementParam.GetShelfX()	Получить свойство (**)
iRemoteElementParam.SetShelfX(shelfX)	Установить свойство (**)

shelfY – Координата y начала полки

Интерфейс...

Тип данных: double

Синтаксис Automation:

shelfY = iRemoteElementParam.shelfY	Получить свойство (*)
iRemoteElementParam.shelfY = shelfY	Установить свойство (*)
shelfY = iRemoteElementParam.GetShelfY()	Получить свойство (**)

iRemoteElementParam.SetShelfY(shelfY)	Установить свойство (**)

shelfDir – Направление полки

Интерфейс...

Тип данных: short

Значения свойства:

1	- вправо,
-1	- влево,
2	- вверх,
3	- вниз.

Синтаксис Automation:

shelfDir = iRemoteElementParam.shelfDir	Получить свойство (*)
iRemoteElementParam.shelfDir = shelfDir	Установить свойство (*)
shelfDir = iRemoteElementParam.GetShelfDir()	Получить свойство (**)
iRemoteElementParam.SetShelfDir(shelfDir)	Установить свойство (**)

smooth – Радиус скругления (для прямоугольника)

Интерфейс...

Тип данных: double

Синтаксис Automation:

iRemoteElementParam.smooth	Получить свойство (*)
iRemoteElementParam.smooth = smooth	Установить свойство (*)
smooth = iRemoteElementParam.GetSmooth()	Получить свойство (**)
iRemoteElementParam.SetSmooth(smooth)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

style = iRemoteElementParam.style	Получить свойство (*)
iRemoteElementParam.style = style	Установить свойство (*)
style = iRemoteElementParam.GetStyle()	Получить свойство (**)
iRemoteElementParam.SetStyle(style)	Установить свойство (**)

Примечание:

Если style == INDICATIN_TEXT_LINE_ARR, создается динамический массив строк текста TEXT_LINE_ARR, структура которого описана в StructTextLineParam. В противном случае создается динамический массив указателей на строки символов CHAR_STR_ARR.

width – Ширина выносного элемента (для прямоугольника и скругленного прямоугольника)

Интерфейс...

Тип данных: double

Синтаксис Automation:

width = iRemoteElementParam.width	Получить свойство (*)
iRemoteElementParam.width = width	Установить свойство (*)
width = iRemoteElementParam.GetWidth()	Получить свойство (**)
iRemoteElementParam.SetWidth(Width)	Установить свойство (**)

x – Координата x центра выносного элемента

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iRemoteElementParam.x	Получить свойство (*)
iRemoteElementParam.x = x	Установить свойство (*)
x = iRemoteElementParam.GetX()	Получить свойство (**)
iRemoteElementParam.SetX(x)	Установить свойство (**)

y – Координата y центра выносного элемента

Интерфейс...

Тип данных: double

Синтаксис Automation:

y = iRemoteElementParam.y	Получить свойство (*)
iRemoteElementParam.y = y	Установить свойство (*)
y = iRemoteElementParam.GetY()	Получить свойство (**)
iRemoteElementParam.SetY(y)	Установить свойство (**)

ksRemoteElementParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init (long style);

Входной параметр:

(long) style	- стиль текста.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Метод обнуляет поля структуры параметров RemoteElementParam, которую обслуживает.
2. Если style == INDICATIN_TEXT_LINE_ARR, то создается динамический массив строк текста TEXT_LINE_ARR. Его структура описывается TextLineParam.
3. Если style == 0, то создается динамический массив указателей на строки символов CHAR_STR_ARR.

GetPText – Получить указатель на интерфейс динамического массива строк текста

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPText();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray.

Примечания:

1. Если style == INDICATIN_TEXT_LINE_ARR, то тип динамического массива будет TEXT_LINE_ARR.
2. Если style == 0, то тип динамического массива будет CHAR_STR_ARR.

SetPText – Установить динамический массив строк текста

Интерфейс...

Синтаксис Automation:

BOOL SetPText (LPDISPATCH strArr)

Входной параметр:

strArr	- указатель на интерфейс динамического массива ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Если style == INDICATIN_TEXT_LINE_ARR, то тип динамического массива будет TEXT_LINE_ARR.
2. Если style == 0, то тип динамического массива будет CHAR_STR_ARR.

Знак изменения (Интерфейс ksChangeLeaderParam)

Интерфейс параметров линии-выноски для обозначения изменения.

Аналог данных параметров при использовании API экспортных функций - ChangeLeaderParam.

Описание.

Интерфейс позволяет задать параметры для создания обозначения изменения (см. функцию ksDocument2D::ksChangeLeader, а также используется для получения и изменения параметров обозначения изменения (см. функции ksDocument2D::ksGetObjParam и ksDocument2D::ksSetObjParam).

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksChangeLeaderParam – свойства

leaderLength – Длина ответвлений

Интерфейс...

Тип данных: double

Синтаксис Automation:

leaderLength = ksChangeLeaderParam.leaderLength	Получить свойство (*)
ksChangeLeaderParam.leaderLength = leaderLength	Установить свойство (*)
leaderLength = ksChangeLeaderParam.GetleaderLength()	Получить свойство (**)
ksChangeLeaderParam.SetleaderLength(dirX)	Установить свойство (**)

Примечание:

Если leaderLength >= 0, длина всех ответвлений одинакова и равна leaderLength.

Если leaderLength < 0 - ответвление на всю длину. Длина для каждого ответвления задается координатами конца ответвления.

signHeight – Высота или диаметр значка

Интерфейс...

Тип данных: double

Синтаксис Automation:

signHeight = ksChangeLeaderParam.signHeight	Получить свойство (*)
ksChangeLeaderParam.signHeight = signHeight	Установить свойство (*)
signHeight = ksChangeLeaderParam.GetsignHeight()	Получить свойство (**)
ksChangeLeaderParam.SetsignHeight(signHeight)	Установить свойство (**)

signType – Тип значка

Интерфейс...

Тип данных: short

Значения свойства:

0	- квадрат,
1	- окружность,
2	- квадратные скобки,
3	- круглые скобки,
4	- угловые скобки.

Синтаксис Automation:

signType = ksChangeLeaderParam.signType	Получить свойство (*)
ksChangeLeaderParam.signType = signType	Установить свойство (*)
signType = ksChangeLeaderParam.GetSignType()	Получить свойство (**)
ksChangeLeaderParam.SetSignType(signType)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля, номер пользовательского стиля.	
INDICATIN_TEXT_LINE_ARR	- в этом случае текст задается массивом строк, и стиль текста может быть указан индивидуально для каждой строки.

Системные стили текстов...

Синтаксис Automation:

style = iChangeLeaderParam.style	Получить свойство (*)
iChangeLeaderParam.style = style	Установить свойство (*)
style = iChangeLeaderParam.GetStyle()	Получить свойство (**)
iChangeLeaderParam.SetStyle(style)	Установить свойство (**)

Примечание:

Если style = INDICATIN_TEXT_LINE_ARR, то динамический массив pText - типа TEXT_LINE_ARR. В остальных случаях - типа CHAR_STR_ARR.

x – Координата X базовой точки обозначения изменения

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = ksChangeLeaderParam.x	Получить свойство (*)
ksChangeLeaderParam.x = x	Установить свойство (*)
x = ksChangeLeaderParam.GetX()	Получить свойство (**)
ksChangeLeaderParam.SetX(x)	Установить свойство (**)

y – Координата Y базовой точки обозначения изменения

Интерфейс...

Тип данных: double

Синтаксис Automation:

y = ksChangeLeaderParam.y	Получить свойство (*)
ksChangeLeaderParam.y = y	Установить свойство (*)
y = ksChangeLeaderParam.GetY()	Получить свойство (**)
ksChangeLeaderParam.SetY(y)	Установить свойство (**)

ksChangeLeaderParam – методы

GetpPolyline – Получить указатель на интерфейс динамического массива ответвлений линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpPolyline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также: ksMathPointParam

GetpTextline – Получить указатель на интерфейс динамического массива строк текста знака изменения ksDynamicArray типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetpTextline();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Смотрите также: ksTextLineParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив pTextline типа TEXT_LINE_ARR.
3. Создается динамический массив pPolytline типа POLYLINE_ARR.

SetpPolyline – Установить указатель на интерфейс динамического массива ответвлений линии-выноски ksDynamicArray типа POLYLINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpPolyline (LPDISPATCH pPolyline);

Входной параметр:

pPolyline - указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

В общем случае одно ответвление - это ломаная линия. Первый узел - базовая точка (ее в массив помещать не нужно, т.к. она общая для всех ответвлений), остальные узлы - изломы (могут отсутствовать), последний узел - конец ответвления (указывает на объект).

Смотрите также:

ksMathPointParam

SetpTextline – Установить указатель на интерфейс динамического массива текста знака изменения ksDynamicArray типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetpTextline (LPDISPATCH pTextline);

Входной параметр:

pTextline	- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также: ksTextLineParam

Технические требования (Интерфейс ksTechnicalDemandParam)

Справка системы КОМПАС...

KOMPAS.chm::/413_Glava47_Tekhnicheskie_trebo.htm

Интерфейс параметров технических требований.

Аналог данных параметров при использовании API экспортных функций - TechnicalDemandParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksTechnicalDemandParam – свойства

strCount – Количество строк в технических требованиях

Интерфейс...

Тип данных: short

Синтаксис Automation:

strCount = iTechnicalDemandParam.strCount	Получить свойство (*)
iTechnicalDemandParam.strCount = strCount	Установить свойство (*)
strCount = iTechnicalDemandParam.GetStrCount()	Получить свойство (**)
iTechnicalDemandParam.SetStrCount(strCount)	Установить свойство (**)

style – Стиль текста

Интерфейс...

Тип данных: long

Системные стили текста...

Синтаксис Automation:

style = iTechnicalDemandParam.style	Получить свойство (*)
iTechnicalDemandParam.style = style	Установить свойство (*)
style = iTechnicalDemandParam.GetStyle()	Получить свойство (**)
iTechnicalDemandParam.SetStyle(style)	Установить свойство (**)

ksTechnicalDemandParam – методы

GetPGab – Получить указатель на интерфейс динамического массива ksDynamicArray параметров габаритных прямоугольников типа RECT_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetPGab();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray параметров габаритных прямоугольников типа RECT_ARR.

Смотрите также: ksRectParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Метод обнуляет все параметры технических требований.

SetPGab – Установить динамический массив параметров ksDynamicArray габаритных прямоугольников типа RECT_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetPGab(LPDISPATCH pGab);

Входной параметр:

pGab	- указатель на интерфейс динамического массива ksDynamicArray параметров габаритных прямоугольников типа RECT_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Знак неуказанной шероховатости(Интерфейс ksSpecRoughParam)

Справка системы КОМПАС...

KOMPAS.chm::/363_41_3_Znak_neukazannoj_shero.htm

Интерфейс параметров знака неуказанной шероховатости.

Аналог данных параметров при использовании API экспортных функций - SpecRoughParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksSpecRoughParam – свойства

s – Текст

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
s = iSpecRoughParam.s                Получить свойство (* )
iSpecRoughParam.s = s                Установить свойство (* )
s = iSpecRoughParam.GetS()           Получить свойство (**)
iSpecRoughParam.SetS( s )            Установить свойство (**)
```

sign – Тип знака неуказанной шероховатости

Интерфейс...

Тип данных: short

Значения свойства:

```
0    - вид обработки не устанавливается,
1    - обработка удалением слоя материала,
2    - обработка без удаления слоя материала.
```

Синтаксис Automation:

```
sign = iSpecRoughParam.sign          Получить свойство (* )
iSpecRoughParam.sign = sign          Установить свойство (* )
sign = iSpecRoughParam.GetSign()     Получить свойство (**)
iSpecRoughParam.SetSign( sign )      Установить свойство (**)
```

style – Номер стиля текста

Интерфейс...

Тип данных: long

Системные стили текста...

Синтаксис Automation:

```
style = iSpecRoughParam.style        Получить свойство (* )
iSpecRoughParam.style = style        Установить свойство (* )
style = iSpecRoughParam.GetStyle()   Получить свойство (**)
iSpecRoughParam.SetStyle (style)     Установить свойство (**)
```

t – Признак наличия знака в скобках

Интерфейс...

Тип данных: BOOL

Значения свойства:

FALSE	- знака нет,
TRUE	- знак есть.

Синтаксис Automation:

t = iSpecRoughParam.t	Получить свойство (*)
iSpecRoughParam.t = t	Установить свойство (*)
t = iSpecRoughParam.GetT()	Получить свойство (**)
iSpecRoughParam.SetT(t)	Установить свойство (**)

ksSpecRoughParam – методы

Init– Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры знака неуказанной шероховатости.

Интерфейс параметров копирования объекта (Интерфейс ksCopyObjectParam)

Справка системы КОМПАС...

`KOMPAS.chm::/CM_COPY_PROPERTY.htm`

Интерфейс параметров копирования объекта графического документа.

Аналог данных параметров при использовании API экспортных функций - CopyObjectParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

ksCopyObjectParam – свойства

angle – Угол поворота в градусах

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
angle = iCopyObjParam.angle                 Получить свойство (* )
iCopyObjParam.angle = angle                 Установить свойство (* )
angle = iCopyObjParam.GetAngle()            Получить свойство (**)
iCopyObjParam.SetAngle(angle)               Установить свойство (**)
```

attrCopy – Признак копирования атрибутов

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - копировать атрибуты,
FALSE       - не копировать атрибуты.
```

Синтаксис Automation:

```
attrCopy = iCopyObjParam.attrCopy               Получить свойство (* )
iCopyObjParam.attrCopy = attrCopy               Установить свойство (* )
attrCopy = iCopyObjParam.GetAttrCopy()          Получить свойство (**)
iCopyObjParam.SetAttrCopy(attrCopy)             Установить свойство (**)
```

dimLineScale – Признак масштабирования выносных линий

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - выносные линии масштабировать,
FALSE       - выносные линии не масштабировать.
```

Синтаксис Automation:

```
dimLineScale = iCopyObjParam.dimLineScale               Получить свойство (* )
iCopyObjParam.dimLineScale = dimLineScale               Установить свойство (* )
dimLineScale = iCopyObjParam.GetDimLineScale()          Получить свойство (**)
iCopyObjParam.SetDimLineScale(dimLineScale)             Установить свойство (**)
```

hyperLinksCopy – Копировать ссылки

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

hyperLinksCopy = Object.hyperLinksCopy	Получить свойство (*)
Object.hyperLinksCopy = hyperLinksCopy	Установить свойство (*)
hyperLinksCopy = Object.GetHyperLinksCopy()	Получить свойство (**)
Object.SetHyperLinksCopy(hyperLinksCopy)	Установить свойство (**)

Значения свойства:

TRUE	- копировать ссылки,
FALSE	- не копировать ссылки.

objRef – Указатель на копируемый объект, группу, вид, слой

Интерфейс...

Тип данных: long

Синтаксис Automation:

objRef = iCopyObjParam.objRef	Получить свойство (*)
iCopyObjParam.objRef = objRef	Установить свойство (*)
objRef = iCopyObjParam.GetObjRef()	Получить свойство (**)
iCopyObjParam.SetObjRef(objRef)	Установить свойство (**)

scale – Масштаб

Интерфейс...

Тип данных: double

Синтаксис Automation:

scale = iCopyObjParam.scale	Получить свойство (*)
iCopyObjParam.scale = scale	Установить свойство (*)
scale = iCopyObjParam.GetScale()	Получить свойство (**)
iCopyObjParam.SetScale(scale)	Установить свойство (**)

spcObjCopy – Копировать объекты спецификации

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

objRef = iCopyObjParam.objRef	Получить свойство (*)
iCopyObjParam.objRef = objRef	Установить свойство (*)
objRef = iCopyObjParam.GetObjRef()	Получить свойство (**)
iCopyObjParam.SetObjRef(objRef)	Установить свойство (**)

Значения свойства:

TRUE	- копировать объекты спецификации,
FALSE	- не копировать объекты спецификации.

storagesCopy – Копировать пользовательские данные и свойства

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

storagesCopy = Object.storagesCopy	Получить свойство (*)
Object.storagesCopy = storagesCopy	Установить свойство (*)
storagesCopy = Object.GetStoragesCopy()	Получить свойство (**)
Object.SetStoragesCopy(storagesCopy)	Установить свойство (**)

Значения свойства:

TRUE	- копировать пользовательские данные и свойства,
FALSE	- не копировать пользовательские данные и свойства.

xNew, yNew – Координаты точки вставки

Интерфейс...

Тип данных: double

Синтаксис Automation:

xNew = iCopyObjParam.xNew	Получить свойство (*)
iCopyObjParam.xNew = xNew	Установить свойство (*)
xNew = iCopyObjParam.GetXNew()	Получить свойство (**)
iCopyObjParam.SetXNew(xNew)	Установить свойство (**)

Примечание:

Значения xNew и yNew задаются в системе координат текущего вида.

xOld, yOld – Координата базовой точки копируемого объекта

Интерфейс...

Тип данных: double

Синтаксис Automation:

xOld = iCopyObjParam.xOld	Получить свойство (*)
iCopyObjParam.xOld = xOld	Установить свойство (*)
xOld = iCopyObjParam.GetXOld()	Получить свойство (**)
iCopyObjParam.SetXOld(xOld)	Установить свойство (**)

Примечание:

Значения xOld и yOld задаются в системе координат текущего вида.

ksCopyObjectParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Итератор (Интерфейс ksIterator)

Интерфейс итератора.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetIterator.

Смотрите также: KompasObject

ksIterator – свойства

reference – Указатель на итератор

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iIterator.reference	Получить свойство (*)
iIterator.reference = ref	Установить свойство (*)
ref = iIterator.GetReference()	Получить свойство (**)

iIterator.SetReference(ref)	Установить свойство (**)

ksIterator – методы

ksCreateAttrIterator – Создать итератор для перебора атрибутов объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CreateAttrIterator.

Синтаксис Automation:

BOOL ksCreateAttrIterator(long obj,

long key1,

long key2,

long key3,

long key4,

double numb);

Входные параметры:

obj	- указатель на объект,
key1, key2, key3, key4	- ключи для поиска по ключам,
numb	- номер типа атрибута для поиска по номеру.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Если obj = 0, то производится перемещение по объектам с заданным атрибутом внутри графического документа.
2. Если obj не равен 0, то объектом может быть объект вида, вид, группа, слой, тогда движение будет происходить по атрибутам этого объекта.
3. Если obj не равен 0, то объектом может быть документ, тогда движение будет происходить по атрибутам документа.
4. Итератор работает с атрибутами активного документа.

ksCreateIterator – Создать итератор для перемещения по объектам документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CreateIterator.

Синтаксис Automation:

BOOL ksCreateIterator (long tipSearch,

long parent);

Входные параметры:

tipSearch	- тип объекта,
parent	- указатель на объект (для движения по группе, внутри макроэлемента, по слою).

Типы объектов и интерфейсы...

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Для перемещения по документам, видам, группам, слоям parent = 0.
2. Передвижение по объектам документа ("навигация") производится в соответствии с условиями, заданными в специальном блоке параметров. Он содержит тип объектов tipSearch, определяющий режим перемещения (например, по видам, по слоям, по всем объектам, по объектам заданного типа) и указатель комплексного объекта (макроэлемента, контура, группы) при перемещении по составляющим его объектам. Итератор "привязан" к конкретному режиму графического редактора (например, документу, виду), поэтому невозможно использовать один и тот же итератор для навигации в разных видах, штампах и т.п. Итератор сохраняет свое действие до окончания сеанса работы с библиотекой. При переходе внутри библиотечной функции под управление КОМПАС-ГРАФИК после возврата значение всех итераторов будет "сброшено".
3. Итератор работает с атрибутами активного документа.

ksCreateQualityIterator – Создать итератор для перемещения по квалитетам

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCreateQualityIterator.

Синтаксис Automation:

BOOL ksCreateQualityIterator (short system,

short withLimitation);

Входные параметры:

system	- система допуска 1 - система отверстия, 0 - система вала,

withLimitation	- признак учета ограничений, наложенных в системе: 0 - без учёта ограничений, 1 - с учётом ограничений.

Возвращаемое значение:

указатель на итератор	- в случае успешного завершения,
0	- в случае неудачи.

ksCreateSpcIterator – Создать итератор для перебора объектов спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CreateSpcIterator.

Синтаксис Automation:

BOOL ksCreateSpcIterator (BSTR nameLib,

long styleNumb,

long spcObjType);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
styleNumb	- номер стиля спецификации в библиотеке,
spcObjType	- тип объектов: 0 - базовые, 1 - вспомогательные, 2 - базовые и вспомогательные из сортированного массива, 3 - все объекты.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Если функция вызывается в графическом документе, то итератор создается для объектов, образованных по описанию, определяемому именем библиотеки стилей спецификации и номером стиля.

В документе-спецификации параметры nameLib и styleNumb не используются (т.к. все объекты в ней создаются по единственному описанию, однозначно определяемому текущим стилем).

ksDeleteIterator – Удалить блок параметров навигации по модели

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteIterator.

Синтаксис Automation:

long ksDeleteIterator();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksMoveAttrIterator – Перемещаться по атрибутам

Интерфейс...

Аналог данного метода при использовании API экспортных функций - MoveAttrIterator.

Синтаксис Automation:

long ksMoveAttrIterator (BSTR ch,

long* pObj);

Входные параметры:

ch	- направление перемещения итератора: F - первый атрибут, N - следующий атрибут,
pObj	- указатель на группу.

Возвращаемое значение:

указатель на атрибут	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если итератор создан для движения по элементам с определенным атрибутом, то pObj - указатель на объект с данным атрибутом.

ksMoveIterator – Переместить итератор (позиционироваться на объекте)

Интерфейс...

Аналог данного метода при использовании API экспортных функций - MoveIterator.

Синтаксис Automation:

long ksMoveIterator (BSTR ch);

Входной параметр:

ch	- направление перемещения итератора: F - первый объект, N - следующий объект.

Возвращаемое значение:

указатель на объект	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция перемещает текущий указатель итератора (блока параметров навигации) на очередной объект модели. Режим позиционирования (тип объектов, участвующих в поиске) определяется при создании итератора методом ksCreateIterator.

При позиционировании на именованную группу она автоматически становится текущей (рабочей). К ней будут относиться операции добавления и исключения объектов (при исключении всех объектов из именованной группы она автоматически удаляется).

ksMoveQualityIterator – Перемещаться по квалитетам

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksMoveQualityIterator.

Синтаксис Automation:

BOOL ksMoveQualityIterator (LPDISPATCH param,

short inMM,

BSTR ch);

Входные параметры:

param	- указатель на интерфейс параметров квалитета ksQualityContensParam,
inMM	- размерность параметров интервала квалитета minLimit, maxLimit, high, low : 1 - миллиметры, 0 - единицы измерения текущего документа.
ch	- направление перемещения итератора: F - предыдущий квалитет, N - следующий квалитет.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Хранение данных некоторого типа (Интерфейс ksLtVariant)

Интерфейс для хранения данных некоторого типа.

Аналог данных параметров при использовании API экспортных функций - LtVariant

Типы данных...

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksLtVariant – свойства

charVal – Символ

Интерфейс...

Тип данных: short

Синтаксис Automation:

charVal = iLtVariant.charVal	Получить свойство (*)
iLtVariant.charVal = charVal	Установить свойство (*)
charVal = iLtVariant.GetCharVal()	Получить свойство (**)
iLtVariant.SetCharVal(charVal)	Установить свойство (**)

doubleVal – Двойное вещественное значение

Интерфейс...

Тип данных: double

Синтаксис Automation:

doubleVal = iLtVariant.doubleVal	Получить свойство (*)
iLtVariant.doubleVal = doubleVal	Установить свойство (*)
doubleVal = iLtVariant.GetDoubleVal()	Получить свойство (**)
iLtVariant.SetDoubleVal(doubleVal)	Установить свойство (**)

floatVal – Вещественное значение

Интерфейс...

Тип данных: float

Синтаксис Automation:

floatVal = iLtVariant.floatVal	Получить свойство (*)
iLtVariant.floatVal = floatVal	Установить свойство (*)
floatVal = iLtVariant.GetFloatVal()	Получить свойство (**)
iLtVariant.SetFloatVal(floatVal)	Установить свойство (**)

intVal – Целое

Интерфейс...

Тип данных: long

Синтаксис Automation:

intVal = iLtVariant.intVal	Получить свойство (*)
iLtVariant.intVal = intVal	Установить свойство (*)
intVal = iLtVariant.GetIntVal()	Получить свойство (**)
iLtVariant.SetIntVal(intVal)	Установить свойство (**)

longVal – Длинное целое

Интерфейс...

Тип данных: long

Синтаксис Automation:

longVal = iLtVariant.longVal	Получить свойство (*)
iLtVariant.longVal = longVal	Установить свойство (*)
longVal = iLtVariant.GetLongVal()	Получить свойство (**)
iLtVariant.SetLongVal(longVal)	Установить свойство (**)

shortVal – Короткое целое

Интерфейс...

Тип данных: short

Синтаксис Automation:

shortVal = iLtVariant.shortVal	Получить свойство (*)
iLtVariant.shortVal = shortVal	Установить свойство (*)
shortVal = iLtVariant.GetShortVal()	Получить свойство (**)
iLtVariant.SetShortVal(shortVal)	Установить свойство (**)

strVal – Строка в 255 символов для типа ltv_Str

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

strVal = iLtVariant.strVal	Получить свойство (*)
iLtVariant.strVal = strVal	Установить свойство (*)
strVal = iLtVariant.GetStrVal()	Получить свойство (**)
iLtVariant.SetStrVal(strVal)	Установить свойство (**)

uCharVal – Значение типа byte (беззнаковый char)

Интерфейс...

Тип данных: short

Синтаксис Automation:

uCharVal = iLtVariant.uCharVal	Получить свойство (*)
iLtVariant.uCharVal = uCharVal	Установить свойство (*)
uCharVal = iLtVariant.GetUCharVal()	Получить свойство (**)
iLtVariant.SetUCharVal(uCharVal)	Установить свойство (**)

uIntVal – Беззнаковое целое

Интерфейс...

Тип данных: long

Синтаксис Automation:

uIntVal = iLtVariant.uIntVal	Получить свойство (*)
iLtVariant.uIntVal = uIntVal	Установить свойство (*)
uIntVal = iLtVariant.GetUIntVal()	Получить свойство (**)
iLtVariant.SetUIntVal(uIntVal)	Установить свойство (**)

valType – Тип хранимого значения

Интерфейс...

Тип данных: short

Типы данных...

Синтаксис Automation:

valType = iLtVariant.valType	Получить свойство (*)
valType = iLtVariant.GetValType()	Получить свойство (**)

wstrVal – Строка в 255 символов для типа ltv_WStr

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

.wstrVal	Получить свойство (*)
iLtVariant.wstrVal = wstrVal	Установить свойство (*)
.GetwstrVall()	Получить свойство (**)
iChar255.SetStr(str)	Установить свойство (**)

Примечание:

Свойство предназначено только для чтения.

ksLtVariant – методы

Init – Инициализировать параметры

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Параметры, определяемые пользователем(Интерфейс ksUserParam)

Интерфейс пользовательских параметров.

Примечания:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Интерфейс используется для передачи массива пользовательских данных:

1. Установить/получить массив значений ячеек таблицы атрибутов.
   Смотрите: ksAttributeParam::SetValues, ksAttributeParam::GetValues.
2. Установить/получить значение ячейки в таблице атрибута.
   Смотрите: ksAttributeObject::ksSetAttrValue, ksAttributeObject::ksSetAttrValue.
3. Установить/получить данные строки в таблице атрибута.
   Смотрите: ksAttributeObject::ksGetAttrRow, ksAttributeObject::ksSetAttrRow.
4. Добавить строку к табличному атрибуту.
   Смотрите: ksAttributeObject::ksAddAttrRow.
5. Установить/получить пользовательские параметры макроэлемента.
   Смотрите: ksDocument2D::ksSetMacroParam , ksDocument2D::ksGetMacroParam.
6. Получить запись из базы данных.
   Смотрите: ksDataBaseObject::ksReadRecord.
7. Изменить значение компоненты в колонке.
   Смотрите: ksSpecification::ksSpcChangeValue.
8. Установить/получить пользовательские параметры компоненты (модели подсборки).
9. Установить/получить пользовательские параметры макроэлемента 3D.
   Смотрите: ksMacro3DDefinition::SetUserParam, ksMacro3DDefinition::GetUserParam.

10. Для поддержки Unicode в интерфейс ksLtVariant добавлено свойство ksLtVariant::wstrVal .В динамический массив для строковых данных требуется добавлять однотипные типы строковых параметров. Они должны быть или все типа ltv_WStr или все типа ltv_Str. Это критично при работе с атрибутами.

    При получении пользовательских параметров требуется использовать тот же тип, что и при записи параметров. При работе с базой данных при создании отношения с помощью функции ksDataBaseObject::ksRChar требуется в массив добавить ksLtVariant с типом ltv_Str. При использовании функции ksDataBaseObject::ksRCharW требуется добавить ksLtVariant с типом ltv_WStr.

11. Одновременное использование методов ksUserParam::userParams и ksUserParam::GetUserArray и ksUserParam::SetUserArray не допускается.

    Смотрите также: KompasObject

ksUserParam – свойства

fileName – Имя файла библиотеки, в которой находится функция редактирования макроэлемента

Пример...

Интерфейс…

Тип данных: строка

Синтаксис Automation:

fileName = iUserParam.fileName	Получить свойство (*)
iUserParam.fileName = fileName	Установить свойство (*)
fileName = iUserParam.GetFileName()	Получить свойство (**)
iUserParam.SetFileName(fileName)	Установить свойство (**)

Примечание:

Свойство используется для сохранения/получения пользовательских параметров макроэлемента.

Смотрите также:

ksDocument2D::ksSetMacroParam

ksDocument2D::ksGetMacroParam

libName – Имя библиотеки, в которой находится функция редактирования макроэлемента

Пример...

Интерфейс…

Тип данных: строка

Синтаксис Automation:

libName = iUserParam.libName	Получить свойство (*)

iUserParam.libName = libName	Установить свойство (*)
libName = iUserParam.GetLibName()	Получить свойство (**)
iUserParam.SetLibName(libName)	Установить свойство (**)

Примечание:

Свойство используется для сохранения/получения пользовательских параметров макроэлемента.

Смотрите также:

ksDocument2D::ksSetMacroParam

ksDocument2D::ksGetMacroParam

number – Номер функции редактирования

Пример...

Интерфейс…

Тип данных: long

Синтаксис Automation:

number = iUserParam.number	Получить свойство (*)
iUserParam.number = number	Установить свойство (*)
number = iUserParam.GetNumber()	Получить свойство (**)
iUserParam.SetNumber(number)	Установить свойство (**)

Примечание:

Свойство используется для сохранения/получения пользовательских параметров макроэлемента.

Смотрите также:

ksDocument2D::ksSetMacroParam

ksDocument2D::ksGetMacroParam

userParams – Безопасный массив SAFEARRAY байтов пользовательских параметров объекта

Интерфейс…

Тип данных: VARIANT

Синтаксис Automation:

userParams = iUserParam.UserParams	Получить свойство (*)
iUserParam.UserParams = userParams	Установить свойство (*)
userParams = iUserParam.GetUserParams()	Получить свойство (**)
iUserParam.SetUserParams	Установить свойство (**)

Значение свойства:

Принимает\возвращает безопасный массив SAFEARRAY типа VT_ARRAY | VT_UI1.

Примечание:

Свойство используется для сохранения/получения пользовательских параметров макроэлемента в виде безопасного массива SAFEARRAY байтов.

Смотрите также

ksUserParam::GetUserArray

ksUserParam::SetUserArray

ksUserParam – методы

GetUserArray – Получить указатель на интерфейс динамического массива ksDynamicArray типа LTVARIANT_ARR

Интерфейс…

Синтаксис Automation:

LPDISPATCH GetUserArray();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа LTVARIANT_ARR.

Примечание.

Для поддержки Unicode в интерфейс ksLtVariant добавлено свойство ksLtVariant::wstrVal. В динамический массив для строковых данных требуется добавлять однотипные типы строковых параметров. Они должны быть или все типа ltv_WStr, или все типа ltv_Str. Это критично при работе с атрибутами.

При получении пользовательских параметров требуется использовать тот же тип, что и при записи параметров. При работе с базой данных при создании отношения с помощью функции ksDataBaseObject::ksRChar требуется в массив добавить ksLtVariant с типом ltv_Str. При использовании функции ksDataBaseObject::ksRCharW требуется добавить ksLtVariant с типом ltv_WStr.

Смотрите также: ksLtVariant

Init – Инициализировать параметры

Интерфейс…

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все пользовательские параметры.
2. Динамический массив не создается.

SetUserArray – Установить динамический массив ksDynamicArray типа LTVARIANT_ARR

Интерфейс…

Синтаксис Automation:

BOOL SetUserArray (LPDISPATCH Array);

Входной параметр:

Array	- указатель на интерфейс динамического массива ksDynamicArray типа LTVARIANT_ARR.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание.

Для поддержки Unicode в интерфейс ksLtVariant добавлено свойство ksLtVariant::wstrVal. В динамический массив для строковых данных требуется добавлять однотипные типы строковых параметров. Они должны быть или все типа ltv_WStr, или все типа ltv_Str. Это критично при работе с атрибутами.

При получении пользовательских параметров требуется использовать тот же тип, что и при записи параметров. При работе с базой данных при создании отношения с помощью функции ksDataBaseObject::ksRChar требуется в массив добавить ksLtVariant с типом ltv_Str. При использовании функции ksDataBaseObject::ksRCharW требуется добавить ksLtVariant с типом ltv_WStr.

Смотрите также: ksLtVariant

Данные типа DOUBLE (Интерфейс ksDoubleValue)

Интерфейс для типа double.

Примечания:

1. Данный интерфейс используется в качестве параметров узла NURBS и в динамическом массиве ksDynamicArray типа DOUBLE_ARR.
2. Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

   Смотрите также: KompasObject

ksDoubleValue – свойства

value – Значение

Интерфейс...

Тип данных: double

Синтаксис Automation:

value = iDoubleValue.value	Получить свойство (*)
iDoubleValue.value = value	Установить свойство (*)
value = iDoubleValue.GetValue()	Получить свойство (**)
iDoubleValue.SetValue(value)	Установить свойство (**)

ksDoubleValue – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры узла NURBS.

Строка до 255 символов (Интерфейс ksChar255)

Интерфейс строки в 255 символов для массива ksDynamicArray типа CHAR_STR_ARR.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksChar25 – свойства

str – Строка

Интерфейс...

Тип данных: строка

Синтаксис Automation:

str = iChar255.str	Получить свойство (*)

iChar255.str = str	Установить свойство (*)
str = iChar255.GetStr()	Получить свойство (**)
iChar255.SetStr(str)	Установить свойство (**)

ksChar25 – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет строку.

Интерфейс параметров параметризации группы объектов (Интерфейсы ksParametrizationParam, IParametrizationParam)

Справка системы КОМПАС...

KOMPAS.chm::/476_Glava55_Nalozhenie_svjazej_.htm

Интерфейс параметров параметризации группы объектов.

ksParametrizationParam	- интерфейс Automation
IParametrizationParam	- интерфейс COM

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

ksParametrizationParam – свойства

angleLimit – Угловой допуск, градусов

Интерфейс...

Тип данных: double

Синтаксис Automation:

angleLimit = iParametrizationParam.angleLimit	Получить свойство (*)
iParametrizationParam.angleLimit = angleLimit	Установить свойство (*)
angleLimit = iParametrizationParam.GetAngleLimit()	Получить свойство (**)
iParametrizationParam.SetAngleLimit (angleLimit)	Установить свойство (**)

Примечание:

Значение должно быть в интервале [0...10] градусов.

horizontal – Горизонтальность

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

horizontal = iParametrizationParam.horizontal	Получить свойство (*)
iParametrizationParam.horizontal = horizontal	Установить свойство (*)
horizontal = iParametrizationParam.GetHorizontal()	Получить свойство (**)
iParametrizationParam.SetHorizontal (horizontal)	Установить свойство (**)

Примечание:

1. Если horizontal = TRUE, то на горизонтальные отрезки в группе накладывается ограничения на горизонтальность.
2. Отклонение от горизонтали angleLimit должно быть в интервале [0..10] градусов.

nearestPoints – Совпадение точек

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

nearestPoints = iParametrizationParam.nearestPoints	Получить свойство (*)
iParametrizationParam.nearestPoints = nearestPoints	Установить свойство (*)
nearestPoints = iParametrizationParam.GetNearestPoints()	Получить свойство (**)
iParametrizationParam.SetNearestPoints(nearestPoints)	Установить свойство (**)

Примечание:

1. Если nearestPoints = TRUE, то при совпадении точек у объектов в параметризуемой группе на объекты накладывается ограничения на совпадение точек.
2. Параметр используется совместно с pointsLimit.

parallel – Параллельность

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

parallel = iParametrizationParam.parallel	Получить свойство (*)
iParametrizationParam.parallel = parallel	Установить свойство (*)
parallel = iParametrizationParam.GetParallel()	Получить свойство (**)
iParametrizationParam.SetParallel (parallel)	Установить свойство (**)

Примечание:

1. Если parallel = TRUE, то на параллельные отрезки в группе накладывается ограничения на параллельность.
2. Допуск на отклонение от параллельности angleLimit должно быть в интервале [0..10] градусов.

perpendicular – Перпендикулярность

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

perpendicular = iParametrizationParam.perpendicular	Получить свойство (*)
iParametrizationParam.perpendicular = perpendicular	Установить свойство (*)
perpendicular = iParametrizationParam.GetPerpendicular()	Получить свойство (**)
iParametrizationParam.SetPerpendicular (perpendicular)	Установить свойство (**)

Примечание:

1. Если perpendicular = TRUE, то на параллельные отрезки в группе накладывается ограничения на перпендикулярность.
2. Допуск на отклонение от перпендикулярности angleLimit должно быть в интервале [0...10] градусов.

pointsLimit – Допуск на совпадение точек, мм

Интерфейс...

Тип данных: double

Синтаксис Automation:

pointsLimit = iParametrizationParam.pointsLimit	Получить свойство (*)
iParametrizationParam.pointsLimit = pointsLimit	Установить свойство (*)
pointsLimit = iParametrizationParam.GetPointsLimit()	Получить свойство (**)
iParametrizationParam.SetPointsLimit (pointsLimit)	Установить свойство (**)

Примечание:

1. Значение должно быть в интервале [0...10] мм.
2. Параметр используется совместно с nearestPoints. Если nearestPoints = TRUE, то при совпадении точек у объектов в параметризуемой группе на объекты накладываются ограничения на совпадение точек.

vertical – Вертикальность

Интерфейс...

Тип данных: VARIANT_BOOL

Синтаксис Automation:

```
vertical = iParametrizationParam.vertical                  Получить свойство (* )
iParametrizationParam.vertical = vertical                  Установить свойство (* )
vertical = iParametrizationParam.GetVertical()             Получить свойство (**)
iParametrizationParam.SetVertical (vertical)               Установить свойство (**)
```

Примечание:

1. Если vertical = TRUE, то на вертикальные отрезки в группе накладываются ограничения на горизонтальность.
2. Отклонение от вертикали angleLimit должно быть в интервале [0..10] градусов.

ksParametrizationParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис:

VARIANT_BOOL Init();

Интерфейс информации о текущей привязке (Интерфейс ISnapInfo)

Интерфейс информации о текущей привязке.

Примечание.

Интерфейс позволяет получить информацию о привязке к объектам документа, сетке документа и о других типах привязки в процессах CursorEx, PlacementEx.

Типы возможных привязок см. в перечислении ksSnapTypeEnum.

Получить указатель на интерфейс можно с помощью функции ksGetSnapInfo.

ISnapInfo – методы

GetObject1 – Первый объект

Интерфейс...

Синтаксис COM:

long Get Object1();

Возвращаемое значение:

- Указатель (reference) первого объекта, для которого сработала привязка.

GetObject2 – Второй объект

Интерфейс...

Синтаксис COM:

long Get Object2();

Возвращаемое значение:

- Указатель (reference) второго объекта, для которого сработала привязка. Например, для типа пересечение.

GetPoint – Точка привязки

Интерфейс...

Синтаксис COM:

BOOL GetPoint(double * x, double * y);

Выходные параметры:

x, y - координаты точки привязки.

GetSnapType1 – Тип привязки на первом объекте

Интерфейс...

Синтаксис COM:

long GetSnapType1();

Возвращаемое значение:

- Тип привязки из перечисления ksSnapTypeEnum.

GetSnapType2 – Тип привязки на втором объекте

Интерфейс...

Синтаксис COM:

long GetSnapType2();

Возвращаемое значение:

- Тип привязки из перечисления ksSnapTypeEnum.

Параметрические связи и ограничения (Интерфейс ksConstraintParam)

Справка системы КОМПАС...

KOMPAS.chm::/466_54_1_Chto_takoe_parametrich.htm

Интерфейс параметрических связей и ограничений.

Аналог данных параметров при использовании API экспортных функций - ConstraintParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также: KompasObject

ksConstraintParam – свойства

constrType – Тип параметрического ограничения

Интерфейс...

Тип данных: short

Синтаксис Automation:

constrType = iConstraintParam.constrType	Получить свойство (*)
iConstraintParam.constrType = constrType	Установить свойство (*)
constrType = iConstraintParam.GetConstrType()	Получить свойство (**)
iConstraintParam.SetConstrType(constrType)	Установить свойство (**)

Типы параметрических ограничений...

index – Индекс точки на объекте

Интерфейс...

Тип данных: long

Синтаксис Automation:

index = iConstraintParam.index	Получить свойство (*)
iConstraintParam.index = index	Установить свойство (*)
index = iConstraintParam.GetIndex()	Получить свойство (**)
iConstraintParam.SetIndex(index)	Установить свойство (**)

Примечание:

Индексирование точек начинается с 0, у дуги и окружности 0 - центр.

partner – Указатель на второй объект

Интерфейс...

Тип данных: long

Синтаксис Automation:

partner = iConstraintParam.partner	Получить свойство (*)
iConstraintParam.partner = partner	Установить свойство (*)
partner = iConstraintParam.GetPartner()	Получить свойство (**)
iConstraintParam.SetPartner(partner)	Установить свойство (**)

partnerIndex – Индекс точки на втором объекте

Интерфейс...

Тип данных: long

Синтаксис Automation:

partnerIndex = iConstraintParam.partnerIndex	Получить свойство (*)
iConstraintParam.partnerIndex = partnerIndex	Установить свойство (*)
partnerIndex = iConstraintParam.GetPartnerIndex()	Получить свойство (**)
iConstraintParam.SetPartnerIndex(partnerIndex)	Установить свойство (**)

Примечание:

Индексирование точек начинается с 0, у дуги и окружности 0 - центр.

ksConstraintParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Параметры сохранения растра (Интерфейсы ksRasterFormatParam, IRasterFormatParam)

См. раздел...

Интерфейсы спецификации

Интерфейс ksSpecification

Интерфейс ISpecification3D

Текстовый документ (Интерфейс ksDocumentTxt)

События...

Справка системы КОМПАС...

KOMPAS.chm::/48_2_1_3_Tekstovye_dokumenty.htm

Интерфейс текстового документа.

ksDocumentTxt - интерфейс Automation

Примечание:

Данный интерфейс в автоматизации может быть получен от интерфейса API Kompas KompasObject с помощью методов KompasObject::DocumentTxt, KompasObject::ActiveDocumentTxt;

ksDocumentTxt – свойства

reference – Указатель на текстовый документ системы КОМПАС

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iDocumentTXT.reference	Получить свойство (*)
iDocumentTXT.reference = ref	Установить свойство (*)
ref = iDocumentTXT.GetReference()	Получить свойство (**)
iDocumentTXT.SetReference(ref)	Установить свойство (**)

ksDocumentTxt – методы

GetStamp – Получить указатель на интерфейс основной надписи

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Синтаксис Automation:

LPDISPATCH GetStamp();

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

GetStampEx – Получить указатель на интерфейс основной надписи чертежа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Синтаксис Automation:

LPDISPATCH GetStampEx (long sheetNumb);

Входные параметры:

sheetNumb - номер листа, начиная с 1.

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

GetTxtDocumentPagesCount – Получить количество листов текстового документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetTxtDocumentPagesCount.

Синтаксис Automation:

long ksGetTxtDocumentPagesCount();

Возвращаемое значение:

Количество листов

ksCloseDocument – Закрыть документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CloseDocument.

Синтаксис Automation:

BOOL ksCloseDocument();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ksCreateDocument – Создать текстовый документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CreateTextDocument.

Синтаксис Automation:

BOOL ksCreateDocument (LPDISPATCH par);

Входные параметры:

par - указатель на интерфейс параметров текстового документа ksTextDocumentParam.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Созданный документ становится текущим.

ksGetDocumentPagesCount – Получить количество листов документа

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetDocumentPagesCount.

Синтаксис Automation:

long ksGetDocumentPagesCount();

Возвращаемое значение:

Количество листов документа.

ksGetObjParam – Получить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjParam.

Синтаксис Automation:

long ksGetObjParam (long reference, LPDISPATCH param, long paramType);

Входные параметры:

reference - указатель на объект,
paramType - тип параметров объекта.

Выходные параметры:

param	- указатель на интерфейс параметров.

Типы объектов и интерфейсы...

Возвращаемое значение:

тип объекта	- в случае успеха,
0	- в случае неудачи.

Примечания:

1. Параметр paramType может принимать значения ALLPARAM, SHEET_ALLPARAM, DOCUMENT_STATE.
2. Возвращаемым значением может быть TXT_DOCUMENT_OBJ.
3. Перед применением метода указатель param должен быть задан, например, с помощью метода KompasObject::GetParamStruct.
4. Вызов метода с нулевыми значениями параметров param и parType позволяет получить тип объекта по его reference.

ksOpenDocument – Открыть документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - OpenDocument.

Синтаксис Automation:

BOOL ksOpenDocument (BSTR nameDoc, BOOL regim);

Входные параметры:

nameDoc	- имя документа,
regim	- режим работы с документом:
	0 - видимый,
	1 - невидимый ("слепой").

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

1. Невидимый режим применяется при пакетном (скрытом) формировании документа.
2. Открытый документ автоматически становится текущим.

ksSaveDocument – Сохранить документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SaveDocument.

Синтаксис Automation:

BOOL ksSaveDocument (BSTR fileName);

Входные параметры:

fileName	- полное имя файла.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

ksSaveDocumentEx – Сохранить документ в выбранной версии

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SaveDocumentEx.

Синтаксис Automation:

BOOL ksSaveDocumentEx (BSTR fileName, long saveMode);

Входные параметры:

fileName	- полное имя файла,
saveMode	- версия для сохранения:
	-1 - в предыдущую версию,
	0 - в текущую версию,
	1 - в версию 5.11.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Если fileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.

ksSetObjParam – Установить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetObjParam.

Синтаксис Automation:

long ksSetObjParam (long reference, LPDISPATCH param, long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров объекта,
param	- указатель на интерфейс параметров.

Типы объектов и интерфейсы...

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечания:

1. Параметр paramType может принимать значения ALLPARAM, SHEET_ALLPARAM, DOCUMENT_STATE.
2. Типом объекта может быть TXT_DOCUMENT_OBJ.
3. Перед применением метода указатель param должен быть задан, например, с помощью метода KompasObject::GetParamStruct.

RasterFormatParam – Получить указатель на интерфейс параметров сохранения документа в растровом формате

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm`

Синтаксис Automation:

LPDISPATCH RasterFormatParam();

Синтаксис COM:

LPRASTERFORMATPARAM RasterFormatParam();

Возвращаемое значение:

указатель на интерфейс ksRasterFormatParam	- в случае успеха,
NULL	- в случае неудачи.

SaveAsToRasterFormat – Сохранить документ в растровом формате

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данного метода при использовании API экспортных функций - ksSaveAsToRasterFormat.

Синтаксис Automation:

BOOL SaveAsToRasterFormat (BSTR fileName, LPDISPATCH rasterPar);

Входные параметры:

fileName	- полное имя файла документа,
rasterPar	- указатель на интерфейс ksRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

SaveAsToUncompressedRasterFormat – Сохранить документ в растровый формат без сжатия

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данного метода при использовании API экспортных функций - ksSaveAsToUncompressedRasterFormat.

Синтаксис Automation:

BOOL SaveAsToUncompressedRasterFormat (BSTR fileName, LPDISPATCH rasterPar);

Входные параметры:

fileName	- полное имя файла документа,
rasterPar	- Указатель на интерфейс ksRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE	- в случае успеха,

FALSE - в случае неудачи.

Примечания:

Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

Основная надпись Интерфейс ksStamp

См. Интерфейс ksStamp

Параметры сохранения растра (Интерфейсы ksRasterFormatParam и IRasterFormatParam)

См. Интерфейс ksRasterFormatParam и IRasterFormatParam

Параметры текстового документа (Интерфейс ksTextDocumentParam)

Справка системы КОМПАС...

KOMPAS.chm::/562_Glava67_Tekst_v_grafichesko.htm

Интерфейс параметров текстового документа.

Аналог данных параметров при использовании API экспортных функций - TextDocumentParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода ompasObject::GetParamStruct.

ksTextDocumentParam – свойства

author – Имя автора документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

author = iTextDocumentParam.author	Получить свойство (*)
iTextDocumentParam.author = author	Установить свойство (*)
author = iTextDocumentParam.GetAuthor()	Получить свойство (**)
iTextDocumentParam.SetAuthor(author)	Установить свойство (**)

comment – Комментарий документа

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

comment = iTextDocumentParam.comment	Получить свойство (*)
iTextDocumentParam.comment = comment	Установить свойство (*)
comment = iTextDocumentParam.GetComment()	Получить свойство (**)
iTextDocumentParam.SetComment(comment)	Установить свойство (**)

fileName – Полный путь к файлу с растровым изображением

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

fileName = iTextDocumentParam.fileName	Получить свойство (*)
iTextDocumentParam.fileName = fileName	Установить свойство (*)
fileName = iTextDocumentParam.GetFileName()	Получить свойство (**)
iTextDocumentParam.SetFileName(fileName)	Установить свойство (**)

regime – Режим редактирования документа

Интерфейс...

Тип данных: short

Значения свойства:

0	- видимый,
1	- "слепой".

Синтаксис Automation:

regime = iTextDocumentParam.regime	Получить свойство (*)
iTextDocumentParam.regime = regime	Установить свойство (*)
regime = iTextDocumentParam.GetRegime()	Получить свойство (**)
iTextDocumentParam.SetRegime(regime)	Установить свойство (**)

type – Тип документа

Интерфейс...

Тип данных: short

Синтаксис Automation:

type = iTextDocumentParam.type	Получить свойство (*)
iTextDocumentParam.type = type	Установить свойство (*)
type = iTextDocumentParam.GetType()	Получить свойство (**)
iTextDocumentParam.SetType(type)	Установить свойство (**)

ksTextDocumentParam – методы

GetArrTailSheet – Получить указатель на интерфейс массива оформлений листов заключительной части

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrTailSheet();

Возвращаемое значение:

- указатель на интерфейс ksDynamicArray. Тип массива - LIBRARY_STYLE_ARR.

GetArrTitleSheet – Получить указатель на интерфейс массива оформлений титульных листов

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrTitleSheet();

Возвращаемое значение:

- указатель на интерфейс ksDynamicArray. Тип массива - LIBRARY_STYLE_ARR.

GetEvenSheet – Получить указатель на интерфейс оформления четных листов (имя библиотеки стилей, номер стиля в библиотеке)

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetEvenSheet();

Возвращаемое значение:

- указатель на интерфейс ksLibraryStyleParam.

GetFirstSheet – Получить указатель на интерфейс оформления первого листа (имя библиотеки стилей, номер стиля в библиотеке)

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFirstSheet();

Возвращаемое значение:

- указатель на интерфейс ksLibraryStyleParam.

GetOddSheet – Получить указатель на интерфейс оформления нечетных листов (имя библиотеки стилей, номер стиля в библиотеке)

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetOddSheet();

Возвращаемое значение:

- указатель на интерфейс ksLibraryStyleParam.

GetSheetParam – Получить указатель на интерфейс параметров пользовательского или стандартного листа

Интерфейс...

Синтаксис Automation:

GetSheetParam()

Возвращаемое значение:

- возвращаемое значение: указатель на интерфейс ksSheetSize параметров пользовательского или ksStandartSheet стандартного листа.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае успеха,

FALSE - в случае неудачи.

Интерфейсы параметров элементов текста

Интерфейс ksTextParam

Интерфейс параметров текста.

Аналог данных параметров при использовании API экспортных функций - TextParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksTextParam – методы

GetParagraphParam – Получить указатель на интерфейс параметров параграфа ksParagraphParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetParagraphParam();

Возвращаемое значение:

- указатель на интерфейс параметров параграфа ksParagraphParam.

GetTextLineArr – Получить указатель на интерфейс динамического массива ksDynamicArray строк текста типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTextLineArr();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Смотрите также ksTextLineParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры текста.

SetParagraphParam – Установить параметры параграфа ksParagraphParam

Интерфейс...

Синтаксис Automation:

BOOL SetParagraphParam(LPDISPATCH par);

Входной параметр:

par - указатель на интерфейс параметров параграфа ksParagraphParam.

Возвращаемое значение:

TRUE - в случае удачного завершения.

SetTextLineArr – Заменить динамический массив ksDynamicArray строк текста типа TEXT_LINE_ARR

Интерфейс...

Синтаксис Automation:

BOOL SetTextLineArr (LPDISPATCH pLineItem);

Входной параметр:

pLineItem - указатель на интерфейс динамического массива ksDynamicArray типа TEXT_LINE_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Смотрите также ksTextLineParam

Интерфейс ksTextLineParam

Интерфейс параметров строки текста.

Аналог данных параметров при использовании API экспортных функций - TextLineParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода ompasObject::GetParamStruct.

Смотрите также KompasObject

ksTextLineParam – свойства

style – Стиль строки текста

Интерфейс...

Тип данных: long

Значения свойства:

номер системного стиля,
номер пользовательского стиля.

Синтаксис Automation:

style = iTextLineParam.style	Получить свойство (*)
iTextLineParam.style = style	Установить свойство (*)
style = iTextLineParam.GetStyle()	Получить свойство (**)
iTextLineParam.SetStyle(style)	Установить свойство (**)

Системные стили текстов...

ksTextLineParam – методы

GetTextItemArr – Получить указатель на интерфейс динамического массива компонент строки текста

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTextItemArr();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа TEXT_ITEM_ARR.

Смотрите также ksTextItemParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры параграфа текста.

SetTextItemArr – Установить указатель на интерфейс динамического массива компонент строки текста

Интерфейс...

Синтаксис Automation:

BOOL SetTextItemArr (LPDISPATCH pTextItem);

Входной параметр:

pTextItem - указатель на интерфейс динамического массива ksDynamicArray типа TEXT_ITEM_ARR.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Смотрите также ksTextItemParam

Интерфейс ksParagraphParam

Справка системы КОМПАС...

KOMPAS.chm::/CM_TEXT.htm

Интерфейс параметров параграфа текста.

Аналог данных параметров при использовании API экспортных функций - ParagraphParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksParagraphParam – свойства

ang – Угол наклона текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
ang = iParagraphParam.ang                  Получить свойство (* )
iParagraphParam.ang = ang                  Установить свойство (* )
ang = iParagraphParam.GetAng()             Получить свойство (**)
iParagraphParam.SetAng( ang )              Установить свойство (**)
```

height – Высота блока форматирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

```
height = iParagraphParam.height            Получить свойство (* )
iParagraphParam.height = height            Установить свойство (* )
height = iParagraphParam.GetHeight()       Получить свойство (**)
iParagraphParam.SetHeight( height )        Установить свойство (**)
```

hFormat – Признак форматирования по горизонтали

Интерфейс...

Тип данных: long

Значения свойства:

```
0                                          - нет форматирования,
1                                          - сужение текста,
2                                          - перенос на другую строку.
```

Синтаксис Automation:

```
hFormat = iParagraphParam.hFormat          Получить свойство (* )
iParagraphParam.hFormat = hFormat          Установить свойство (* )
hFormat = iParagraphParam.GetHFormat()     Получить свойство (**)
iParagraphParam.SetHFormat(hFormat )       Установить свойство (**)
```

style – Стиль текста

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
style = iParagraphParam.style              Получить свойство (* )
iParagraphParam.style = style              Установить свойство (* )
style = iParagraphParam.GetStyle()         Получить свойство (**)
iParagraphParam.SetStyle( style )          Установить свойство (**)
```

Системные стили текстов...

vFormat – Признак форматирования по вертикали

Интерфейс...

Тип данных: long

Значения свойства:

0	- нет форматирования,
-1	- изменение шага строк.

Синтаксис Automation:

vFormat = iParagraphParam.vFormat	Получить свойство (*)
iParagraphParam.vFormat = vFormat	Установить свойство (*)
vFormat = iParagraphParam.GetVFormat()	Получить свойство (**)
iParagraphParam.SetVFormat(vFormat)	Установить свойство (**)

width – Ширина блока форматирования

Интерфейс...

Тип данных: double

Синтаксис Automation:

width = iParagraphParam.width	Получить свойство (*)
iParagraphParam.width = width	Установить свойство (*)
width = iParagraphParam.GetWidth()	Получить свойство (**)
iParagraphParam.SetWidth(width)	Установить свойство (**)

x, y – Координаты точки привязки текста

Интерфейс...

Тип данных: double

Синтаксис Automation:

x = iParagraphParam.x	Получить свойство (*)
iParagraphParam.x = x	Установить свойство (*)
x = iParagraphParam.GetX()	Получить свойство (**)
iParagraphParam.SetX(x)	Установить свойство (**)

ksParagraphParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры параграфа текста.

Интерфейс ksTextItemFont

Справка системы КОМПАС...

KOMPAS.chm::/562_Glava67_Tekst_v_grafichesko.htm

Интерфейс параметров шрифта компоненты строки текста.

Аналог данных параметров при использовании API экспортных функций - TextItemFont.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также

KompasObject

ksTextItemFont – свойства

bitvector – Битовый вектор, определяющий начертание символов (наклон, толщину, подчеркивание, тип составной части – дробь, отклонение и т.д.)

Тип данных: long

Синтаксис Automation:

bitvector = iTextItemFont.bitvector	Получить свойство (*)
iTextItemFont.bitvector = bitvector	Установить свойство (*)
bitvector = iTextItemFont.GetBitVector()	Получить свойство (**)
iTextItemFont.SetBitVector(bitvector)	Установить свойство (**)

Признаки начертания текста...

color – Цвет символов

Тип данных: long

Синтаксис Automation:

color = iTextItemFont.color	Получить свойство (*)
iTextItemFont.color = color	Установить свойство (*)
color = iTextItemFont.GetColor()	Получить свойство (**)
iTextItemFont.SetColor(color)	Установить свойство (**)

fontName – Имя шрифта

Тип данных: строка

Синтаксис Automation:

fontName = iTextItemFont.fontName	Получить свойство (*)
iTextItemFont.fontName = fontName	Установить свойство (*)
fontName = iTextItemFont.GetFontName()	Получить свойство (**)
iTextItemFont.SetFontName(fontName)	Установить свойство (**)

height – Высота шрифта (в миллиметрах)

Тип данных: double

Синтаксис Automation:

height = iTextItemFont.height	Получить свойство (*)
iTextItemFont.height = height	Установить свойство (*)
height = iTextItemFont.GetHeight()	Получить свойство (**)
iTextItemFont.SetHeight(height)	Установить свойство (**)

ksu – Коэффициент сужения символов

Тип данных: double

Синтаксис Automation:

ksu = iTextItemFont.ksu	Получить свойство (*)
iTextItemFont.ksu = ksu	Установить свойство (*)
ksu = iTextItemFont.GetKsu()	Получить свойство (**)
iTextItemFont.SetKsu(ksu)	Установить свойство (**)

ksTextItemFont – методы

GetBitVectorValue – Получить значение битового вектора, определяющего начертание символов

Синтаксис Automation:

BOOL GetBitVectorValue (long bitVector);

Выходной параметр:

bitVector	- значение битового флага.

Признаки начертания текста...

Возвращаемое значение:

TRUE	- если флаги "взведены",
FALSE	- если флаги "сброшены".

Примечание:

В отличие от GetBitVector этот метод позволяет получить значение конкретного флага.

Init – Инициализировать параметры

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

Метод обнуляет все параметры шрифта компоненты строки текста.

SetBitVectorValue – "Взвести" или "сбросить" значение битового вектора

Синтаксис Automation:

BOOL SetBitVectorValue (long bitVector, BOOL state);

Входные параметры:

bitVector	- значение битового вектора,
state	- состояние вектора: TRUE - включить, FALSE - выключить.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

В отличие от SetBitVector этот метод позволяет установить значение конкретного флага.

Интерфейс ksTextItemParam

Справка системы КОМПАС...

KOMPAS.chm::/534_65_9_Specialqnye_vstavki.htm

Интерфейс параметров компоненты строки текста.

Аналог данных параметров при использовании API экспортных функций - TextItemParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода ompasObject::GetParamStruct.

Смотрите также KompasObject

ksTextItemParam – свойства

iSNumb – Номер спецсимвола, символа из произвольного шрифта или тип отрисовки дроби, или выражения типа суммы, или 0

Интерфейс...

Тип данных: long

Синтаксис Automation:

iSNumb = iTextItemParam.iSNumb	Получить свойство (*)
iTextItemParam.iSNumb = iSNumb	Установить свойство (*)
iSNumb = iTextItemParam.GetISNumb()	Получить свойство (**)
iTextItemParam.SetISNumb(iSNumb)	Установить свойство (**)

Примечание.

Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.

s – Массив символов для компоненты текста

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

s = iTextItemParam.s	Получить свойство (*)
iTextItemParam.s = s	Установить свойство (*)
s = iTextItemParam.GetS()	Получить свойство (**)
iTextItemParam.SetS(s)	Установить свойство (**)

type – Тип символа

Интерфейс...

Тип данных: long

Синтаксис Automation:

type = iTextItemParam.type	Получить свойство (*)
iTextItemParam.type = type	Установить свойство (*)
type = iTextItemParam.GetType()	Получить свойство (**)
iTextItemParam.SetType(type)	Установить свойство (**)

ksTextItemParam – методы

GetItemFont – Получить указатель на интерфейс свойств шрифта компоненты строки текста

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetItemFont();

Возвращаемое значение:

- указатель на интерфейс ksTextItemFont.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры компоненты строки текста.

SetItemFont – Установить указатель на интерфейс свойств шрифта компоненты строки текста

Интерфейс...

Синтаксис Automation:

BOOL SetItemFont(LPDISPATCH font);

Входной параметр:

font - указатель на интерфейс свойств шрифта компоненты строки текста ksTextItemFont.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Документ спецификация (Интерфейс ksSpcDocument)

События...

Справка системы КОМПАС...

kompas.chm::/1168_Glava135_Rabota_s_dokument.htm

Интерфейс документа-спецификации.

Примечание:

Указатель на интерфейс можно получить при помощи метода

KompasObject::SpcDocument, KompasObject::SpcActiveDocument

Смотрите также KompasObject

ksSpcDocument - свойства

reference - Указатель на спецификацию

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iSpcDocument.reference	Получить свойство (*)
iSpcDocument.reference = ref	Установить свойство (*)
ref = iSpcDocument.GetReference()	Получить свойство (**)
iSpcDocument.SetReference(ref)	Установить свойство (**)

ksSpcDocument - методы

GetSpecification - Получить указатель на интерфейс для работы с объектами спецификации ksSpecification

Интерфейс...

Синтаксис Automation

LPDISPATCH GetSpecification();

Возвращаемое значение:

- указатель на интерфейс для работы с объектами спецификации ksSpecification.

GetSpcDocumentNotify – Получить источник событий для документа спецификации

Интерфейс...

Синтаксис Automation:

SpcDocumentNotify* GetSpcDocumentNotify();

Возвращаемое значение:

- указатель на интерфейс источника событий SpcDocumentNotify.

GetStamp – Получить указатель на интерфейс основной надписи ksStamp

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetStamp();

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

GetStampEx – Получить указатель на интерфейс основной надписи

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetStampEx (long sheetNumb);

Входные параметры:

sheetNumb - номер листа, начиная с 1.

Возвращаемое значение:

- указатель на интерфейс основной надписи ksStamp.

ksCloseDocument – Закрыть документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CloseDocument.

Синтаксис Automation:

BOOL ksCloseDocument();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Если документ не сохранен, взводится соответствующая ошибка.

ksCreateDocument – Создать документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CreateDocument.

Синтаксис Automation:

BOOL ksCreateDocument (LPDISPATCH par);

Входной параметр:

par - указатель на интерфейс ksDocumentParam параметров документа.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Документ становится текущим (видимым или не видимым).

ksDeleteObj – Удалить объект спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteObj.

Синтаксис Automation:

long ksDeleteObj (long ref);

Входной параметр:

ref - указатель на удаляемый объект.

Возвращаемое значение:

1 - в случае удачного завершения.

ksExistObj – Проверить, существует ли в спецификации указанный объект

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ExistObj.

Синтаксис Automation:

long ksExistObj (long ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- объект существует,
0	- объект не существует.

ksGetObjParam – Получить параметры объекта спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetObjParam.

Синтаксис Automation:

long ksGetObjParam (long reference,

LPDISPATCH param,

long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров,
param	- указатель на интерфейс параметров ksUserParam.

Типы параметров объектов...

Возвращаемое значение:

тип объекта	- в случае удачного завершения,
0	- в случае неудачи.

Типы объектов и интерфейсы...

Примечания:

1. Параметр paramType может принимать значения ALLPARAM, SHEET_ALLPARAM, DOCUMENT_STATE, SPC_TUNING_PARAM.
2. Возвращаемым значением может быть SPC_OBJ или SPC_DOCUMENT_OBJ.
3. Вызов метода с нулевыми значениями параметров param и parType позволяет получить тип объекта по его reference.

ksGetSpcDocumentPagesCount – Получить количество листов спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcDocumentPagesCount.

Синтаксис Automation:

long ksGetSpcDocumentPagesCount();

Возвращаемое значение:

- количество листов спецификации.

ksGetSpcSheetSB – Получить указатель на интерфейс динамического массива листов сборочного чертежа, подключенных к спецификации, – ksDynamicArray типа CHAR_STR_ARR

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksGetSpcSheetSB.

Синтаксис Automation:

LPDISPATCH ksGetSpcSheetSB();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа CHAR_STR_ARR.

ksOpenDocument – Открыть документ-спецификацию

Интерфейс...

Аналог данного метода при использовании API экспортных функций - OpenDocument.

Синтаксис Automation:

BOOL ksOpenDocument (BSTR nameDoc, short regim);

Входные параметры:

name	- полное имя файла открываемого документа,
regim	- режим открытия: 0 - видимый, 1 - "слепой", 3 - видимый без синхронизации со сборочным чертежом, 4 - "слепой" без синхронизации со сборочным чертежом.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Открытый документ становится текущим.

ksSaveDocument – Сохранить документ

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SaveDocument.

Синтаксис Automation:

BOOL ksSaveDocument (BSTR fileName);

Входной параметр:

fileName	- полное имя файла, (NULL - используется имя существующего файла из документа).

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

fileName определяет имя файла при сохранении в режиме Save As...

Если и в документе имя файла NULL - взводится соответствующая ошибка.

ksSaveDocumentEx – Сохранить документ с новым именем файла

Интерфейс...

Синтаксис Automation:

BOOL ksSaveDocumentEx(BSTR FileName, long SaveMode);

Синтаксис COM:

BOOL ksSaveDocumentEx(LPOLESTR FileName, long SaveMode);

Входные параметры:

FileName	- новое имя файла документа,
SaveMode	- версия для сохранения:
	-1 - в предыдущую версию,
	0 - в текущую версию,
	1 - в версию 5.11.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Если FileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.

ksSaveToDXF – Сохранить документ в формате DXF

Интерфейс...

Синтаксис Automation:

BOOL ksSaveToDXF (LPCTSTR DXFFileName);

Входные параметры:

DXFFileName	- имя файла для записи в формате DXF.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

Если FileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.

ksSetObjParam – Установить параметры объекта

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetObjParam.

Синтаксис Automation:

long ksSetObjParam (long reference,

LPDISPATCH param,

long paramType);

Входные параметры:

reference	- указатель на объект,
paramType	- тип параметров,
param	- указатель на интерфейс параметров ksUserParam.

Типы параметров объектов...

Типы объектов и интерфейсы...

Возвращаемое значение:

1	- в случае удачного завершения.

Примечания:

1. Параметр paramType может принимать значения ALLPARAM, SHEET_ALLPARAM, DOCUMENT_STATE, SPC_TUNING_PARAM.
2. Типом объекта может быть SPC_OBJ или SPC_DOCUMENT_OBJ.

ksSetSpcSheetSB – Установить указатель на интерфейс динамического массива листов сборочного чертежа, подключенных к спецификации

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSetSpcSheetSB.

Синтаксис Automation:

long ksSetSpcSheetSB (LPDISPATCH arr);

Входной параметр:

arr	- указатель на интерфейс динамического массива ksDynamicArray типа CHAR_STR_ARR.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

RasterFormatParam – Получить указатель на интерфейс, определяющий параметры записи в растровый формат

Интерфейс...

Синтаксис Automation:

LPDISPATCH RasterFormatParam();

Возвращаемое значение:

указатель на интерфейс ksRasterFormatParam	- в случае успешного завершения,
NULL	- в случае неудачи.

SaveAsToRasterFormat – Сохранить документ в растровом формате

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSaveAsToRasterFormat.

Синтаксис Automation:

BOOL SaveAsToRasterFormat (BSTR fileName,

LPDISPATCH rasterPar);

Входные параметры:

fileName	- полное имя файла документа,
rasterPar	- указатель на интерфейс ksRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

SaveAsToUncompressedRasterFormat – Сохранить документ в растровый формат без сжатия

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksSaveAsToUncompressedRasterFormat.

Синтаксис Automation:

BOOL SaveAsToUncompressedRasterFormat(BSTR fileName, IDispatch* rasterPar));

Входные параметры:

fileName	- полное имя файла документа,

rasterPar - указатель на интерфейс ksRasterFormatParam, определяющий параметры записи в растровый формат.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод позволяет сохранить документ спецификации в растровом формате без сжатия, с заданными свойствами: цветом, форматом (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) и т.п. Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

Подробнее см. описание ksRasterFormatParam.

Источник событий для документа-спецификации (Интерфейс SpcDocumentNotify)

В Automation источник событий для документа-спецификации SpcDocumentNotify можно получить при помощи метода ksSpcDocument::GetSpcDocumentNotify.

В COM источником для подписки является документ-спецификация. См.Интерфейс SpcDocumentNotify

Источник событий для объекта спецификации (Интерфейс ksSpcObjectNotify)

См. Интерфейс ksSpcObjectNotify

Параметры объектов спецификации

Объект спецификации (Интерфейс ksSpcObjParam)

Справка системы КОМПАС...

`KOMPAS.chm::/1138_130_1_Obwekt_specifikacii_.htm`

Интерфейс параметров объекта спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcObjParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также:

KompasObject

ksSpcObjParam – свойства

blockNumber – Номер блока

Интерфейс...

Тип данных: long

Синтаксис Automation:

blockNumber = iSpcObjParam.blockNumber	Получить свойство (*)
iSpcObjParam.blockNumber = blockNumber	Установить свойство (*)
blockNumber = iSpcObjParam.GetBlockNumber()	Получить свойство (**)
iSpcObjParam.SetBlockNumber(blockNumber)	Установить свойство (**)

Примечание:

Нумерация блоков начинается с 0.

draw – Признак отображения объекта в таблице спецификации

Интерфейс...

Тип данных: short

Значения свойства:

1	- показывать объект,
0	- не показывать объект (обычно применяется для последующих одинаковых объектов).

Синтаксис Automation:

draw = iSpcObjParam.draw	Получить свойство (*)
iSpcObjParam.draw = draw	Установить свойство (*)
draw = iSpcObjParam.GetDraw()	Получить свойство (**)
iSpcObjParam.SetDraw(draw)	Установить свойство (**)

first – Признак одинаковых объектов спецификации

Интерфейс...

Тип данных: short

Значения свойства:

1	- первый из одинаковых объектов,
0	- последующие одинаковые объекты.

Синтаксис Automation:

first = iSpcObjParam.first	Получить свойство (*)
first = iSpcObjParam.GetFirst()	Получить свойство (**)

Примечание:

Свойство только для чтения.

firstOnShee0t – Признак объекта, с которого всегда начинается страница

Интерфейс...

Тип данных: short

Значения свойства:

1 - данный объект всегда первый на странице (перед ним вставляется "разрыв страницы"),
0 - объект располагается на странице сразу за предыдущим объектом.

Синтаксис Automation:

firstOnSheet = iSpcObjParam.firstOnSheet	Получить свойство (*)
iSpcObjParam.firstOnSheet = firstOnSheet	Установить свойство (*)
firstOnSheet = iSpcObjParam.GetFirstOnSheet()	Получить свойство (**)
iSpcObjParam.SetFirstOnSheet (firstOnSheet)	Установить свойство (**)

GetDocArr – Получить указатель на интерфейс динамического массива структур параметров документов, подключенных к объекту спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetDocArr();

Возвращаемое значение:

- указатель на динамический массив ksDynamicArray типа DOC_SPCOBJ_ARR.

insFrgType – Признак связи объекта с внешним фрагментом

Интерфейс...

Тип данных: short

Значения свойства:

0 - объект создан непосредственно в документе,
1 - объект "пришел" в документ из вставки фрагмента,
2 - объект, полученный из вставки фрагмента, редактировался пользователем вручную.

Синтаксис Automation:

insFrgType = iSpcObjParam.insFrgType	Получить свойство (*)
insFrgType = iSpcObjParam.GetInsFrgType()	Получить свойство (**)

Примечание:

Свойство только для чтения.

ispoln – Признак объекта – "исполнения"

Интерфейс...

Тип данных: short

Значения свойства:

1	- объект является "исполнением",
0	- обычный объект.

Синтаксис Automation:

ispoln = iSpcObjParam. ispoln	Получить свойство (*)
iSpcObjParam. ispoln = ispoln	Установить свойство (*)
numbSubSection = iSpcObjParam.GetIspoln()	Получить свойство (**)
iSpcObjParam.SetIspoln (ispoln)	Установить свойство (**)

numbSection – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

numbSection = iSpcObjParam.numbSection	Получить свойство (*)
numbSection = iSpcObjParam.GetNumbSection()	Получить свойство (**)

Примечание:

Свойство только для чтения.

numbSubSection – Номер подраздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
numbSubSection = iSpcObjParam.numbSubSection                    Получить свойство (* )
iSpcObjParam.numbSubSection = numbSubSection                    Установить свойство (* )
numbSubSection = iSpcObjParam.GetNumbSubSection()               Получить свойство (**)
iSpcObjParam.SetNumbSubSection(numbSubSection)                  Установить свойство (**)
```

posInc – Признак возрастания номера позиции объекта

Интерфейс...

Тип данных: short

Значения свойства:

1	- номер позиции объекта возрастает,
0	- номер позиции объекта не возрастает (остается таким же, как номер позиции предыдущего объекта).

Синтаксис Automation:

```
posInc = iSpcObjParam.posInc                                    Получить свойство (* )
iSpcObjParam.posInc = posInc                                    Установить свойство (* )
posInc = iSpcObjParam.GetPosInc()                               Получить свойство (**)
iSpcObjParam.SetPosInc(posInc)                                  Установить свойство (**)
```

posNotDraw – Признак отображения номера позиции объекта в таблице спецификации

Интерфейс...

Тип данных: short

Значения свойства:

1	- не показывать номер позиции,
0	- показывать номер позиции.

Синтаксис Automation:

```
posNotDraw = iSpcObjParam.posNotDraw                            Получить свойство (* )
iSpcObjParam.posNotDraw = posNotDraw                            Установить свойство (* )
posNotDraw = iSpcObjParam.GetPosNotDraw()                       Получить свойство (**)
iSpcObjParam.SetPosNotDraw (posNotDraw)                         Установить свойство (**)
```

subSectionName – Имя подраздела

Интерфейс...

Тип данных: строка

Синтаксис Automation:

subSectionName = iSpcObjParam.subSectionName — Получить свойство (*)
subSectionName = iSpcObjParam.GetSubSectionName() — Получить свойство (**)

Примечание:

Свойство только для чтения.

typeObj – Тип строки спецификации

Интерфейс...

Тип данных: long

Синтаксис Automation:

typeObj = iSpcObjParam.typeObj — Получить свойство (*)
iSpcObjParam.typeObj = typeObj — Установить свойство (*)
typeObj = iSpcObjParam.GetTypeObj() — Получить свойство (**)
iSpcObjParam.GetTypeObj(typeObj) — Установить свойство (**)

Типы строк спецификации...

ksSpcObjParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры объекта спецификации.

SetDocArr – Установить указатель на интерфейс динамического массива структур параметров документов, подключенных к объекту спецификации

Интерфейс...

Синтаксис Automation:

BOOL SetDocArr (LPDISPATCH docArr);

Входной параметр:

docArr - указатель на динамический массив ksDynamicArray типа DOC_SPCOBJ_ARR.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Документ, подключенный к объекту спецификации (Интерфейс ksDocAttachedSpcParam)

Справка системы КОМПАС...

KOMPAS.chm::/1189_135_6_3_Podkljuchenie_doku.htm

Интерфейс параметров документа, подключенного к объекту спецификации.

Аналог данных параметров при использовании API экспортных функций - DocAttachedSpcParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также:KompasObject

ksDocAttachedSpcParam – свойства

comment – Комментарий к подключенному документу

Интерфейс...

Тип данных: строка

Синтаксис Automation:

comment = iDocAttachedSpcParam. comment	Получить свойство (*)
iDocAttachedSpcParam. comment = comment	Установить свойство (*)
comment = iDocAttachedSpcParam.GetComment()	Получить свойство (**)
iDocAttachedSpcParam.SetComment(comment)	Установить свойство (**)

fileName – Имя файла подключенного документа

Интерфейс...

Тип данных: строка.

Синтаксис Automation:

fileName = iDocAttachedSpcParam.fileName	Получить свойство (*)
iDocAttachedSpcParam.fileName = fileName	Установить свойство (*)

fileName = iDocAttachedSpcParam.GetFileName()	Получить свойство (**)
iDocAttachedSpcParam.SetFileName(fileName)	Установить свойство (**)

transmit – Признак передачи изменений объекта спецификации в подключенный к нему документ

Интерфейс...

Тип данных: long

Значения свойства:

1	- передавать изменения,
0	- не передавать изменения.

Синтаксис Automation:

transmit = iDocAttachedSpcParam.transmit	Получить свойство (*)
iDocAttachedSpcParam.transmit = transmit	Установить свойство (*)
transmit = iDocAttachedSpcParam.GetTransmit()	Получить свойство (**)
iDocAttachedSpcParam.SetTransmit(transmit)	Установить свойство (**)

ksDocAttachedSpcParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры документа, подключенного к объекту спецификации.

Стиль спецификации (Интерфейс ksSpcStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Интерфейс параметров стиля спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода ompasObject::GetParamStruct.

Смотрите также

ksSpcStyleParam – свойства

layoutName1 – Имя файла библиотеки, в которой хранится оформление для первого листа спецификации

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
layoutName1 = iSpcStyleParam.layoutName1                Получить свойство (* )
layoutName1 = iSpcStyleParam.GetLayoutName1()           Получить свойство (**)
```

Примечание:

Данное свойство предназначено только для чтения.

layoutName2 – Имя файла библиотеки, в которой хранится оформление для последующих листов спецификации

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
layoutName2 = iSpcStyleParam.layoutName2                Получить свойство (* )
layoutName2 = iSpcStyleParam.GetLayoutName2()           Получить свойство (**)
```

Примечание:

Данное свойство предназначено только для чтения.

sectionOn – Признак деления на разделы

Интерфейс...

Тип данных: short

Значения свойства:

```
0          - выключено,
1          - включено.
```

Синтаксис Automation:

```
sectionOn = iSpcStyleParam.sectionOn                    Получить свойство (* )
sectionOn = iSpcStyleParam.GetSectionOn()               Получить свойство (**)
```

Примечание:

Данное свойство предназначено только для чтения.

type – Признак формата листа

Интерфейс...

Тип данных: short

Значения свойства:

0 - стандартный,
1 - пользовательский.

Синтаксис Automation:

type = iSpcStyleParam.type Получить свойство (*)
type = iSpcStyleParam.GetType() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

shtType1 – Номер оформления из библиотеки для первого листа спецификации

Интерфейс...

Тип данных: long

Синтаксис Automation:

shtType1 = iSpcStyleParam.shtType1 Получить свойство (*)
shtType1 = iSpcStyleParam.GetShtType1() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

shtType2 – Номер оформления из библиотеки для последующих листов спецификации

Интерфейс...

Тип данных: long

Синтаксис Automation:

shtType2 = iSpcStyleParam.shtType2 Получить свойство (*)
shtType2 = iSpcStyleParam.GetShtType2() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

variant – Вариант оформления спецификации

Интерфейс...

Тип данных: short

Варианты оформления спецификации...

Синтаксис Automation:

```
variant = iSpcStyleParam.variant                Получить свойство (* )
variant = iSpcStyleParam.GetVariant()           Получить свойство (**)
```

Примечание:

Данное свойство предназначено только для чтения.

ksSpcStyleParam – методы

GetArrAdditionalColumn – Получить указатель на интерфейс динамического массива дополнительных колонок ksDynamicArray типа SPCSTYLECOLUMN_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrAdditionalColumn();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа SPCSTYLECOLUMN_ARR.

Смотрите также ksSpcStyleColumnParam

GetArrColumn – Получить указатель на интерфейс динамического массива колонок ksDynamicArray типа SPCSTYLECOLUMN_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrColumn();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа SPCSTYLECOLUMN_ARR.

Смотрите также ksSpcStyleColumnParam

GetArrSection– Получить указатель на интерфейс динамического массива разделов спецификации ksDynamicArray типа SPCSTYLESEC_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrSection();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа SPCSTYLESEC_ARR.

Смотрите также ksSpcStyleSectionParam

GetSheetParam – Получить указатель на интерфейс параметров листа документа ksStandartSheet или ksSheetSize

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSheetParam();

Возвращаемое значение:

- указатель на интерфейс параметров листа документа ksStandartSheet или ksSheetSize.

Примечание:

Тип возвращаемого интерфейса определяется свойством ksSpcStyleParam::type.

GetTuning – Получить умолчательные настройки, считанные из библиотеки стилей спецификации

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetTuning();

Возвращаемое значение:

- указатель на интерфейс параметров стиля настроек спецификации ksSpcTuningStyleParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив arrColumn типа SPCSTYLECOLUMN_ARR.
3. Создается динамический массив arrAdditionalColumn типа SPCSTYLECOLUMN_ARR.
4. Создается динамический массив arrSection типа SPCSTYLESEC_ARR.

Настройки спецификации (Интерфейс ksSpcTuningStyleParam)

Справка системы КОМПАС...

KOMPAS.chm::/1305_Glava146_Nastrojka_sushche.htm

Интерфейс параметров настроек стиля спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcTuningStyleParam.

Примечание:

Указатель на интерфейс можно получить при помощи методов KompasObject::GetParamStruct, ksSpcStyleParam::GetTuning

Смотрите также KompasObject

ksSpcTuningStyleParam – свойства

blocOnNewPage – Признак размещения блоков

Интерфейс...

Тип данных: short

Значения свойства:

0 - начинать блоки после предыдущих блоков,
1 - начинать блоки с новой страницы.

Синтаксис Automation:

blocOnNewPage = iSpcTuningStyleParam.blocOnNewPage Получить свойство (*)
iSpcTuningStyleParam.blocOnNewPage = blocOnNewPage Установить свойство (*)

blocOnNewPage = iSpcTuningStyleParam.GetBlocOnNewPage()	Получить свойство (**)
iSpcTuningStyleParam.SetBlocOnNewPage(blocOnNewPage)	Установить свойство (**)

countBlock – Количество блоков (в групповой спецификации)

Интерфейс...

Тип данных: short

Синтаксис Automation:

countBlock = iSpcTuningStyleParam.countBlock	Получить свойство (*)
iSpcTuningStyleParam.countBlock = countBlock	Установить свойство (*)
countBlock = iSpcTuningStyleParam.GetCountBlock()	Получить свойство (**)
iSpcTuningStyleParam.SetCountBlock(countBlock)	Установить свойство (**)

countIspoln – Количество исполнений (в групповой спецификации)

Интерфейс...

Тип данных: short

Синтаксис Automation:

countIspoln = iSpcTuningStyleParam.countIspoln	Получить свойство (*)
iSpcTuningStyleParam.countIspoln = countIspoln	Установить свойство (*)
countIspoln = iSpcTuningStyleParam.GetCountIspoln()	Получить свойство (**)
iSpcTuningStyleParam.SetCountIspoln(countIspoln)	Установить свойство (**)

disableEmptyStr – Признак наличия пустых строк вокруг заголовка раздела

Интерфейс...

Тип данных: short

Значения свойства:

0	- пустые строки есть,
1	- пустых строк нет.

Синтаксис Automation:

disableEmptyStr = iSpcTuningStyleParam.disableEmptyStr	Получить свойство (*)
iSpcTuningStyleParam.disableEmptyStr = disableEmptyStr	Установить свойство (*)
disableEmptyStr = iSpcTuningStyleParam.GetDisableEmptyStr()	Получить свойство (**)
iSpcTuningStyleParam.SetDisableEmptyStr(disableEmptyStr)	Установить свойство (**)

disableEmptyBlockStr – Признак наличия пустых строк вокруг заголовка блока

Интерфейс...

Тип данных: short

Значения свойства:

0	- пустые строки есть,
1	- пустых строк нет.

Синтаксис Automation:

disableEmptyBlockStr = iSpcTuningStyleParam.disableEmptyBlockStr	Получить свойство (*)
iSpcTuningStyleParam.disableEmptyBlockStr = disableEmptyBlockStr	Установить свойство (*)
disableEmptyBlockStr = iSpcTuningStyleParam.GetDisableEmptyBlockStr()	Получить свойство (**)
iSpcTuningStyleParam.SetDisableEmptyBlockStr(disableEmptyBlockStr)	Установить свойство (**)

geometryDel – Признак удаления геометрии удаленного объекта спецификации

Интерфейс...

Тип данных: short

Значения свойства:

0	- не удалять геометрию,
1	- удалять геометрию.

Синтаксис Automation:

geometryDel = iSpcTuningStyleParam.geometryDel	Получить свойство (*)
iSpcTuningStyleParam.geometryDel = geometryDel	Установить свойство (*)
geometryDel = iSpcTuningStyleParam.GetGeometryDel()	Получить свойство (**)
iSpcTuningStyleParam.SetGeometryDel(geometryDel)	Установить свойство (**)

grToSP – Признак связи сборочного чертежа со спецификацией

Интерфейс...

Тип данных: short

Значения свойства:

0 - нет связи,
1 - только вставка объектов в спецификации,
2 - связь с расчетом позиций.

Синтаксис Automation:

grToSP = iSpcTuningStyleParam.grToSP — Получить свойство (*)
iSpcTuningStyleParam.grToSP = grToSP — Установить свойство (*)
grToSP = iSpcTuningStyleParam.GetGrToSP() — Получить свойство (**)
iSpcTuningStyleParam.SetGrToSP(grToSP) — Установить свойство (**)

insertDash – Признак автоматически формируемого номера исполнения

Интерфейс...

Тип данных: short

Значения свойства:

0 - не вставлять тире перед числом,
1 - вставлять тире перед числом.

Синтаксис Automation:

insertDash = iSpcTuningStyleParam.insertDash — Получить свойство (*)
iSpcTuningStyleParam.insertDash = insertDash — Установить свойство (*)
insertDash = iSpcTuningStyleParam.GetInsertDash() — Получить свойство (**)
iSpcTuningStyleParam.SetInsertDash(insertDash) — Установить свойство (**)

insertNull – Признак автоматически формируемого номера исполнения

Интерфейс...

Тип данных: short

Значения свойства:

0 - не вставлять нули перед числом,
1 - вставлять нули перед числом.

Синтаксис Automation:

insertNull = iSpcTuningStyleParam.insertNull — Получить свойство (*)
iSpcTuningStyleParam.insertNull = insertNull — Установить свойство (*)
insertNull = iSpcTuningStyleParam.GetInsertNull() — Получить свойство (**)
iSpcTuningStyleParam.SetInsertNull(insertNull) — Установить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ispolnMarkFull – Признак отображения номеров исполнений объектов

Интерфейс...

Тип данных: short

Значения свойства:

0	- показывать только номер исполнения,
1	- показывать обозначение исполнения полностью.

Синтаксис Automation:

ispolnMarkFull = iSpcTuningStyleParam.ispolnMarkFull	Получить свойство (*)
iSpcTuningStyleParam.ispolnMarkFull = ispolnMarkFull	Установить свойство (*)
ispolnMarkFull = iSpcTuningStyleParam.GetIspolnMarkFull()	Получить свойство (**)
iSpcTuningStyleParam.SetIspolnMarkFull(ispolnMarkFull)	Установить свойство (**)

ispolnOn – Признак создания исполнений объектов в спецификации

Интерфейс...

Тип данных: short

Значения свойства:

0	- создание исполнений объектов запрещено,
1	- создание исполнений объектов разрешено.

Синтаксис Automation:

ispolnOn = iSpcTuningStyleParam.ispolnOn	Получить свойство (*)
iSpcTuningStyleParam.ispolnOn = ispolnOn	Установить свойство (*)
ispolnOn = iSpcTuningStyleParam.GetIspolnOn()	Получить свойство (**)
iSpcTuningStyleParam.SetIspolnOn(ispolnOn)	Установить свойство (**)

showInfoByDetBlock – Признак, определяющий порядок представления информации в групповой спецификации

Интерфейс...

Тип данных: short

Значения свойства:

0	- выдавать информацию блоками,

1 - выдавать информацию по объектам.

Синтаксис Automation:

showInfoByDetBlock = iSpcTuningStyleParam.showInfoByDetBlock Получить свойство (*)
iSpcTuningStyleParam.showInfoByDetBlock = showInfoByDetBlock Установить свойство (*)
showInfoByDetBlock = Получить свойство (**)
iSpcTuningStyleParam.GetShowInfoByDetBlock()
iSpcTuningStyleParam.SetShowInfoByDetBlock(showInfoByDetBlock) Установить свойство (**)

Примечания:

1. Значение данного свойства учитывается, только если количество исполнений больше, чем количество колонок, предназначенных для записи количества на исполнение.
2. Значение данного свойства используется только для групповой спецификации (вариант Б).

showSectionName – Признак показа имен разделов в таблице

Интерфейс...

Тип данных: short

Значения свойства:

0 - не показывать имена разделов,
1 - показывать имена разделов.

Синтаксис Automation:

showSectionName = iSpcTuningStyleParam.showSectionName Получить свойство (*)
iSpcTuningStyleParam.showSectionName = showSectionName Установить свойство (*)
showSectionName = iSpcTuningStyleParam.GetShowSectionName() Получить свойство (**)
iSpcTuningStyleParam.SetShowSectionName(showSectionName) Установить свойство (**)

positionCalc – Признак расчета номеров позиций

Интерфейс...

Тип данных: short

Значения свойства:

0 - рассчитывать позиции,
1 - не рассчитывать позиции.

Синтаксис Automation:

positionCalc = iSpcTuningStyleParam.positionCalc Получить свойство (*)

iSpcTuningStyleParam.positionCalc = positionCalc	Установить свойство (*)
positionCalc = iSpcTuningStyleParam.GetPositionCalc()	Получить свойство (**)
iSpcTuningStyleParam.SetPositionCalc(positionCalc)	Установить свойство (**)

predefinedTextFileName – Имя файла предопределенных текстов, использующегося при заполнении спецификации

Интерфейс...

Тип данных: строка

Синтаксис Automation:

predefinedTextFileName = iSpcTuningStyleParam.predefinedTextFileName	Получить свойство (*)
iSpcTuningStyleParam.predefinedTextFileName = predefinedTextFileName	Установить свойство (*)
predefinedTextFileName = iSpcTuningStyleParam.GetPredefinedTextFileName()	Получить свойство (**)
iSpcTuningStyleParam.SetPredefinedTextFileName(predefinedTextFileName)	Установить свойство (**)

Примечания:

Файлы текстовых шаблонов имеют расширение *.tdp.

userTextStyle – Стиль текста объектов спецификации

Интерфейс...

Тип данных: short

Значения свойства:

0	- стиль текста из стиля спецификации,
1	- пользовательский стиль текста.

Синтаксис Automation:

userTextStyle = iSpcTuningStyleParam.userTextStyle	Получить свойство (*)
iSpcTuningStyleParam.userTextStyle = userTextStyle	Установить свойство (*)
userTextStyle = iSpcTuningStyleParam.GetUserTextStyle()	Получить свойство (**)
iSpcTuningStyleParam.SetUserTextStyle(userTextStyle)	Установить свойство (**)

zoneCalc – Признак расчета зон

Интерфейс...

Тип данных: short

Значения свойства:

0	- не рассчитывать зоны,
1	- рассчитывать зоны.

Синтаксис Automation:

zoneCalc = iSpcTuningStyleParam.zoneCalc	Получить свойство (*)
iSpcTuningStyleParam.zoneCalc = zoneCalc	Установить свойство (*)
zoneCalc = iSpcTuningStyleParam.GetZoneCalc()	Получить свойство (**)
iSpcTuningStyleParam.SetZoneCalc(zoneCalc)	Установить свойство (**)

ksSpcTuningStyleParam – методы

copySpcObjOnCopyGeometry – Копировать объекты спецификации при копировании геометрии

Интерфейс...

Тип данных: short

Синтаксис Automation:

copySpcObjOnCopyGeometry = iSpcTuningStyleParam.copySpcObjOnCopyGeometry	Получить свойство (*)
iSpcTuningStyleParam.copySpcObjOnCopyGeometry = copySpcObjOnCopyGeometry	Установить свойство (*)
copySpcObjOnCopyGeometry = iSpcTuningStyleParam.GetcopySpcObjOnCopyGeometry()	Получить свойство (**)
iSpcTuningStyleParam.SetcopySpcObjOnCopyGeometry(copy SpcObjOnCopyGeometry)	Установить свойство (**)

GetArrSection – Получить указатель на интерфейс динамического массива структур параметров настроек разделов ksDynamicArray типа SPCTUNINGSEC_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrSection();

Возвращаемое значение:

- указатель на интерфейс ksDynamicArray типа SPCTUNINGSEC_ARR.

Смотрите также ksSpcTuningSectionParam

GetObjectTextStyle – Получить указатель на интерфейс параметров стиля текста объектов спецификации ksTextStyleParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetObjectTextStyle();

Возвращаемое значение:

- указатель на интерфейс ksTextStyleParam.

GetSectionTextStyleFirst – Получить указатель на интерфейс параметров стиля текста первой строки заголовков разделов ksTextStyleParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSectionTextStyleFirst();

Возвращаемое значение:

- указатель на интерфейс ksTextStyleParam.

GetSectionTextStyleNext – Получить указатель на интерфейс параметров стиля текста последующих строк заголовков разделов ksTextStyleParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetSectionTextStyleNext();

Возвращаемое значение:

- указатель на интерфейс ksTextStyleParam.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив arrSection типа SPCTUNINGSEC_ARR.

SetArrSection – Изменить массив настроек разделов для спецификации

Интерфейс...

Синтаксис Automation:

BOOL SetArrSection(IDispatch* arr);

Выходной параметр:

arr	- указатель на интерфейс параметров ksDynamicArray массива настроек разделов для спецификации SpcTuningSectionParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetObjectTextStyle – Изменить стиль текста объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL SetObjectTextStyle(IDispatch* style);

Выходной параметр:

style	- указатель на интерфейс параметров ksTextStyleParam структуры параметров стилей текста заголовка раздела.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetSectionTextStyleFirst – Изменить стиль текста заголовка раздела – первая строка

Интерфейс...

Синтаксис Automation:

BOOL SetSectionTextStyleFirst(IDispatch* style);

Выходной параметр:

style	- указатель на интерфейс параметров ksTextStyleParam структуры параметров стилей текста заголовка раздела.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

SetSectionTextStyleNext – Изменить стиль текста заголовка раздела – следующих строк

Интерфейс...

Синтаксис Automation:

BOOL SetSectionTextStyleNext(IDispatch* style);

Выходной параметр:

style	- указатель на интерфейс параметров ksTextStyleParam структуры параметров стилей текста заголовка раздела.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Описание спецификации (Интерфейс ksSpcDescrParam)

Справка системы КОМПАС...

KOMPAS.chm::/1156_132_4_Opisanie_specifikaci.htm

Интерфейс параметров описания спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcDescrParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода ompasObject::GetParamStruct.

Смотрите также:

KompasObject

ksSpcDescrParam – свойства

layoutName – Имя файла библиотеки стилей

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
layoutName = iSpcDescrParam.layoutName                Получить свойство (* )
iSpcDescrParam.layoutName = layoutName                Установить свойство (* )
layoutName = iSpcDescrParam.GetLayoutName()           Получить свойство (**)
iSpcDescrParam.SetLayoutName(layoutName)              Установить свойство (**)
```

spcName – Имя подключенного файла спецификации

Интерфейс...

Тип данных: строка

Синтаксис Automation:

```
spcName = iSpcDescrParam.spcName                      Получить свойство (* )
iSpcDescrParam.spcName = spcName                      Установить свойство (* )
spcName = iSpcDescrParam.GetSpcName()                 Получить свойство (**)
iSpcDescrParam.SetSpcName( spcName )                  Установить свойство (**)
```

styleId – Номер стиля в библиотеке

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
styleId = iSpcDescrParam.styleId                      Получить свойство (* )
iSpcDescrParam.styleId = styleId                      Установить свойство (* )
styleId = iSpcDescrParam.GetStyleId()                 Получить свойство (**)
iSpcDescrParam.SetStyleId(styleId)                    Установить свойство (**)
```

ksSpcDescrParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры описания спецификации.

Колонка спецификации

Стиль колонки (Интерфейс ksSpcStyleColumnParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс параметров стиля колонки спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcStyleColumnParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksSpcStyleColumnParam – свойства

columnType – Тип колонки спецификации

Интерфейс...

Тип данных: long

Типы колонок спецификации...

Синтаксис Automation:

columnType = iSpcStyleColumnParam.columnType Получить свойство (*)
columnType = iSpcStyleColumnParam.GetColumnType() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

createSum – Признак, указывающий, разрешен ли расчет суммы значений в колонке

Интерфейс...

Тип данных: short

Значения свойства:

0 - не рассчитывать сумму значений в колонке,
1 - рассчитывать сумму значений в колонке.

Синтаксис Automation:

createSum = iSpcStyleColumnParam.createSum Получить свойство (*)
createSum = iSpcStyleColumnParam.GetCreateSum() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

edit – Признак, указывающий, разрешено ли редактирование колонки в данном разделе

Интерфейс...

Тип данных: short

Значения свойства:

0 - не редактировать колонку в данном разделе,
1 - редактировать колонку.

Синтаксис Automation:

edit = iSpcStyleColumnParam.edit Получить свойство (*)
edit = iSpcStyleColumnParam.GetEdit() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ispoln – Номер колонки данного типа

Интерфейс...

Тип данных: long

Синтаксис Automation:

ispoln = iSpcStyleColumnParam.ispoln Получить свойство (*)
ispoln = iSpcStyleColumnParam.GetIspoln() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

linkId – Номер ячейки штампа для связи с данной колонкой

Интерфейс...

Тип данных: long

Синтаксис Automation:

linkId = iSpcStyleColumnParam.linkId — Получить свойство (*)
linkId = iSpcStyleColumnParam.GetLinkId() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

multiplyToCount – Признак, указывающий, разрешено ли умножение при расчете суммы

Интерфейс...

Тип данных: short

Значения свойства:

0 - при расчете суммы не умножать значения в колонке на количество,
1 - умножать значения в колонке на количество.

Синтаксис Automation:

multiplyToCount = iSpcStyleColumnParam.multiplyToCount — Получить свойство (*)
multiplyToCount = iSpcStyleColumnParam.GetMultiplyToCount() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

nameColumn – Имя колонки

Интерфейс...

Тип данных: строка

Синтаксис Automation:

nameColumn = iSpcStyleColumnParam.nameColumn — Получить свойство (*)
nameColumn = iSpcStyleColumnParam.GetNameColumn() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

textDn – Признак выравнивания по вертикали текста в колонке

Интерфейс...

Тип данных: short

Значения свойства:

0 - выравнивать текст в колонке по верхней строке объекта спецификации,
1 - выравнивать текст в колонке по нижней строке.

Синтаксис Automation:

textDn = iSpcStyleColumnParam.textDn Получить свойство (*)
textDn = iSpcStyleColumnParam.GetTextDn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

typeVal –Тип значения в колонке

Интерфейс...

Тип данных: long

Типы данных в колонках...

Синтаксис Automation:

typeVal = iSpcStyleColumnParam.typeVal Получить свойство (*)
typeVal = iSpcStyleColumnParam.GetTypeVal() Получить свойство (**)

Примечание:

1. Тип данных в колонках спецификации может принимать значения: INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.
2. Данное свойство предназначено только для чтения.

useForSectionTitle – Признак размещения имен разделов

Интерфейс...

Тип данных: short

Значения свойства:

0 - не использовать данную колонку для показа имени раздела,
1 - использовать данную колонку для показа имени раздела.

Синтаксис Automation:

useForSectionTitle = iSpcStyleColumnParam.useForSectionTitle Получить свойство (*)
useForSectionTitle = iSpcStyleColumnParam.GetUseForSectionTitle() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ksSpcStyleColumnParam – методы

GetAdditionalParam – Получить указатель на интерфейс дополнительной информации ksRecordTypeAttrParam или ksNumberTypeAttrParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetAdditionalParam();

Возвращаемое значение:

- указатель на интерфейс ksRecordTypeAttrParam или ksNumberTypeAttrParam

Примечание:

Тип возвращаемого интерфейса зависит от типа значения в колонке ksSpcStyleColumnParam::typeVal:

Тип значения	Интерфейс
SPC_RECORD	ksRecordTypeAttrParam
DOUBLE_ATTR_TYPE	ksNumberTypeAttrParam
LINT_ATTR_TYPE	ksNumberTypeAttrParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. По умолчанию свойство typeVal = RECORD_ATTR_TYPE.

Колонка (Интерфейс ksSpcColumnParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс параметров для колонки спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcColumnParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct с параметром ko_SpcColumnParam.

Смотрите также KompasObject

ksSpcColumnParam – свойства

block – Номер блока исполнений

Интерфейс...

Тип данных: long

Синтаксис Automation:

block = iSpcColumnParam.block	Получить свойство (*)
iSpcColumnParam.block = block	Установить свойство (*)
block = iSpcColumnParam.GetBlock()	Получить свойство (**)
iSpcColumnParam.SetBlock(block)	Установить свойство (**)

columnType –Тип колонки

Интерфейс...

Тип данных: long

Синтаксис Automation:

style = iSpcColumnParam.style	Получить свойство (*)
iSpcColumnParam.style = style	Установить свойство (*)
style = iSpcColumnParam.GetStyle()	Получить свойство (**)
iSpcColumnParam.SetStyle(style)	Установить свойство (**)

Типы колонок спецификации...

ispoln – Номер исполнения данного типа, начиная с 1

Интерфейс...

Тип данных: long

Синтаксис Automation:

ispoIn = iSpcColumnParam.ispoIn	Получить свойство (*)
iSpcColumnParam.ispoIn = ispoIn	Установить свойство (*)
ispoIn = iSpcColumnParam.GetIspoIn()	Получить свойство (**)
iSpcColumnParam.SetIspoIn(ispoIn)	Установить свойство (**)

name – Имя колонки

Интерфейс...

Тип данных: строка

Синтаксис Automation:

name = iSpcColumnParam.name	Получить свойство (*)
iSpcColumnParam.name = name	Установить свойство (*)
name = iSpcColumnParam.GetName()	Получить свойство (**)
iSpcColumnParam.SetName(name)	Установить свойство (**)

typeVal – Тип значений в колонке

Интерфейс...

Тип данных: long

Синтаксис Automation:

typeVal = iSpcColumnParam.typeVal	Получить свойство (*)
iSpcColumnParam.typeVal = typeVal	Установить свойство (*)
typeVal = iSpcColumnParam.GetTypeVal()	Получить свойство (**)
iSpcColumnParam.SetTypeVal(typeVal)	Установить свойство (**)

Типы данных в колонках...

Примечание:

Тип данных в колонках спецификации может принимать значения: INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.

ksSpcColumnParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Шаблон записи в колонке (Интерфейс ksRecordTypeAttrParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс дополнительных параметров для типа значения RECORD_ATTR_TYPE в стиле колонки таблицы спецификации.

Аналог данных параметров при использовании API экспортных функций - RecordTypeAttrParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct

Смотрите также KompasObject

ksRecordTypeAttrParam – свойства

attrLibName – Имя файла библиотеки типов атрибутов

Интерфейс...

Тип данных: строка

Синтаксис Automation:

attrLibName = iRecordTypeAttrParam.attrLibName — Получить свойство (*)
attrLibName = iRecordTypeAttrParam.GetAttrLibName() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

key1, key2,key3, key4 – Значения ключей атрибутов, служащих шаблонами заполнения колонки спецификации в данном разделе

Интерфейс...

Тип данных: long.

Синтаксис Automation:

key1 = iRecordTypeAttrParam.key1 — Получить свойство (*)
key1 = iRecordTypeAttrParam.GetKey1() — Получить свойство (**)

Примечания:

1. Данные свойства предназначены только для чтения.
2. Если значение какого-либо ключа 0, то этот ключ не учитывается.

ksRecordTypeAttrParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Диапазон числовых значений в колонке (Интерфейс ksNumberTypeAttrParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_COLUMN_DIALOG.htm

Интерфейс дополнительных параметров для типов значения DOUBLE_ATTR_TYPE и LINT_ATTR_TYPE в стиле колонки таблицы спецификации.

Аналог данных параметров при использовании API экспортных функций - NumberTypeAttrParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksNumberTypeAttrParam – свойства

maxValue – Максимальное значение в колонке спецификации

Интерфейс...

Тип данных: double

Синтаксис Automation:

maxValue = iNumberTypeAttrParam.maxValue	Получить свойство (*)
maxValue = iNumberTypeAttrParam.GetMaxValue()	Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

minValue – Минимальное значение в колонке спецификации

Интерфейс...

Тип данных: double

Синтаксис Automation:

minValue = iNumberTypeAttrParam.minValue Получить свойство (*)
minValue = iNumberTypeAttrParam.GetMinValue() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ksNumberTypeAttrParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Раздел спецификации

Стиль раздела (Интерфейс ksSpcStyleSectionParam)

Справка системы КОМПАС...

KOMPAS.chm::/1143_131_1_Razdely.htm

Интерфейс параметров стиля раздела спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcStyleSectionParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct

Смотрите также KompasObject

ksSpcStyleSectionParam – свойства

dataType – Способ ввода данных в колонку

Интерфейс...

Тип данных: short

Значения свойства:

0 - ручное заполнение колонок,
1 - ручное заполнение или чтение данных из основной надписи подключенного документа.

Синтаксис Automation:

dataType = iSpcStyleSectionParam.dataType — Получить свойство (*)
dataType = iSpcStyleSectionParam.GetDataType() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

number – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

number = iSpcStyleSectionParam.number — Получить свойство (*)
number = iSpcStyleSectionParam.GetNumber() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

sectionName – Имя раздела

Интерфейс...

Тип данных: строка

Синтаксис Automation:

sectionName = iSpcStyleSectionParam.sectionName — Получить свойство (*)
sectionName = iSpcStyleSectionParam.GetSectionName() — Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

sortColumnType – Общий тип колонки, по данным в которой производится сортировка

Интерфейс...

Тип данных: long

Типы колонок спецификации

Синтаксис Automation:

sortColumnType = iSpcStyleSectionParam.sortColumnType Получить свойство (*)
sortColumnType = iSpcStyleSectionParam.GetSortColumnType() Получить свойство (**)

Примечания:

1. Данное свойство предназначено только для чтения.
2. Общий тип формируется из типа колонки и номера колонки этого типа в стиле спецификации.

sortIspoIn – Номер колонки, по данным в которой производится сортировка

Интерфейс...

Тип данных: long

Синтаксис Automation:

sortIspoIn = iSpcStyleSectionParam.sortIspoIn Получить свойство (*)
sortIspoIn = iSpcStyleSectionParam.GetSortIspoIn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

sortType – Тип сортировки объектов в разделе

Интерфейс...

Тип данных: long

Типы сортировки в разделе спецификации...

Синтаксис Automation:

sortType = iSpcStyleSectionParam.sortType Получить свойство (*)
sortType = iSpcStyleSectionParam.GetSortType() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ksSpcStyleSectionParam – методы

GetArrColumn – Получить указатель на интерфейс динамического массива параметров стиля колонок ksDynamicArray типа SPCSTYLECOLUMN_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrColumn();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа SPCSTYLECOLUMN_ARR.

Смотрите также ksSpcStyleColumnParam

GetArrAdditionalColumn – Получить указатель на интерфейс динамического массива параметров стиля дополнительных колонок ksDynamicArray типа SPCSTYLECOLUMN_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrAdditionalColumn();

Возвращаемое значение:

- указатель на интерфейс динамического массива параметров стиля дополнительных колонок ksDynamicArray типа SPCSTYLECOLUMN_ARR.

Смотрите также ksSpcStyleColumnParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры стиля раздела спецификации.

Раздел (Интерфейс ksSpcTuningSectionParam)

Справка системы КОМПАС...

KOMPAS.chm::/1143_131_1_Razdely.htm

Интерфейс параметров настройки раздела спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcTuningSectionParam.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksSpcTuningSectionParam – свойства

firstOnSheet – Признак размещения раздела

Интерфейс...

Тип данных: short

Значения свойства:

0	- начинать раздел после предыдущего раздела,
1	- начинать раздел с новой страницы.

Синтаксис Automation:

firstOnSheet = iSpcTuningSectionParam.firstOnSheet	Получить свойство (*)
firstOnSheet = iSpcTuningSectionParam.GetFirstOnSheet()	Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

geometryOn – Признак подключения геометрии к объектам раздела

Интерфейс...

Тип данных: short

Значения свойства:

0	- подключение геометрии запрещено,
1	- подключение геометрии разрешено.

Синтаксис Automation:

geometryOn = iSpcTuningSectionParam.geometryOn Получить свойство (*)
geometryOn = iSpcTuningSectionParam.GetGeometryOn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

number – Номер раздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

number = iSpcTuningSectionParam.number Получить свойство (*)
number = iSpcTuningSectionParam.GetNumber() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

positionOn – Признак простановки номеров позиций в разделе

Интерфейс...

Тип данных: short

Значения свойства:

0 - не проставлять позиции,
1 - проставлять позиции.

Синтаксис Automation:

positionOn = iSpcTuningSectionParam.positionOn Получить свойство (*)
positionOn = iSpcTuningSectionParam.GetPositionOn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

sortOn – Признак наличия сортировки объектов в разделе

Интерфейс...

Тип данных: short

Значения свойства:

0 - сортировка выключена,
1 - сортировка включена.

Синтаксис Automation:

sortOn = iSpcTuningSectionParam.sortOn Получить свойство (*)
sortOn = iSpcTuningSectionParam.GetSortOn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

subsectionOn – Признак деления на подразделы

Интерфейс...

Тип данных: short

Значения свойства:

0 - деление на подразделы включено,
1 - деление на подразделы выключено.

Синтаксис Automation:

subsectionOn = iSpcTuningSectionParam.subsectionOn Получить свойство (*)
subsectionOn = iSpcTuningSectionParam.GetSubsectionOn() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

rezervCount – Количество резервных строк и позиций

Интерфейс...

Тип данных: long

Синтаксис Automation:

rezervCount = iSpcTuningSectionParam.rezervCount Получить свойство (*)
rezervCount = iSpcTuningSectionParam.GetRezervCount() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ksSpcTuningSectionParam – методы

GetArrSubSection – Получить указатель на интерфейс динамического массива параметров подраздела

спецификации ksDynamicArray типа SPCSUBSECTION_ARR

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetArrSubSection();

Возвращаемое значение:

- указатель на интерфейс динамического массива ksDynamicArray типа SPCSUBSECTION_ARR.

Смотрите также ksSpcSubSectionParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Создается динамический массив arrSubSection типа SPCSUBSECTION_ARR.

Подраздел (Интерфейс ksSpcSubSectionParam)

Справка системы КОМПАС...

KOMPAS.chm::/1145_131_2_Podrazdely.htm

Интерфейс параметров подраздела спецификации.

Аналог данных параметров при использовании API экспортных функций - SpcSubSectionParam

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetParamStruct.

Смотрите также KompasObject

ksSpcSubSectionParam – свойства

name – Имя подраздела

Интерфейс...

Тип данных: строка

Синтаксис Automation:

name = iSpcSubSectionParam.name Получить свойство (*)
name = iSpcSubSectionParam.GetName() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

number – Номер подраздела

Интерфейс...

Тип данных: long

Синтаксис Automation:

number = iSpcSubSectionParam.number Получить свойство (*)
number = iSpcSubSectionParam.GetNumber() Получить свойство (**)

Примечание:

Данное свойство предназначено только для чтения.

ksSpcSubSectionParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Параметры сохранения растра (Интерфейсы ksRasterFormatParam и IRasterFormatParam)

См.Интерфейсы сохранения растра

Интерфейсы спецификации ksSpecification, ISpecification3D

См. Интерфейс ksSpecificatio/Интерфейс ISpecification

Интерфейс фильтрации объектов документа-модели (Интерфейс ksObjectsFilter3D)

Справка системы КОМПАС...

KOMPAS.chm::/683_82_7_2_Filqtry_obwektov.htm

Интерфейс фильтрации объектов документа-модели.

ksObjectsFilter3D	- интерфейс Automation
IObjectsFilter3D	- интерфейс COM

Примечание:

1. Служит для включения режима, в котором возможен динамический поиск и указание курсором объектов модели: осей, плоскостей, граней, ребер и вершин.
2. Если установить свойство filterAll, то остальные флаги снимаются.
3. После установки любого свойства, разрешающего поиск и указание объектов определенного типа, свойство **Фильтровать все** отключается.
4. Если выключаются флаги, соответствующие типам объектов, то флаг **Фильтровать все** автоматически включается (то есть отключить указание всех типов объектов невозможно).
5. Указатель на интерфейс можно получить, используя свойство KompasObject::ksGetObjectsFilter3D или экспортную функцию ksGetObjectsFilter3D.

IObjectsFilter3D - свойства

filterAll - Фильтровать все

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- разрешить фильтрацию для всех типов объектов,
FALSE	- фильтрация разрешена только для конкретных типов объектов.

Синтаксис Automation:

filterAll = iObjFilter3D.filterAll;	Получить свойство (*)
iObjFilter3D.filterAll = filterAll;	Установить свойство (*)
filterAll = iObjFilter3D.GetFilterAll();	Получить свойство (**)
iObjFilter3D.SetFilterAll(filterAll);	Установить свойство (**)

Синтаксис COM:

filterAll = iObjFilter3D.GetFilterAll();	Получить свойство
iObjFilter3D.SetFilterAll(filterAll);	Установить свойство

Примечание:

Напрямую установить свойство равным FALSE нельзя. Свойство устанавливается равным FALSE после установки любых флагов для конкретных типов объектов.

filterCAxis – Фильтровать конструктивные оси

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- разрешить фильтрацию конструктивных осей,
FALSE	- запретить фильтрацию конструктивных осей.

Синтаксис Automation:

filterCAxis = iObjFilter3D.filterCAxis;	Получить свойство (*)
iObjFilter3D.filterCAxis = filterCAxis;	Установить свойство (*)
filterCAxis = iObjFilter3D.GetFilterCAxis();	Получить свойство (**)
iObjFilter3D.SetFilterCAxis(filterCAxis);	Установить свойство (**)

Синтаксис COM:

filterCAxis = iObjFilter3D.GetFilterCAxis();	Получить свойство
iObjFilter3D.SetFilterCAxis(filterCAxis);	Установить свойство

Примечание:

1. После установки свойства равным TRUE флаг Фильтровать все снимается.
2. Если после установки флага равным FALSE все флаги для типов объектов окажутся равными FALSE, то будет включено свойство **Фильтровать все**.

filterCPlanes – Фильтровать конструктивные плоскости

Интерфейс...

Тип данных: BOOL

Значения свойства:

TRUE	- разрешить фильтрацию конструктивных плоскостей,
FALSE	- запретить фильтрацию конструктивных плоскостей.

Синтаксис Automation:

filterCPlanes = iObjFilter3D.filterCPlanes;	Получить свойство (*)
iObjFilter3D.filterVertexs = filterCPlanes;	Установить свойство (*)
filterCPlanes = iObjFilter3D.GetFilterCPlanes();	Получить свойство (**)

```
iObjFilter3D.SetFilterCPlanes(filterCPlanes);          Установить свойство (**)
```

Синтаксис COM:

```
filterCPlanes = iObjFilter3D.GetFilterCPlanes();       Получить свойство
iObjFilter3D.SetFilterCPlanes(filterCPlanes);          Установить свойство
```

Примечание:

1. После установки свойства равным TRUE флаг Фильтровать все снимается.
2. Если после установки флага равным FALSE все флаги для типов объектов окажутся равными FALSE, то будет включено свойство **Фильтровать все**.

filterEdges – Фильтровать ребра

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - разрешить фильтрацию ребер,
FALSE       - запретить фильтрацию ребер.
```

Синтаксис Automation:

```
filterEdges = iObjFilter3D.filterEdges;                Получить свойство (* )
iObjFilter3D.filterEdges = filterEdges;                Установить свойство (* )
filterEdges = iObjFilter3D.GetFilterEdges();           Получить свойство (**)
iObjFilter3D.SetFilterEdges(filterEdges);              Установить свойство (**)
```

Синтаксис COM:

```
filterEdges = iObjFilter3D.GetFilterEdges();           Получить свойство
iObjFilter3D.SetFilterEdges(filterEdges);              Установить свойство
```

Примечание:

1. После установки свойства равным TRUE флаг Фильтровать все снимается.
2. Если после установки флага равным FALSE все флаги для типов объектов окажутся равными FALSE, то будет включено свойство **Фильтровать все**.

filterFaces – Фильтровать грани

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE        - разрешить фильтрацию граней,
```

FALSE - запретить фильтрацию граней.

Синтаксис Automation:

```
filterFaces = iObjFilter3D.filterFaces;          Получить свойство (* )
iObjFilter3D.filterFaces = filterFaces;          Установить свойство (* )
filterFaces = iObjFilter3D.GetFilterFaces();     Получить свойство (**)
iObjFilter3D.SetFilterFaces(filterFaces);        Установить свойство (**)
```

Синтаксис COM:

```
filterFaces = iObjFilter3D.GetFilterFaces();     Получить свойство
iObjFilter3D.SetFilterFaces(filterFaces);        Установить свойство
```

Примечание:

1. После установки свойства равным TRUE флаг Фильтровать все снимается.
2. Если после установки флага равным FALSE все флаги для типов объектов окажутся равными FALSE, то будет включено свойство **Фильтровать все**.

filterVertexs – Фильтровать вершины

Интерфейс...

Тип данных: BOOL

Значения свойства:

```
TRUE    - разрешить фильтрацию вершин,
FALSE   - запретить фильтрацию вершин.
```

Синтаксис Automation:

```
filterVertexs = iObjFilter3D.filterVertexs;          Получить
                                                     свойство (* )
iObjFilter3D.filterVertexs = filterVertexs;          Установить
                                                     свойство (* )
filterVertexs                              =         Получить
iObjFilter3D.GetFilterVertexs();                     свойство (**)
iObjFilter3D.SetFilterVertexs(filterVertexs);        Установить
                                                     свойство (**)
```

Синтаксис COM:

```
filterVertexs = iObjFilter3D.GetFilterVertexs();     Получить свойство
iObjFilter3D.SetFilterVertexs(filterVertexs);        Установить свойство
```

Примечание:

1. После установки свойства равным TRUE флаг Фильтровать все снимается.
2. Если после установки флага равным FALSE все флаги для типов объектов окажутся равными FALSE, то будет включено свойство **Фильтровать все**.

Интерфейсы работы с библиотеками

Библиотека моделей (Интерфейсы ksModelLibrary и IModelLibrary)

Интерфейс библиотеки трехмерных моделей.

ksModelLibrary	- интерфейс Automation
IModelLibrary	- интерфейс COM

Примечания:

Указатель на этот интерфейс при использовании Automation можно получить при помощи метода KompasObject::GetModelLibrary.

При использовании COM интерфейс можно получить при помощи экспортной функции ksGetModelLibrary.

ksModelLibrary – методы

AddD3DocumentToLibrary – Добавить модель с указанным именем файла в указанную библиотеку моделей

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

long AddD3DocumentToLibrary (BSTR libName,

BSTR libFile);

Синтаксис COM:

long AddD3DocumentToLibrary (LPOLESTR libName,

LPOLESTR fileName);

Входные параметры:

libName	- имя файла библиотеки моделей,
libFile	- имя файла модели.

Возвращаемое значение:

1	- в случае успешного завершения.

Примечание:

Имя файла библиотеки моделей должно содержать путь внутри библиотеки и имя модели, например: "c:\gr\lib1.l3d|детали|литье|фланец",
где
c:\gr\lib1.l3d - имя файла библиотеки моделей,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя модели.

CheckModelLibrary – Проверить, открыта ли указанная библиотека моделей

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

long CheckModelLibrary (BSTR libName,

BOOL possibleMessage);

Синтаксис COM:

long CheckModelLibrary (LPOLESTR libName,

BOOL possibleMessage);

Входные параметры:

libName	- имя файла библиотеки моделей,
possibleMessage	- признак, показывающий, нужно ли выдать сообщение об открытии библиотеки: TRUE - сообщение выдается, FALSE - сообщение не выдается.

Возвращаемое значение:

1	- библиотека открыта (успешное завершение),
0	- библиотека закрыта.

ChoiceModelFromLib – Выбор модели из библиотеки

Интерфейс...

Синтаксис Automation:

BSTR ChoiceModelFromLib (BSTR libFile,

long* type);

Синтаксис COM:

LPOLESTR ChoiceModelFromLib (LPOLESTR libFile,

long* type);

Входной параметр:

libFile - путь к файлу в библиотеке моделей.

Выходной параметр:

type - тип возвращаемого пути.

Типы пути в библиотеке моделей...

Возвращаемое значение:

путь к модели - в случае успешного завершения.

ExistModelInLibrary – Проверить, существует ли в библиотеке указанная модель или папка

Интерфейс...

Синтаксис Automation:

long ExistModelInLibrary (BSTR name);

Синтаксис COM:

long ExistModelInLibrary (LPOLESTR name);

Входной параметр:

name - имя файла библиотеки моделей и модели (или папки) в нем.

Возвращаемое значение:

-1 - указанная библиотека не существует,
0 - указанная модель или папка не существует,
1 - модель или папка существует.

Примечание:

Допускается задание имени файла библиотеки следующего вида: "c:\gr\lib1.l3d|детали|литье|фланец",
где
c:\gr\lib1.l3d - имя файла библиотеки моделей,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя модели.

ModelLibraryOperation – Действия с библиотекой моделей

Функция не поддерживается

Интерфейс...

Синтаксис Automation:

long ModelLibraryOperation (BSTR libName,

long type);

Синтаксис COM:

long ModelLibraryOperation (LPOLESTR libName,

long type);

Входные параметры:

libName	- имя файла библиотеки моделей,
type	- тип действия.

Типы действий с библиотекой моделей...

Возвращаемое значение:

1	- в случае успешного завершения.

Примечания:

1. При выполнении операций -1, 0, 1, 4, 5 libName - полное имя файла библиотеки моделей.
2. При выполнении операций 2, 3 libName - полное имя файла библиотеки моделей путь внутри библиотеки с именем редактируемой модели.
3. Допускается задание имени файла библиотеки следующего вида: "c:\gr\lib1.l3d|детали|литье|фланец",
   где
   c:\gr\lib1.l3d - имя файла библиотеки моделей,

   |детали|литье| - разделы, подразделы внутри библиотеки моделей,

   фланец - имя модели.

Библиотека фрагментов (Интерфейс ksFragmentLibrary)

Интерфейс библиотеки фрагментов.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetFragmentLibrary.

Смотрите также KompasObject

ksFragmentLibrary – методы

ksAddFragmentToLibrary – Добавить фрагмент в библиотеку

Функция не поддерживается

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksAddFragmentToLibrary.

Синтаксис Automation:

long ksAddFragmentToLibrary (BSTR libName,

BSTR frwName);

Входные параметры:

libName	- имя фрагмента в библиотеке,
libFile	- имя файла фрагмента.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец",
где
c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

ksCheckFragmentLibrary – Проверить, открыта ли библиотека фрагментов

Функция не поддерживается

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksCheckFragmentLibrary.

Синтаксис Automation:

long ksCheckFragmentLibrary (BSTR libName,

BOOL possibleMessage);

Входные параметры:

libName	- имя файла библиотеки фрагментов,

possibleMessage	- признак выдачи сообщения: TRUE - выдать сообщение, FALSE - не выдавать сообщение.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksChoiceFragmentFromLib – Выбрать имя фрагмента или папки в библиотеке фрагментов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksChoiceFragmentFromLib.

Синтаксис Automation:

BSTR ksChoiceFragmentFromLib (BSTR frwLibFile,

long* type);

Входные параметры:

frwLibFile	- имя файла библиотеки фрагментов,
type	- тип возвращаемого значения: 3 - имя фрагмента, 2 - имя папки, 1- корень библиотеки фрагментов, 0 - ошибка.

Возвращаемое значение:

имя фрагмента.

Примечание:

Библиотека фрагментов открывается в режиме диалога.

Ее редактирование запрещено.

Enter - выбор имени фрагмента. Esc - отказ.

ksExistFragmentInLibrary – Проверить наличие указанного фрагмента или раздела в библиотеке фрагментов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksExistFragmentInLibrary.

Синтаксис Automation:

long ksExistFragmentInLibrary (BSTR frwName);

Входной параметр:

name	- имя файла библиотеки моделей.

Возвращаемое значение:

1	- указанный фрагмент или папка есть в библиотеке,
0	- указанного фрагмента или папки нет,
-1	- нет указанной библиотеки.

Примечание:

Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

ksFragmentLibraryOperation – Открыть библиотеку фрагментов

Функция не поддерживается

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ksFragmentLibrary.

Синтаксис Automation:

long ksFragmentLibraryOperation (LPCTSTR libName, long type);

Входные параметры:

libName	- имя файла библиотеки фрагментов,
type	- тип действия.

Типы действий с библиотекой фрагментов...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

При выполнении операций типа -1, 0, 1, 4, 5 libName - полное имя файла библиотеки фрагментов.

При выполнении операций типа 2, 3 libName - полное имя файла библиотеки фрагментов + путь внутри библиотеки с именем редактируемого фрагмента.

Параметры узла дерева библиотеки (Интерфейс ksTreeNodeParam)

Интерфейс параметров узла дерева библиотеки документов, библиотеки атрибутов.

Аналог данных параметров при использовании API экспортных функций - ksTreeNodeParam.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса KompasObject::ksGetLibraryTreeStruct.

ksTreeNodeParam – свойства

name – Имя узла

Интерфейс...

Тип данных: BSTR

Синтаксис Automation:

name = iTreeNodeParam.name		Получить свойство (*)
name	=	Получить свойство
iTreeNodeParam.GetName()		(**)

Возвращаемое значение:

Имя узла.

Примечание:

Свойство доступно только для чтения.

type – Тип узла: корень, папка, файл

Интерфейс...

Тип данных: long

Синтаксис Automation:

type = iTreeNodeParam.type	Получить свойство (*)
type = iTreeNodeParam.GetType()	Получить свойство (**)

Возвращаемое значение:

- тип узла: корень, папка, файл из LtNodeType.

Примечание:

Свойство доступно только для чтения.

ksTreeNodeParam – методы

GetComment – Получить указатель на интерфейс массива строк комментария

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetComment();

Возвращаемое значение:

- указатель на интерфейс ksDynamicArray массива строк комментария; тип массива - CHAR_STR_ARR.

GetNodes – Получить указатель на интерфейс ksDynamicArray массива дочерних узлов

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetNodes();

Возвращаемое значение:

- указатель на интерфейс ksDynamicArray массива дочерних узлов; тип массива - TREENODEPARAM_ARR.

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Интерфейс фантома (Интерфейс ksPhantom)

см. Интерфейс ksPhantom

База данных (Интерфейс ksDataBaseObject)

Интерфейс базы данных.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::DataBaseObject.

Смотрите также KompasObject

ksDataBaseObject – методы

ksCloseTextFile – Закрыть текстовый файл запросов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - CloseTextFile.

Синтаксис Automation:

BOOL ksCloseTextFile (long F);

Входной параметр:

F	- указатель на текстовый файл.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksCondition – Задать или изменить условие запроса

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Condition.

Синтаксис Automation:

long ksCondition (long db,

long r,

BSTR stSQL);

Входные параметры:

db	- указатель на объект БД,
r	- указатель на отношение,
stSQL	- запрос.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

При работе с базой данных через ODBC-интерфейс происходит переопределение составляющей SQL-запроса, начинающейся с ключевого слова WHERE. Состав обрабатываемых полей записи, определенный в отношении, не изменяется.

При работе с текстовыми файлами использование функции ksDataBaseObject::ksCondition является единственной возможностью определения условия запроса, так как в функции ksDataBaseObject::ksDoStatement определяется только список обрабатываемых полей записи.

Следует отметить, что при работе с текстовыми базами данных не обрабатывается вложенность условий (например, "where d > 10 and d < 14").

ksConnectDB – Связать объект БД с конкретной базой данных

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ConnectDB.

Синтаксис Automation:

long ksConnectDB (long db,

BSTR DBName);

Входные параметры:

db	- указатель на объект БД,
DBName	- имя БД (для ODBC - имя БД в администраторе ODBC, для текстового файла - имя файла).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksCreateDB – Создать блок заголовка базы данных

Интерфейс...

Синтаксис Automation:

long ksCreateDB (BSTR typeBD);

Входной параметр:

typeBD	- тип базы данных: TXT_DB - база данных текстового формата, ODBC_DB - база данных, доступная через интерфейс ODBC.

Возвращаемое значение:

указатель на блок заголовка базы данных	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция позволяет открыть БД. Одновременно может быть открыто несколько БД.

ksDeleteDB – Удалить блок заголовка базы данных

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteDB.

Синтаксис Automation:

long ksDeleteDB(long db);

Входной параметр:

db	- указатель на объект БД.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Происходит автоматическое отсоединение базы данных и удаление всех созданных ранее запросов.

ksDisconnectDB – Отключиться от базы данных

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DisconnectDB.

Синтаксис Automation:

long ksDisconnectDB (long db);

Входной параметр:

db	- указатель на объект БД.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция отсоединяет блок заголовка с указателем bd от базы данных.

Следует отметить, что при удалении блока заголовка (функция ksDataBaseObject::ksDeleteDB) отсоединение выполняется автоматически, поэтому использовать ее рекомендуется только при переопределении базы данных.

ksDoStatement – Установить запрос для объекта БД

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DoStatement.

Синтаксис Automation:

long ksDoStatement (long db, long r,

BSTR stSQL);

Входные параметры:

db	- указатель на объект БД,
r	- действительный указатель на отношение,
stSQL	- запрос.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр stSQL при работе через ODBC-интерфейс содержит строку SQL-запроса, а при работе с текстовыми файлами - номера полей (колонок) или пустую строку (если обрабатываются все поля таблицы).

Для текстовых баз данных созданный запрос будет обрабатывать все записи, а непосредственно условие выборки определяется функцией ksDataBaseObject::ksCondition.

Для ODBC-баз отношение не обязательно в случае вставки, удаления и обновления записи. Для текстового файла отношение в этих случаях необходимо, чтобы определить имена колонок.

Примеры:

Select d, s, p from bolt where d = 10 - пример запроса выборки из БД,
где d, s, p - названия колонок или * для всех колонок или номера колонок "2, 4, 7" для текстового файла, начиная с единицы слева направо,
bolt - имя таблицы в БД или пустая строка для всех колонок текстового файла,
d - имя колонки в отношении.

Insert into bolt (d,p,s) values(10, 1.5, 14) - пример запроса для вставки строки в таблицу, где bolt - имя таблицы в БД или пустая строка для всех колонок текстового файла.

Delete from bolt where d = 10 - пример запроса для удаления строки из таблицы bolt.

Update bolt set p = 2.5, s = 20 where d =10 - пример запроса для замены данных в строке таблицы bolt.

ksEndRelation – Завершить описание отношения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - EndRelation.

Синтаксис Automation:

BOOL ksEndRelation();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание :

Описание отношения начинается методом ksDataBaseObject::ksRelation.

ksFreeStatement – Освободить отношения

Интерфейс...

Аналог данного метода при использовании API экспортных функций - FreeStatement.

Синтаксис Automation:

long ksFreeStatement (long db, long r, long fOption);

Входные параметры:

db	- указатель на объект БД,
r	- указатель на отношение,
fOption	- тип освобождения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Происходит освобождение памяти, отведенной для буфера записи функцией ksDataBaseObject::ksRelation.

Параметр fOption имеет значение при работе через ODBC-интерфейс и означает тип освобождения (описан в помощи модуля ODBC).

ksGetColumnName – Считать имя колонки таблицы из базы данных

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetColumnName.

Синтаксис Automation:

BSTR ksGetColumnName (long db,

BSTR tableName,

long* res,

BSTR firstOrNext);

Входные параметры:

db	- указатель на объект БД,
tableName	- ODBC - имя таблицы, текстовая БД - имя файла,
firstOrNext	- признак колонки:
	F - первая колонка,
	N - следующая колонка.

Выходной параметр:

res - результат работы:
1 - если в указанной таблице или базе еще существуют несчитанные имена колонок,
0 - если все имена колонок считаны.

Возвращаемое значение:

имя колонки.

Примечание:

Для текстовых БД возвращаются номера колонок.

ksGetTableName – Считать имя таблицы

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetTableName.

Синтаксис Automation:

BSTR ksGetTableName (long db,

long* res,

BSTR firstOrNext);

Входные параметры:

db	- указатель на объект БД,
firstOrNext	- признак таблицы:
	F - первая таблица,
	N - следующая таблица.

Выходной параметр:

res - результат работы:
1- если в указанной базе еще существуют несчитанные имена таблиц,
0 - если все имена таблиц считаны.

Возвращаемое значение:

имя таблицы.

ksIsODBCOkey – Проверить подключение ODBC

Интерфейс...

Аналог данного метода при использовании API экспортных функций - IsODBCOkey.

Синтаксис Automation:

long ksIsODBCOkey();

Возвращаемое значение:

1 - если соединение с ODBC установлено,
0 - если соединения нет.

ksOpenTextFile – Открыть текстовый файл запросов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - OpenTextFile.

Синтаксис Automation:

long ksOpenTextFile (BSTR fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

указатель на открытый файл - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Открыть текстовый файл запросов, настроенный на базу данных.

Функция работает по правилам стандартной процедуры WINDOWS OpenFile.

Реализованные в рамках данного раздела функции работы с текстовыми файлами обеспечивают их построчную обработку и позволяют вынести обращения к базе данных во

внешний текстовый файл. Это дает возможность работать с разными платформами СУБД без изменения выполняемого кода приложения.

ksRChar – Определить в отношении строковое поле

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RChar.

Синтаксис Automation:

long ksRChar (BSTR name,

long size,

long type);

Входные параметры:

name	- имя колонки таблицы,
size	- размер буфера,
type	- тип данных, хранящихся в БД (действительно для ODBC-баз).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен при работе с текстовой базой данных.

Параметр type имеет смысл только для баз данных, связанных через ODBC, и описывает действительный тип колонки базы данных, с которой будет связано поле при выполнении операции ksDataBaseObject::ksDoStatement. Это позволяет получать строковое представление записи произвольного типа. Описания типов для ODBC хранятся в файле SQL.h, который поставляется вместе с модулями ODBC.

ksRCharW – Определить в отношении строковое поле

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RCharW.

Синтаксис Automation:

long ksRCharW (BSTR name,

long size,

long type);

Входные параметры:

name	- имя колонки таблицы,

size	- размер буфера,
type	- тип данных, хранящихся в БД (действительно для ODBC-баз).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен при работе с текстовой базой данных.

Параметр type имеет смысл только для баз данных, связанных через ODBC, и описывает действительный тип колонки базы данных, с которой будет связано поле при выполнении операции ksDataBaseObject::ksDoStatement. Это позволяет получать строковое представление записи произвольного типа. Описания типов для ODBC хранятся в файле SQL.h, который поставляется вместе с модулями ODBC.

ksRDouble – Определить в отношении поле типа double

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RDouble.

Синтаксис Automation:

long ksRDouble (BSTR name);

Входной параметр:

name	- имя поля.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

ksReadRecord – Получить запись базы данных

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ReadRecord.

Синтаксис Automation:

long ksReadRecord (long db,

long r,

LPDISPATCH userPars);

Входные параметры:

db	- указатель на объект БД,
r	- указатель на отношение,
userPars	- указатель на интерфейс ksUserParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Количество и типы полей в структуре userPars должны соответствовать отношению r.

ksReadStrFrFile – Считать строку из текстового файла запросов

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ReadStrFromTextFile.

Синтаксис Automation:

BSTR ksReadStrFrFile (long f,

long* res,

long numb);

Входные параметры:

f	- указатель на текстовый файл,
numb	- номер строки.

Выходной параметр:

res	- результат работы метода: 1 - в случае успешного завершения, 0 - в случае неудачи.

Возвращаемое значение:

- строка обращения к базе данных.

Примечание:

Строка обращения к базе данных начинается с номера, далее после двоеточия идет текст обращения, например:

3:select d,f from Table1 where d=10 and f>d

ksRelation – Создать новое отношение

Интерфейс...

Аналог данного метода при использовании API экспортных функций - Relation.

Синтаксис Automation:

long ksRelation (long db);

Входной параметр:

db	- указатель на объект БД.

Возвращаемое значение:

указатель на отношение	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Отношение характеризует один запрос и представляет собой Описание памяти, необходимой для размещения очередной записи выборки, выполняемого функцией ksDataBaseObject::ksReadRecord. Является составным объектом, каждое поле которого описывает тип и имя поля (колонки) в таблице базы данных. Имя действительно только в случае работы с текстовым файлом, так как при обмене через ODBC-интерфейс имена уже описаны в блоке заголовка.

Количество отношений, определенных для базы данных, не ограничивается.

ksRFloat – Определить в отношении поле типа float

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RFloat.

Синтаксис Automation:

long ksRFloat (BSTR name);

Входной параметр:

name	- имя поля.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

ksRInt– Определить в отношении поле типа Int

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RInt.

Синтаксис Automation:

long ksRInt (BSTR name);

Входной параметр:

name	- имя поля.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

ksRLong – Определить в отношении поле типа long

Интерфейс...

Аналог данного метода при использовании API экспортных функций - RLong.

Синтаксис Automation:

long ksRLong (BSTR name);

Входной параметр:

name	- имя поля.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

Динамический массив (Интерфейс ksDynamicArray)

Интерфейс динамического массива.

Указатель на интерфейс можно получить при помощи методов...

Типы динамических массивов...

См. также...

ksDynamicArray – свойства

reference – Указатель на массив

Интерфейс...

Тип данных: long

Синтаксис Automation:

ref = iDynamicArray.reference	Получить свойство (*)
iDynamicArray.reference = ref	Установить свойство (*)
ref = iDynamicArray.GetReference()	Получить свойство (**)
iDynamicArray.SetReference(ref)	Установить свойство (**)

ksDynamicArray – методы

ksAddArrayItem – Добавить элемент в массив

Интерфейс...

Аналог данного метода при использовании API экспортных функций - AddArrayItem.

Синтаксис Automation:

long ksAddArrayItem (long index, LPDISPATCH item);

Входные параметры:

index	- индекс элемента, перед которым нужно вставить новый элемент, нумерация начинается с 0, при index=-1 элемент добавляется в конец массива,
item	- указатель на интерфейс соответствующего типа.

Возвращаемое значение:

1	- в случае удачного завершения.

Смотрите также:

Типы динамических массивов...

ksClearArray – Очистить массив

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ClearArray.

Синтаксис Automation:

long ksClearArray();

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Из массива удаляются все элементы, но сам массив не удаляется.
2. При удалении пользовательского элемента, автоматические вызывается пользовательская функция удаления элемента, заданная пользователем при создании массива.

ksDeleteArray – Удалить массив

Интерфейс...

Аналог данного метода при использовании API экспортных функций - DeleteArray.

Синтаксис Automation:

long ksDeleteArray();

Возвращаемое значение:

1 - в случае удачного завершения.

Примечание:

Может быть удален непустой массив, то есть перед вызовом метода ksDynamicArray::ksDeleteArray вызывать ksDynamicArray::ksClearArray необязательно.

ksExcludeArrayItem – Удалить элемент массива

Интерфейс...

Аналог данного метода при использовании API экспортных функций - ExcludeArrayItem.

Синтаксис Automation:

long ksExcludeArrayItem (long index);

Входной параметр:

index - индекс элемента в массиве, нумерация начинается с 0.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetArrayCount – Получить количество элементов в массиве

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetArrayCount.

Синтаксис Automation:

long ksGetArrayCount();

Возвращаемое значение:

- количество элементов в массиве.

ksGetArrayItem – Получить элемент массива

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetArrayItem.

Синтаксис Automation:

long ksGetArrayItem (long index,

LPDISPATCH item);

Входной параметр:

index - индекс элемента в массиве, нумерация начинается с 0.

Выходной параметр:

item - указатель на интерфейс соответствующего типа.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Смотрите также:

Типы динамических массивов...

ksGetArrayType – Получить тип массива

Интерфейс...

Аналог данного метода при использовании API экспортных функций - GetArrayType.

Синтаксис Automation:

long ksGetArrayType();

Возвращаемое значение:

- тип массива.

Типы динамических массивов...

ksSetArrayItem – Установить параметры элемента в массиве

Интерфейс...

Аналог данного метода при использовании API экспортных функций - SetArrayItem.

Синтаксис Automation:

long ksSetArrayItem(long index,

LPDISPATCH item);

Входные параметры:

index	- индекс элемента в массиве, нумерация начинается с 0,
item	- указатель на интерфейс соответствующего типа.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также:

Типы динамических массивов...

Интерфейсы работы с атрибутами

Интерфейс работы с атрибутами (ksAttributeObject)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Интерфейс для работы с атрибутами.

Примечание:

Указатель на интерфейс можно получить при помощи метода KompasObject::GetAttributeObject.

Смотрите также KompasObject

ksAttributeObject – методы

ksAddAttrRow – Добавить строку к табличному атрибуту неопределенной длины

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ksAddAttrRow.

Синтаксис Automation:

long ksAddAttrRow (reference pAttr,

long rowNumb,

LPDISPATCH flagVisible,

LPDISPATCH value,

BSTR password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего флаги видимости ячеек строки,
value	- указатель на интерфейс ksUserParam, откуда копируются данные,
password	- пароль атрибута.

Возвращаемое значение:

1 - в случае удачного завершения.

Смотрите также ksLtVariant

Примечание.

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. Перед использованием указатель flagVisible должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_LtVariant, методов интерфейса ksLtVariant, ksDynamicArray.
3. Перед использованием указатель value должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_UserParam с последующим наполнением массива с помощью методов интерфейса ksUserParam, ksLtVariant, ksDynamicArray.

ksChoiceAttr – Вывести диалог для просмотра атрибутов объекта и выбора нужного атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ChoiceAttr.

Синтаксис Automation:

long ksChoiceAttr (reference pObj);

Входной параметр:

pObj	- указатель на объект, к которому подключены атрибуты.

Возвращаемое значение:

указатель на атрибут	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Указатель на объект reference pObj может быть получен следующими способами:

- с помощью свойства reference интерфейса соответствующего объекта,
- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateIterator, ksIterator::ksMoveIterator.

ksChoiceAttr3D – Просмотреть атрибуты объекта документа-модели

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данного метода при использовании API экспортных функций - ksChoiceAttr3D.

Синтаксис Automation:

ksAttribute3D * ksChoiceAttr3D (LPDISPATCH pObj);

Входные параметры:

pObj	- указатель на объект.

Возвращаемое значение:

Указатель на интерфейс атрибута ksAttribute3D	- в случае удачного завершения,
0	- в случае неудачи.

ksChoiceAttrTypes – Выбрать указанный пользователем в диалоге тип атрибута из библиотеки или документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ChoiceAttrTypes.

Синтаксис Automation:

double ksChoiceAttrTypes (BSTR libName);

Входной параметр:

libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется из текущего документа.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Диалог "привязан" к главному окну.

ksCreateAttr – Создать атрибут по номеру типа атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/512_63_1_1_Atribut_odnogo_grafi.htm

Аналог данного метода при использовании API экспортных функций - ksCreateAttr.

Синтаксис Automation:

long ksCreateAttr (reference pObj,

LPDISPATCH attr,

double attrID,

BSTR libName);

Входные параметры:

pObj	- указатель на объект (группа, вид, отдельный объект), атрибут которого создается. Если pObj = 0, то создается атрибут документа,
attr	- указатель на интерфейс параметров атрибута ksAttributeParam,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется из текущего документа.

Возвращаемое значение:

указатель на атрибут	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

1. Указатель на объект reference pObj может быть получен следующими способами:

- с помощью свойства reference интерфейса соответствующего объекта,
- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,

2. Параметры атрибута должны быть заданы с помощью метода KompasObject::GetParamStruct с параметром ko_Attribute с последующим наполнением массива с помощью свойств и методов интерфейса ksAttributeParam.
3. Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid. Тип атрибута должен быть предварительно создан, например, с помощью метода ksAttributeObject::ksCreateAttrType.

ksCreateAttr3D – Создать атрибут по номеру типа атрибута из библиотеки

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/512_63_1_1_Atribut_odnogo_grafi.htm`

Аналог данного метода при использовании API экспортных функций - ksCreateAttr3D.

Синтаксис Automation:

```
ksAttribute3D* ksCreateAttr3D (LPDISPATCH pObj,
LPDISPATCH attr,
double attrID,
BSTR libname);
```

Входные параметры:

pObj	- указатель на объект для которого создается атрибут,
attr	- указатель на интерфейс параметров атрибута ksAttributeParam,
attrID	- уникальный номер типа атрибута,
libname	- имя библиотеки атрибутов если libname = NULL- тип атрибута берется в документе.

Возвращаемое значение:

Указатель на интерфейс атрибута ksAttribute3D	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Атрибут можно добавить ко всем объектам дерева построений, кроме компонентов, сопряжений, группы сопряжений.
2. Параметры атрибута должны быть заданы с помощью метода KompasObject::GetParamStruct с параметром ko_Attribute с последующим наполнением массива с помощью свойств и методов интерфейса ksAttributeParam.
3. Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid. Тип атрибута должен быть предварительно создан, например, с помощью метода ksAttributeObject::ksCreateAttrType.

ksCreateAttr3DEx – Создать атрибут по номеру типа атрибута из библиотеки libname

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/512_63_1_1_Atribut_odnogo_grafi.htm`

Аналог данного метода при использовании API экспортных функций - ksCreateAttr3DEx.

Синтаксис Automation:

```
ksAttribute3D * ksCreateAttr3DEx (LPDISPATCH pObj,
LPDISPATCH pSourcePart,
LPDISPATCH attr,
double attrID,
BSTR libname);
```

Входные параметры:

pObj	- указатель на объект для которого создается атрибут,
sourcePart	- указатель на интерфейс вставки детали IPart,
attr	- указатель на структуру параметров атрибута ksAttribute,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется в документе.

Возвращаемое значение:

Указатель на атрибут IAttribute3D	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если libname = NULL - тип атрибута берется в документе.
2. pObj - может быть равен:

▼ NULL или указателю на интерфейс 3d документа IDocument3D - создается атрибут у документа,

- указателю на интерфейс коллекции объектов дерева IFeatureCollection - атрибут групповой,
- указателю на интерфейс объекта дерева IFeature - атрибут у определенного объекта.

3. Атрибут можно добавить ко всем объектам дерева построений, кроме верхнего компонентов, сопряжений, группы сопряжений.
4. sourcePart - может быть:

- NULL - атрибут будет создан в текущем документе,
- указателю на интерфейс детали или подсборки вставленной в сборку IPart - атрибут будет создан в документе-источнике,
- если sourcePart == pObj создается атрибут документа в источнике.

ksCreateAttrType – Создать тип атрибута в библиотеке

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/512_63_1_1_Atribut_odnogo_grafi.htm`

Аналог данного метода при использовании API экспортных функций - ksCreateAttrType.

Синтаксис Automation:

double ksCreateAttrType (LPDISPATCH attrType,

BSTR libName);

Входные параметры:

attrType	- указатель на интерфейс параметров типа табличного атрибута ksAttributeTypeParam,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута создается в текущем документе.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Параметры атрибута должны быть заданы с помощью метода KompasObject::GetParamStruct с параметром ko_AttributeType с последующим наполнением массива с помощью свойств и методов интерфейса ksAttributeTypeParam.

ksDeleteAttr – Удалить атрибут

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - DeleteAttr.

Синтаксис Automation:

long ksDeleteAttr (reference pObj,

reference pAttr,

BSTR password);

Входные параметры:

pObj	- указатель на объект, к которому подключен атрибут. Если pObj = 0 - атрибут текущего документа,
pAttr	- указатель на удаляемый атрибут,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

1. Указатель на объект reference pObj может быть получен следующими способами:

- с помощью свойства reference интерфейса соответствующего объекта,
- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateIterator, ksIterator::ksMoveIterator.

2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksDeleteAttrRow– Удалить строку табличного атрибута неопределенной длины

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Аналог данного метода при использовании API экспортных функций - ksDeleteAttrRow.

Синтаксис Automation:

long ksDeleteAttrRow (reference pAttr,

long rowNumb,

BSTR password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksDeleteAttrType – Удалить тип атрибута из библиотеки

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данного метода при использовании API экспортных функций - DeleteAttrType.

Синтаксис Automation:

long ksDeleteAttrType (double attrID,

BSTR libName,

BSTR password);

Входные параметры:

attrID	- уникальный номер типа,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута удаляется из текущего документа,
password	- пароль типа атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid или с помощью метода ksAttributeObject::ksGetAttrKeysInfo.

ksDeleteAttr3D – Удалить атрибут объекта 3D документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/516_63_3_Operacii_s_atributami_.htm

Аналог данного метода при использовании API экспортных функций - ksDeleteAttr3D.

Синтаксис Automation:

long ksDeleteAttr3D (LPDISPATCH pObj, ksAttribute3D *pAttr, BSTR password);

Входные параметры:

pObj - указатель на объект, у которого удаляется атрибут,
pAttr - указатель на атрибут,
password - пароль.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetAttrColumnInfo – Получить информацию о столбце атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - GetAttrColumnInfo.

Синтаксис Automation:

long ksGetAttrColumnInfo (reference pAttr,

long columnNumb,

LPDISPATCH columnInfo);

Входные параметры:

pAttr - указатель на атрибут,
columnNumb - номер колонки.

Выходной параметр:

columnInfo - указатель на интерфейс параметров столбца табличного атрибута ksColumnInfoParam.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksGetAttrKeysInfo – Получить информацию о ключах атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - GetAttrKeysInfo.

Синтаксис Automation:

long ksGetAttrKeysInfo (reference pAttr,

long* key1,

long* key2,

long* key3,

long* key4,

double* attrID);

Входной параметр:

pAttr - указатель на атрибут.

Выходные параметры:

key1, key2, key3, key4 - ключи атрибута,
attrID - уникальный номер типа атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksGetAttrRow – Получить строку атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данного метода при использовании API экспортных функций - ksGetAttrRow.

Синтаксис Automation:

long ksGetAttrRow (reference pAttr,

long rowNumb,

LPDISPATCH flagVisible,

LPDISPATCH columnKeys,

LPDISPATCH value);

Входные параметры:

pAttr - указатель на атрибут,
rowNumb - номер строки.

Выходной параметр:

value - указатель на интерфейс ksUserParam, куда копируются данные.
flagVisible - указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего флаги видимости ячеек строки,
columnKeys - указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего ключи строки.

Возвращаемое значение:

1 - в случае удачного завершения.

Смотрите также:ksLtVariant

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. Перед использованием указатель flagVisible должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_LtVariant.
3. Перед использованием указатель value должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_UserParam.
4. Для получения размера данных строки атрибута может быть использован метод ksAttributeObject::ksGetSizeAttrRow.

ksGetAttrTabInfo – Получить информацию о количестве строк и столбцов атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - GetAttrTabInfo.

Синтаксис Automation:

long ksGetAttrTabInfo (long pAttr,

long* rowsCount,

long* columnsCount);

Входной параметр:

pAttr	- указатель на атрибут.

Выходные параметры:

rowsCount	- количество строк атрибута,
columnsCount	- количество столбцов атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,

- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksGetAttrType – Получить описание типа табличного атрибута из библиотеки или текущего документа

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Аналог данного метода при использовании API экспортных функций - ksGetAttrType.

Синтаксис Automation:

long ksGetAttrType (double attrID,

BSTR libName,

LPDISPATCH attrType);

Входной параметр:

attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута из текущего документа.

Выходной параметр:

attrType	- указатель на интерфейс параметров типа табличного атрибута ksAttributeTypeParam.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid или с помощью метода ksAttributeObject::ksGetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью метода ksAttributeObject::ksCreateAttrType.
2. Перед использованием указатель attrType должен быть получен с помощью метода KompasObject::GetParamStruct с параметром ko_AttributeType.

ksGetAttrValue – Получить значение ячейки из таблицы атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ksGetAttrValue.

Синтаксис Automation:

long ksGetAttrValue (reference pAttr,

long rowNumb,

long columnNumb,

LPDISPATCH flagVisible,

LPDISPATCH columnKeys,

LPDISPATCH value);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки.

Выходной параметр:

value	- указатель на интерфейс ksUserParam, куда копируется значение ячейки,
flagVisible	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего флаги видимости ячеек,
columnKeys	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего ключи полей колонки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также:ksLtVariant

Примечание

1. Для нетабличного атрибута номер колонки равен нулю.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

3. Перед использованием указатель flagVisible должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_LtVariant.

4. Перед использованием указатель value должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_UserParam.
5. Для получения размера данных ячейки атрибута может быть использован метод ksAttributeObject::ksGetSizeAttrValue.

ksGetLibraryAttrTypesArray – Получить указатель на динамический массив типов атрибутов типа LIBRARY_ATTR_TYPE_ARR, находящихся в заданной библиотеке типов – ksDynamicArray

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Аналог данного метода при использовании API экспортных функций - ksGetLibraryAttrTypesArray.

Синтаксис Automation:

LPDISPATCH ksGetLibraryAttrTypesArray (BSTR libName);

Входной параметр:

libName	- имя библиотеки типов атрибутов. Если libname = NULL, то типы атрибутов из текущего документа.

Возвращаемое значение:

указатель на интерфейс массива ksDynamicArray типа LIBRARY_ATTR_TYPE_ARR	- в случае удачного завершения,
0	- в случае неудачи.

ksGetSizeAttrRow – Получить размер данных строки атрибутов

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данного метода при использовании API экспортных функций - ksGetSizeAttrRow.

Синтаксис Automation:

long ksGetSizeAttrRow (reference pAttr,

long* count);

Входной параметр:

pAttr - указатель на атрибут.

Выходной параметр:

count - число ячеек с учетом записей.

Возвращаемое значение:

длина строки атрибута - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. При использовании данного метода, не использующего Unicode, размер данных строкового типа будет рассчитываться для unicode строк, как для ANSI строк.

ksGetSizeAttrRowW – Получить размер данных строки атрибутов (Unicode)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данного метода при использовании API экспортных функций - ksGetSizeAttrRowW.

Синтаксис Automation:

long ksGetSizeAttrRowW (reference pAttr,

long* count);

Входной параметр:

pAttr - указатель на атрибут.

Выходной параметр:

count - число ячеек с учетом записей.

Возвращаемое значение:

длина строки атрибута - в случае удачного завершения,

0 - в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. При использовании метода ksAttributeObject::ksGetSizeAttrRow, не использующего Unicode, размер данных строкового типа будет рассчитываться для unicode строк, как для ANSI строк.

ksGetSizeAttrValue – Получить размер данных ячейки

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данного метода при использовании API экспортных функций - ksGetSizeAttrValue

Синтаксис Automation:

long ksGetSizeAttrValue (reference pAttr,

long columnNumb,

long* count);

Входные параметры:

pAttr - указатель на атрибут,
columnNumb - номер колонки.

Выходной параметр:

count - число ячеек с учетом записей.

Возвращаемое значение:

длина строки ячейки атрибута - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,

- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. При использовании данного метода, не использующего Unicode, размер данных строкового типа будет рассчитываться для unicode строк, как для ANSI строк.

ksGetSizeAttrValueW – Получить размер данных ячейки (Unicode)

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данного метода при использовании API экспортных функций - ksGetSizeAttrValueW

Синтаксис Automation:

long ksGetSizeAttrValueW (reference pAttr,

long columnNumb,

long* count);

Входные параметры:

pAttr	- указатель на атрибут,
columnNumb	- номер колонки.

Выходной параметр:

Count	- число ячеек с учетом записей.

Возвращаемое значение:

длина строки ячейки атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. При использовании метода ksAttributeObject::ksGetSizeAttrValue, не использующего Unicode, размер данных строкового типа будет рассчитываться для unicode строк, как для ANSI строк.

ksSetAttrRow – Установить данные строки в таблице атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ksSetAttrRow.

Синтаксис Automation:

long ksSetAttrRow (reference pAttr,

long rowNumb,

LPDISPATCH flagVisible,

LPDISPATCH columnKeys,

LPDISPATCH value,

BSTR password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего флаги видимости ячеек строки,
columnKeys	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего ключи строки,
value	- указатель на интерфейс ksUserParam, откуда копируются данные,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Смотрите также:ksLtVariant

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

2. Указатель flagVisible может быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_LtVariant, методов интерфейса ksLtVariant, ksDynamicArray.

3. Указатель value может быть получен от интерфейса KompasObject с помощью метода ompasObject::GetParamStruct с параметром ko_UserParam с последующим наполнением массива с помощью методов интерфейса ksUserParam, ksLtVariant, ksDynamicArray.

ksSetAttrType – Изменить тип атрибута в библиотеке или текущем документе

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Аналог данного метода при использовании API экспортных функций - ksSetAttrType.

Синтаксис Automation:

double ksSetAttrType (double attrID,

BSTR libName,

LPDISPATCH attrType,

BSTR password);

Входные параметры:

attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута изменяется в текущем документе,
attrType	- указатель на интерфейс параметров типа табличного атрибута ksAttributeTypeParam,
password	- пароль типа атрибута.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid или с помощью метода ksAttributeObject::ksGetAttrKeysInfo. Тип атрибута типа должен быть предварительно создан, например, с помощью метода ksAttributeObject::ksCreateAttrType.
2. Перед использованием указатель attrType должен быть получен с помощью метода KompasObject::GetParamStruct с параметром ko_AttributeType с последующим наполнением параметрами с помощью свойств и методов интерфейса ksAttributeTypeParam.

ksSetAttrValue – Установить значение ячейки в таблице атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании API экспортных функций - ksSetAttrValue.

Синтаксис Automation:

long ksSetAttrValue (reference pAttr,

long rowNumb,

long columnNumb,

LPDISPATCH

flagVisible,

LPDISPATCH columnKeys,

LPDISPATCH value,

BSTR password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,
flagVisible	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего флаги видимости ячеек,
columnKeys	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray, указывающего ключи полей колонки,
value	- указатель на интерфейс ksUserParam, откуда копируется значение ячейки,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Смотрите также:ksLtVariant

Примечание:

1. Для нетабличного атрибута номер колонки равен нулю.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

3. Перед использованием указатель flagVisible должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_LtVariant, методов интерфейса ksLtVariant, интерфейса ksDynamicArray, метода ksDynamicArray::ksSetArrayItem, ksDynamicArray::ksAddArrayItem.
4. Перед использованием указатель value должен быть получен от интерфейса KompasObject с помощью метода KompasObject::GetParamStruct с параметром ko_UserParam с последующим наполнением массива с помощью методов интерфейса ksUserParam, ksLtVariant, ksDynamicArray.

ksViewEditAttr – Вывести диалог для просмотра или редактирования атрибута

Интерфейс...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_ATR_LST_BODY_ATR.htm`

Аналог данного метода при использовании API экспортных функций - ViewEditAttr.

Синтаксис Automation:

long ksViewEditAttr (reference pAttr,

long type,

BSTR password);

Входные параметры:

pAttr	- указатель на атрибут,
type	- режим работы: - 1 - режим просмотра, 2 - режим редактирования
password	- пароль типа атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

1. Диалог "привязан" к главному окну.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью интерфейса итератора ksIterator и методов ksIterator::ksCreateAttrIterator, ksIterator::ksMoveAttrIterator,
- с помощью методов создания атрибутов ksAttributeObject::ksCreateAttr или выбора атрибутов ksAttributeObject::ksChoiceAttr.

ksViewEditAttrType – Вывести диалог для просмотра или редактирования типа атрибута

Интерфейс...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_ATR_NEW_TYPE.htm

Аналог данного метода при использовании API экспортных функций - ViewEditAttrType.

Синтаксис Automation:

long ksViewEditAttrType (BSTR libName,

long type,

double attrID,

BSTR password);

Входные параметры:

libName	- имя библиотеки типов атрибутов. Если libname = NULL, типы атрибутов берутся из текущего документа,
type	- режим работы: 1 - режим просмотра, 2 - режим редактирования,
attrId	- уникальный номер типа атрибута,
password	- пароль типа атрибута.

Возвращаемое значение:

1	- в случае удачного завершения.

Примечание:

Уникальный номер типа атрибута attrID может быть получен от интерфейса ksLibraryAttrTypeParam с помощью свойства ksLibraryAttrTypeParam::typeid или от интерфейса ksAttributeObject с помощью метода ksAttributeObject::ksGetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью метода ksAttributeObject::ksCreateAttrType.

Интерфейсы параметров атрибутов

Колонка табличного атрибута (Интерфейс ksColumnInfoParam)

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Интерфейс параметров столбца табличного атрибута.

Аналог данных параметров при использовании API экспортных функций - ColumnInfo.

Примечание:

1. Перед использованием указатель на интерфейс должен быть определен с помощью метода KompasObject::GetParamStruct с параметром ko_ColumnInfoParam.
2. Параметры интерфейса могут быть получены с помощью методов ksAttributeObject::ksGetAttrColumnInfo, ksAttributeTypeParam::GetColumns.

Смотрите также KompasObject

ksColumnInfoParam – свойства

def – Значение в колонке по умолчанию

Интерфейс...

Тип данных: строка

Синтаксис Automation:

def = iColumnInfoParam.def	Получить свойство (*)
iColumnInfoParam.def = def	Установить свойство (*)
def = iColumnInfoParam.GetDef()	Получить свойство (**)
iColumnInfoParam.SetDef(def)	Установить свойство (**)

flagEnum – Признак режима, в котором значение поля атрибута заполняется из массива предопределенных значений

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

flagEnum = iColumnInfoParam.flagEnum	Получить свойство (*)
iColumnInfoParam.flagEnum = flagEnum	Установить свойство (*)
flagEnum = iColumnInfoParam.GetFlagEnum()	Получить свойство (**)
iColumnInfoParam.SetFlagEnum(flagEnum)	Установить свойство (**)

Значения свойства:

TRUE	- режим включен,
FALSE	- режим выключен.

header – Заголовок-комментарий типа атрибута

Интерфейс...

Тип данных: строка

Синтаксис Automation:

header = iColumnInfoParam.header	Получить свойство (*)
iColumnInfoParam.header = header	Установить свойство (*)
header = iColumnInfoParam.GetHeader()	Получить свойство (**)
iColumnInfoParam.SetHeader(header)	Установить свойство (**)

key – Дополнительный признак ("ключ"), который позволит отличить две переменные одинакового типа

Интерфейс...

Тип данных: short

Синтаксис Automation:

key = iColumnInfoParam.key	Получить свойство (*)
iColumnInfoParam.key = key	Установить свойство (*)
key = iColumnInfoParam.GetKey()	Получить свойство (**)
iColumnInfoParam.SetKey(key)	Установить свойство (**)

type –Тип данных в столбце

Интерфейс...

Тип данных: short

Синтаксис Automation:

type = iColumnInfoParam.type	Получить свойство (*)
iColumnInfoParam.type = type	Установить свойство (*)
type = iColumnInfoParam.GetType()	Получить свойство (**)
iColumnInfoParam.SetType(type)	Установить свойство (**)

Типы данных в столбце табличного атрибута...

ksColumnInfoParam – методы

GetColumns – Получить указатель на динамический массив информации о колонках типа ATTR_COLUMN_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetColumns();

Возвращаемое значение:

- указатель на динамический массив типа ATTR_COLUMN_ARR ksDynamicArray.

Смотрите также ksColumnInfoParam

GetFieldEnum – Получить указатель на интерфейс динамического массива перечислений (строки) типа CHAR_STR_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFieldEnum();

Возвращаемое значение:

- указатель на интерфейс динамического массива перечислений (строки) типа CHAR_STR_ARR ksDynamicArray.

Смотрите также ksChar255

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Динамические массивы не создаются.

SetColumns – Установить указатель на динамический массив информации о колонках типа ATTR_COLUMN_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetColumns (LPDISPATCH columns);

Входной параметр:

columns - указатель на динамический массив типа ATTR_COLUMN_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также ksColumnInfoParam

SetFieldEnum – Установить указатель на интерфейс динамического массива перечислений (строки) типа CHAR_STR_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetFieldEnum(LPDISPATCH fieldEnum);

Входной параметр:

fieldEnum	- указатель на интерфейс динамического массива перечислений (строки) типа CHAR_STR_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также ksChar255

Тип атрибута в библиотеке (Интерфейс ksLibraryAttrTypeParam)

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Интерфейс параметров типа атрибута в библиотеке типов атрибутов.

Аналог данных параметров при использовании API экспортных функций - LibraryAttrTypeParam.

Примечание:

1. Перед использованием указатель на интерфейс должен быть определен с помощью метода ompasObject::GetParamStruct с параметром ko_LibraryAttrTypeParam.
2. Параметры интерфейса могут быть получены с помощью метода ksAttributeObject::ksGetLibraryAttrTypesArray.
3. Уникальный номер типа атрибута typeid может быть получен с помощью метода ksAttributeObject::ksGetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksAttributeObject::ksCreateAttrType.

Смотрите также KompasObject

ksLibraryAttrTypeParam – свойства

Name – Имя типа атрибута

Интерфейс...

Тип данных: строка

Синтаксис Automation:

name = iLibraryAttrTypeParam.name	Получить свойство (*)
iLibraryAttrTypeParam.name = name	Установить свойство (*)
name = iLibraryAttrTypeParam.GetName()	Получить свойство (**)
iLibraryAttrTypeParam.SetName(name)	Установить свойство (**)

typeId – Номер типа атрибута в библиотеке

Интерфейс...

Тип данных: double

Синтаксис Automation:

typeId = iLibraryAttrTypeParam.typeId	Получить свойство (*)
iLibraryAttrTypeParam.typeId = typeId	Установить свойство (*)
typeId = iLibraryAttrTypeParam.GetTypeId()	Получить свойство (**)
iLibraryAttrTypeParam.SetTypeId(typeId)	Установить свойство (**)

ksLibraryAttrTypeParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Метод обнуляет все параметры.

Тип табличного атрибута (Интерфейс ksAttributeTypeParam)

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Интерфейс параметров типа атрибута.

Аналог данных параметров при использовании API экспортных функций - ksAttributeType.

Примечание:

1. Перед использованием указатель на интерфейс должен быть определен с помощью метода KompasObject::GetParamStruct с параметром ko_AttributeType.
2. Параметры интерфейса могут быть получены с помощью метода ksAttributeObject::ksGetAttrType.
3. При создании типа атрибута ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам ksIterator::ksCreateAttrIterator. При присвоении значений ключам рекомендуется для ключей key1, key3 присваивать код, идентифицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ЗАО "АСКОН".
4. Значения ключей атрибута могут быть получены с помощью метода ksAttributeObject::ksGetAttrKeysInfo.

Смотрите также KompasObject

ksAttributeTypeParam – свойства

flagVisible – Признак видимости колонки в таблице атрибута

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

flagVisible = iAttributeTypeParam.flagVisible	Получить свойство (*)
iAttributeTypeParam.flagVisible = flagVisible	Установить свойство (*)
flagVisible = iAttributeTypeParam.GetFlagVisible()	Получить свойство (**)
iAttributeTypeParam.SetFlagVisible(flagVisible)	Установить свойство (**)

Значения свойства:

TRUE	- невидимый атрибут,
FALSE	- видимый атрибут.

header – Заголовок-комментарий типа атрибута

Интерфейс...

Тип данных: строка

Синтаксис Automation:

header = iAttributeTypeParam.header	Получить свойство (*)

iAttributeTypeParam.header = header	Установить свойство (*)
header = iAttributeTypeParam.GetHeader()	Получить свойство (**)
iAttributeTypeParam.SetHeader(header)	Установить свойство (**)

key1, key3 – Ключи атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

key1 = iAttributeTypeParam.key1	Получить свойство (*)
iAttributeTypeParam.key1 = key1	Установить свойство (*)
key1 = iAttributeTypeParam.GetKey1()	Получить свойство (**)
iAttributeTypeParam.SetKey1(key1)	Установить свойство (**)

Примечание:

Данные ключи рекомендуются как код разработчика.

key2 – Код атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

key2 = iAttributeTypeParam.key2	Получить свойство (*)
iAttributeTypeParam.key2 = key2	Установить свойство (*)
key2 = iAttributeTypeParam.GetKey2()	Получить свойство (**)
iAttributeTypeParam.SetKey2(key2)	Установить свойство (**)

key4 – Системный код атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

key4 = iAttributeTypeParam.key4	Получить свойство (*)
iAttributeTypeParam.key4 = key4	Установить свойство (*)
key4 = iAttributeTypeParam.GetKey4()	Получить свойство (**)
iAttributeTypeParam.SetKey4(key4)	Установить свойство (**)

Примечание:

Значения параметра key4 от 0 до 1000 зарезервированы за ОАО "АСКОН".

Password – Пароль типа атрибута

Интерфейс...

Тип данных: строка

Синтаксис Automation:

password = iAttributeTypeParam.password	Получить свойство (*)
iAttributeTypeParam.password = password	Установить свойство (*)
password = iAttributeTypeParam.GetPassword()	Получить свойство (**)
iAttributeTypeParam.SetPassword(password)	Установить свойство (**)

Примечание:

Если строка не пустая, то пароль защищает тип атрибута от несанкционированного изменения.

rowsCount – Количество строк в таблице атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

rowsCount = iAttributeTypeParam.rowsCount	Получить свойство (*)
iAttributeTypeParam.rowsCount = rowsCount	Установить свойство (*)
rowsCount = iAttributeTypeParam.GetRowsCount()	Получить свойство (**)
iAttributeTypeParam.SetRowsCount(rowsCount)	Установить свойство (**)

ksAttributeTypeParam – методы

GetColumns – Получить указатель на интерфейс динамического массива колонок атрибутов типа ATTR_COLUMN_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetColumns();

Возвращаемое значение:

- указатель на интерфейс динамического массива колонок атрибутов типа ATTR_COLUMN_ARR ksDynamicArray.

Смотрите также ksColumnInfoParam

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры типа атрибута.
2. Создается динамический массив колонок типа ATTR_COLUMN_ARR.

SetColumns – Установить указатель на интерфейс динамического массива колонок атрибутов типа ATTR_COLUMN_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetColumns (LPDISPATCH columns);

Входной параметр:

columns	- указатель на интерфейс динамического массива колонок атрибутов типа ATTR_COLUMN_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также ksColumnInfoParam

Табличный атрибут (Интерфейс ksAttributeParam)

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Интерфейс параметров атрибута.

Аналог данных параметров при использовании API экспортных функций - ksAttribute.

Примечание:

1. Перед использованием указатель на интерфейс должен быть определен с помощью метода KompasObject::GetParamStruct с параметром ko_Attribute.
2. При создании типа атрибута ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам ksIterator::ksCreateAttrIterator. При присвоении значений ключам рекомендуется для ключей key1, key3 присваивать код, иденти-

фицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ЗАО "АСКОН".

3. Значения ключей атрибута могут быть получены с помощью метода ksAttributeObject::ksGetAttrKeysInfo.

Смотрите также KompasObject

ksAttributeParam – свойства

Key1, key3 – Ключи атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

key1 = iAttributeTypeParam.key1	Получить свойство (*)
iAttributeTypeParam.key1 = key1	Установить свойство (*)
key1 = iAttributeTypeParam.GetKey1()	Получить свойство (**)
iAttributeTypeParam.SetKey1(key1)	Установить свойство (**)

Примечание:

Данные ключи рекомендуются как код разработчика.

Key2 – код атрибута key2

Интерфейс...

Тип данных: long

Синтаксис Automation:

key2 = iAttributeTypeParam.key2	Получить свойство (*)
iAttributeTypeParam.key2 = key2	Установить свойство (*)
key2 = iAttributeTypeParam.GetKey2()	Получить свойство (**)
iAttributeTypeParam.SetKey2(key2)	Установить свойство (**)

Key4 – Системный код атрибута

Интерфейс...

Тип данных: long

Синтаксис Automation:

key4 = iAttributeTypeParam.key4	Получить свойство (*)
iAttributeTypeParam.key4 = key4	Установить свойство (*)
key4 = iAttributeTypeParam.GetKey4()	Получить свойство (**)
iAttributeTypeParam.SetKey4(key4)	Установить свойство (**)

Примечание:

Значения параметра key4 от 0 до 1000 зарезервированы за ЗАО "АСКОН".

Password – Пароль типа атрибута

Интерфейс...

Тип данных: строка

Синтаксис Automation:

password = iAttributeTypeParam.password	Получить свойство (*)
iAttributeTypeParam.password = password	Установить свойство (*)
password = iAttributeTypeParam.GetPassword()	Получить свойство (**)
iAttributeTypeParam.SetPassword(password)	Установить свойство (**)

Примечание:

Если строка не пустая, то пароль защищает тип атрибута от несанкционированного изменения.

ksAttributeParam – методы

GetColumnKeys – Получить указатель на интерфейс динамического массива ключей колонок атрибута типа LTVARIANT_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetColumnKeys();

Возвращаемое значение:

- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray.

Смотрите также ksLtVariant

GetFlagVisible – Получить указатель на интерфейс динамического массива типа LTVARIANT_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetFlagVisible();

Возвращаемое значение:

- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray.

Примечание:

Интерфейс динамического массива ksDynamicArray определяет для каждой колонки атрибута видимость-невидимость.

Смотрите также

ksLtVariant

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечания:

1. Метод обнуляет все параметры.
2. Динамические массивы не создаются.

GetValues – Получить указатель на интерфейс массива значений ячеек таблицы атрибутов ksUserParam

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetValues();

Возвращаемое значение:

- указатель на интерфейс ksUserParam.

SetColumnKeys – Установить указатель на интерфейс динамического массива ключей колонок атрибута типа LTVARIANT_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetColumnKeys (LPDISPATCH value);

Входной параметр:

value	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Смотрите также ksLtVariant

SetFlagVisible – Установить указатель на интерфейс динамического массива типа LTVARIANT_ARR – ksDynamicArray

Интерфейс...

Синтаксис Automation:

BOOL SetFlagVisible (LPDISPATCH value);

Входной параметр:

value	- указатель на интерфейс динамического массива типа LTVARIANT_ARR ksDynamicArray.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Примечание:

Интерфейс динамического массива ksDynamicArray определяет для каждой колонки атрибута видимость-невидимость.

Смотрите также ksLtVariant

SetValues – Заполнить массив значений ячеек таблицы атрибутов ksUserParam

Интерфейс...

Синтаксис Automation:

BOOL SetValues (LPDISPATCH value);

Входной параметр:

value	- указатель на интерфейс массива значений ячеек таблицы атрибутов ksUserParam.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Параметры цвета фона (Интерфейс ksViewColorParam)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WINDOW3DCOLOR_SETUP.htm

Интерфейс параметров цвета фона.

ksViewColorParam - интерфейс Automation

Аналог данных параметров при использовании API экспортных функций - ViewColorParam.

Примечание:

Данный интерфейс может быть получен с использованием метода интерфейса KompasObject::ksGetSysOptions. Значение параметра optionsType = VIEWCOLOR_OPTIONS.

ViewColorParam – свойства

BottomColor – Нижний цвет перехода

Интерфейс...

Тип данных: long

Синтаксис Automation:

bottomColor = iViewColorParam.bottomColor	Получить свойство (*)
iViewColorParam.bottomColor = bottomColor	Установить свойство (*)
bottomColor = iViewColorParam.GetBottomColor()	Получить свойство (**)
iViewColorParam.SetBottomColor (bottomColor)	Установить свойство (**)

Примечание:

Свойство доступно только для документов-моделей.

Color – Цвет фона

Интерфейс...

Тип данных: long

Синтаксис Automation:

color = iViewColorParam.color	Получить свойство (*)
iViewColorParam.color = color	Установить свойство (*)
color = iViewColorParam.GetColor()	Получить свойство (**)
iViewColorParam.SetColor(color)	Установить свойство (**)

Значения свойства:

Цвет фона
-1 - если используется цвет окна, установленный в Windows.

TopColor – Верхний цвет перехода

Интерфейс...

Тип данных: long

Синтаксис Automation:

```
topColor = iViewColorParam.topColor                  Получить свойство (* )
iViewColorParam.topColor = topColor                  Установить свойство (* )
topColor = iViewColorParam.GetTopColor()             Получить свойство (**)
iViewColorParam.SetTopColor (topColor)               Установить свойство (**)
```

Примечание:

Свойство доступно только для документов-моделей.

UseGradient – Использовать градиентный переход при полутоновом отображении

Интерфейс...

Тип данных: BOOL

Синтаксис Automation:

```
useGradient = iViewColorParam.useGradient                  Получить свойство (* )
iViewColorParam.useGradient = useGradient                  Установить свойство (* )
useGradient = iViewColorParam.GetUseGradient()             Получить свойство (**)
iViewColorParam.SetUseGradient(useGradient)                Установить свойство (**)
```

Значения свойства:

TRUE - использовать градиентный переход,
FALSE - не использовать.

Примечание:

Свойство доступно только для документов-моделей.

ksViewColorParam – методы

Init – Инициализировать параметры

Интерфейс...

Синтаксис Automation:

BOOL Init();

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Параметры конвертации при сохранении в предыдущую версию (Интерфейсы ISaveToPreviusParam, ksSaveToPreviusParam)

Интерфейс параметров конвертации.

ksSaveToPreviusParam	- интерфейс Automation
ISaveToPreviusParam	- интерфейс COM

Интерфейс используется в событиях ksKompasObjectNotify/IKompasObjectNotify сохранения документа в предыдущую версию

КОМПАС версия v18

ksSaveToPreviusParam – методы

AddOption – Добавить настройку конвертации с возможностью выбора варианта конвертации

Интерфейс...

Синтаксис:

BOOL AddOption(BSTR uniqueID, BSTR optionName, VARIANT options, BSTR defaultValue);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

uniqueID	- уникальный идентификатор,
optionName	- отображаемое имя свойства,
options .	- список возможных значений (массив строк SafeArray VT_ARRAY I VTBSTR).

AddWarning – Добавить предупреждение

Интерфейс...

Синтаксис:

BOOL AddWarning(BSTR uniqueID, BSTR optionName, BSTR text);

Возвращаемое значение:

TRUE	- в случае успешного завершения.

Входные параметры:

uniqueID	- уникальный идентификатор,
optionName	- отображаемое имя свойства,.
text	- текст предупреждения.

GetCurrentOptionValue – Получить текущее значение, выбранное в диалоге параметров конвертации

Интерфейс...

Синтаксис:

BSTR GetCurrentOptionValue(BSTR uniqueID);

Возвращаемое значение:

	- текущее значение свойства.

Входные параметры:

uniqueID	- уникальный идентификатор.

Интерфейсы событий API5

События в КОМПАС
Общие сведения

Генерация событий в КОМПАС осуществляется при помощи компактных интерфейсов событий, в которых события группируются по тематике. Подписка на события проводится во время подключения библиотеки при помощи предопределенной функции LibInterfaceNotifyEntry().

Для Automation реализация основана на стандартном механизме ConnectionPoint. Основные объекты, такие как KompasObject, ksDocument2D и т.п., которые являются источниками событий, наследуют интерфейс IConnectionPointContainer, позволяют подписаться на один основной интерфейс и предоставляют дополнительные объекты-источники событий. Подписаться на события или отменить подписку можно стандартным способом при помощи методов интерфейса IConnectionPoint: IConnectionPoint::Advise и IConnectionPoint::Unadvise.

Для COM реализация основана на интерфейсах IUnknown, которые предоставляет библиотека. Подписаться на события или отменить подписку можно при помощи экспортных функций: ksConnectionAdvise и ksConnectionUnadvise.

Библиотека сама должна позаботиться о своевременной отписке от неактуальных событий. Например, в момент закрытия документа нужно обработать соответствующее событие и отписаться от всех событий, связанных с данным документом (см. таблицу связей). При закрытии приложения следует отписаться от всех событий.

Интерфейс событий приложения...

Интерфейс событий документа, работа с файлом...

Интерфейс событий документа-модели...

Интерфейс событий графического документа...

Интерфейс событий менеджера выделенных объектов...

Интерфейс событий объектов графического документа...

Интерфейс событий объектов документа-модели...

Интерфейс событий основной надписи графического документа...

Интерфейс событий объекта спецификации...

Интерфейс событий спецификации...

Интерфейс событий документа-спецификации...

Связь объектов системы с интерфейсами событий...

Интерфейс событий приложения ksKompasObjectNotify/ IKompasObjectNotify

ksKompasObjectNotify - интерфейс Automation
IKompasObjectNotify - интерфейс COM

Позволяют контролировать состояние приложения.

В Automation источником событий для подписки являются:

- для API интерфейсов версии 5 - объект KompasObject.
- для API интерфейсов версии 7 - объект IApplication.

В COM источника нет.

ksKompasObjectNotify, IKompasObjectNotify – события

ApplicationDestroy – Закрытие приложения

Интерфейс...

Синтаксис Automation:

BOOL ApplicationDestroy();

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL ApplicationDestroy();

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....
3. При обработке данного события необходимо отписаться от всех событий. Библиотека должна сама следить за отпиской событий.

BeginCloseAllDocument – Начало закрытия всех открытых документов

Интерфейс...

Синтаксис Automation:

BOOL BeginCloseAllDocument();

Синтаксис COM:

BOOL BeginCloseAllDocument();

Возвращаемое значение:

TRUE - закрытие всех открытых документов,
FALSE - закрытие всех открытых документов запрещено.

Примечание:

Индекс события задан в перечислении событий приложения....

BeginCreate – Начало создания документа (до диалога выбора типа)

Интерфейс...

Синтаксис Automation:

BOOL BeginCreate (long docType);

Синтаксис COM:

BOOL BeginCreate (long docType);

Входные параметры:

docName	- тип документа,
0	- если пользователь запросил диалог создания документа.

Возвращаемое значение:

TRUE	- создать документ,
FALSE	- создание документа запрещено.

Примечание:

Источником события является интерфейс KompasObject.

BeginOpenDocument – Начало открытия документа

Интерфейс...

Синтаксис Automation:

BOOL BeginOpenDocument (BSTR docName);

Входные параметры:

docName	- имя файла документа.

Возвращаемое значение:

TRUE	- открыть документ,
FALSE	- не открывать документ.

Синтаксис COM:

BOOL BeginOpenDocument (char * docName);

Входные параметры:

docName	- имя файла документа.

Возвращаемое значение:

TRUE - открыть документ,
FALSE - не открывать документ.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

BeginOpenFile – Начало открытия документа (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginOpenFile();

Синтаксис COM:

BOOL BeginOpenFile();

Возвращаемое значение:

TRUE - вызвать диалог открытия документа,
FALSE - вызов диалога открытия документа запрещен.

Примечание:

Индекс события задан в перечислении событий приложения....

BeginRequestFiles – Запрос имен файлов

Интерфейс...

Синтаксис Automation:

BOOL BeginRequestFiles(long requestID, VARIANT * files);

Синтаксис COM:

BOOL BeginRequestFiles(long requestID, VARIANT * files);

Входные параметры:

requestID - тип запроса файлов из перечисления ksRequestFilesTypeEnum.

Выходные параметры:

files - имя файла или список SafeArray файлов.

Возвращаемое значение:

TRUE - использовать стандартный диалог выбора файлов,

FALSE - если список файлов задан - использовать файл или файлы из списка,
если список файлов не задан - отмена выбора файлов.

Примечание:

1. Если выбран один файл, то в VARIANT его имя можно положить как строку, тип VARIANT-а - VT_BSTR. Если выбрано несколько файлов, то их имена в VARIANT нужно положить как массив строк тип VARIANT-а - VT_ARRAY | VT_BSTR.
2. Возможность множественного выбора и допустимые расширения файлов зависят от типа запроса файлов.
3. В данных событиях в отличие от других событий начала какого либо действия требуется обработка трех состояний:

- отказ пользователя или обработка события выполнена полностью на стороне библиотеки - событие возвращает FALSE;
- событие возвращает TRUE и имя файла для продолжения выполнения команды - процесс продолжается без запуска диалога выбора файла или файлов;
- событие возвращает TRUE, но имя файла не возвращается - запускается стандартный диалог выбора файла из системы КОМПАС.

ChangeActiveDocument – Переключение на другой активный документ

Интерфейс...

Синтаксис Automation:

BOOL ChangeActiveDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc - указатель на интерфейс IDispatch открытого документа,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL ChangeActiveDocument(long pDoc, int docType);

Входные параметры:

pDoc - указатель на открытый документ,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

CreateDocument – Документ создан

Интерфейс...

Синтаксис Automation:

BOOL CreateDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc - указатель на интерфейс IDispatch созданного документа,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL CreateDocument (long pDoc, int docType);

Входные параметры:

pDoc - указатель на созданный документ,
docType - тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

KeyDown – Клавиша нажата и удерживается нажатой

Интерфейс...

Синтаксис Automation:

BOOL KeyDown (long * key, long flags, BOOL sysKey);

Синтаксис COM:

BOOL KeyDown(long * key, long flags, BOOL sysKey);

Входные параметры:

key	- код нажатой клавиши,
flags	- флаг клавиатуры,
sysKey	- признак нажатия системной клавиши.

Выходные параметры:

key	- измененный код нажатой клавиши.

Возвращаемое значение:

TRUE	- разрешить обработку нажатой клавиши,
FALSE	- запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения...
2. Благодаря автоматическому повторению кода клавиши при удержании ее нажатой, до появления события wm_keyup может выдаваться несколько событий wm_keydown. Значение разряда 30 параметра flags позволяет определить, было ли событие wm_keydown первым или является повторным во время удержания клавиши нажатой.
3. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

KeyPress – Одиночное нажатие клавиши

Интерфейс...

Синтаксис Automation:

BOOL KeyPress (long * key, BOOL sysKey);

Синтаксис COM:

BOOL KeyPress (long * key, BOOL sysKey);

Входные параметры:

key	- код нажатой клавиши,
flags	- флаг клавиатуры,
sysKey	- признак нажатия системной клавиши.

Выходные параметры:

key	- измененный код клавиши.

Возвращаемое значение:

TRUE - разрешить обработку нажатой клавиши,
FALSE - запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения....

   Если syskey = false, была нажата обычная клавиша, что соответствует событию wm_keyup, если syskey = true, была нажата системная клавиша, что соответствует событию wm_syskeyup.

2. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

KeyUp – Клавиша отпущена

Интерфейс...

Синтаксис Automation:

BOOL KeyUp (long * key, long flags, BOOL sysKey);

Синтаксис COM:

BOOL KeyUp (long * key, long flags, BOOL sysKey);

Входные параметры:

key - код нажатой клавиши,
flags - флаг клавиатуры,
sysKey - признак нажатия системной клавиши,

Выходные параметры:

key - измененный код нажатой клавиши.

Возвращаемое значение:

TRUE - разрешить обработку нажатой клавиши,
FALSE - запретить обработку нажатой клавиши.

Примечание:

1. Индекс события задан в перечислении событий приложения....
2. Если syskey = false, была нажата обычная клавиша, что соответствует событию wm_keyup, если syskey = true, была нажата системная клавиша, что соответствует событию wm_syskeyup.
3. Если библиотека обработала событие нажатия клавиши и нужно запретить обработку события другими подписчиками, то нужно обнулить код клавиши т.е. *key = 0 и вернуть FALSE, если нужно запретить обработку события системой КОМПАС.

OpenDocument – Документ открыт

Интерфейс...

Синтаксис Automation:

BOOL OpenDocument (LPDISPATCH pDoc, long docType);

Входные параметры:

pDoc	- указатель на интерфейс IDispatch открытого документа,
docType	- тип документа DocType.

Возвращаемое значение:

- Не используется.

Синтаксис COM:

BOOL OpenDocument(long pDoc, int docType);

Входные параметры:

pDoc	- указатель на открытый документ,
docType	- тип документа DocType.

Возвращаемое значение:

- Не используется.

Примечание:

1. Источником события является интерфейс KompasObject.
2. Индекс события задан в перечислении событий приложения....

ksKompasObjectNotify/ IKompasObjectNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Интерфейс...

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий документа; работа с файлом ksDocumentFileNotify/IDocumentFileNotify

ksDocumentFileNotify	- интерфейс Automation
IDocumentFileNotify	- интерфейс COM

События интерфейса позволяют контролировать состояние документа.

Источники событий для подписки:

- API 7,
- API 5.

ksDocumentFileNotify/IDocumentFileNotify – события

Activate – Документ активизирован

Интерфейс...

Синтаксис Automation:

BOOL Activate();

Синтаксис COM:

BOOL Activate();

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

В COM источником для подписки на данный интерфейс является указатель на документ.

AutoSaveDocument – Документ автосохранен

Интерфейс...

Синтаксис Automation:

BOOL AutoSaveDocument();

Синтаксис COM:

BOOL AutoSaveDocument();

Возвращаемое значение:

Не используется

BeginAutoSaveDocument – Начало автосохранения документа

Интерфейс...

Синтаксис Automation:

BOOL BeginAutoSaveDocument();

Синтаксис COM:

BOOL BeginAutoSaveDocument();

Возвращаемое значение:

Не используется

BeginCloseDocument – Начало закрытия документа

Интерфейс...

Синтаксис Automation:

BOOL BeginCloseDocument();

Синтаксис COM:

BOOL BeginCloseDocument();

Возвращаемое значение:

TRUE	- закрыть документ,
FALSE	- документ закрывать нельзя.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

▼ ksDocument2D (чертеж, фрагмент),

- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

3. В COM источником событий для подписки на данный интерфейс является указатель на документ.

BeginProcess – Начало процесса

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess(long Id);

Синтаксис COM:

BOOL BeginProcess(long Id);

BeginSaveAsDocument – Начало сохранения документа с другим именем (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginSaveAsDocument();

Синтаксис COM:

BOOL BeginSaveAsDocument();

Возвращаемое значение:

TRUE	- сохранить документ,
FALSE	- документ сохранить нельзя.

Примечание:

1. Индекс события задан в перечислении событий ksDocumentFileNotifyEnum.
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

3. В COM источником событий для подписки на данный интерфейс является указатель на документ.

BeginSaveDocument – Начало сохранения документа

Интерфейс...

Синтаксис Automation:

BOOL BeginSaveDocument (BSTR docName);

Синтаксис COM:

BOOL BeginSaveDocument (char * docName);

Входные параметры:

docName - имя файла документа, с которым он будет сохранен.

Возвращаемое значение:

TRUE - сохранить документ,
FALSE - не сохранять документ.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

3. В COM источником для подписки на данный интерфейс является указатель на документ.

CloseDocument – Документ закрыт

Интерфейс...

Синтаксис Automation:

BOOL CloseDocument();

Синтаксис COM:

BOOL CloseDocument();

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),

- ksDocument3D (документ-модель).

3. В COM источником для подписки на данный интерфейс является указатель на документ.
4. При обработке данного события необходимо отписаться от всех событий, связанных с закрываемым документом. Библиотека должна сама следить за отпиской событий.

Deactivate – Документ деактивирован

Интерфейс...

Синтаксис Automation:

BOOL Deactivate();

Синтаксис COM:

BOOL Deactivate();

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

В COM источником для подписки на данный интерфейс является указатель на документ.

DocumentFrameOpen – Окно документа открылось

Интерфейс..

Синтаксис Automation:

BOOL DocumentFrameOpen (LPDISPATCH v);

Синтаксис COM:

BOOL DocumentFrameOpen (LPUNKNOWN v);

Входные параметры:

v - указатель на интерфейс IDocumentFrame открытого окна документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий ksDocumentFileNotifyEnum.

EndProcess – Завершение процесса

Интерфейс...

Синтаксис Automation:

BOOL EndProcess(long Id, BOOL Success);

Синтаксис COM:

BOOL EndProcess(long Id, BOOL Success);

ProcessActivate – Процесс активизирован

Интерфейс...

Синтаксис Automation:

BOOL ProcessActivate (long Id);

Синтаксис COM:

BOOL ProcessActivate (long Id);

Входные параметры:

Id - идентификатор активизируемого процесса из перечисления Типы процессов в КОМПАС 3D ProcessTypeEnum.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий окна документа ksDocumentFileNotifyEnum.

ProcessDeactivate – Процесс деактивирован

Интерфейс...

Синтаксис Automation:

BOOL ProcessDeactivate (long Id);

Синтаксис COM:

BOOL ProcessDeactivate (long Id);

Входные параметры:

Id Идентификатор деактивируемого процесса из перечисления Типы процессов в КОМПАС 3D ProcessTypeEnum,

x,y - координаты в экранных пикселях.

Возвращаемое значение:

- Не используется.

Примечание:

Индекс события задан в перечислении событий ksDocumentFileNotifyEnum.

SaveDocument – Документ сохранен

Интерфейс...

Синтаксис Automation:

BOOL SaveDocument();

Синтаксис COM:

BOOL SaveDocument();

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс события задан в перечислении событий документа....
2. В Automation источником событий для подписки на данный интерфейс являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

3. В COM источником для подписки на данный интерфейс является указатель на документ.

ksDocumentFileNotify/IDocumentFileNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейсы событий графического документа ksDocument2DNotify/ IDocument2DNotify

ksDocument2DNotify	- интерфейс Automation
IDocument2Dnotify	- интерфейс COM

Интерфейс позволяет контролировать редактирование графического документа.

В Automation источник событий для подписки Document2DNotify можно получить при помощи метода ksDocument2D::GetDocument2DNotify.

В COM источником для подписки является графический документ.

IDocument2DNotify – события

BeginChoiceMaterial – Начало выбора материала

Интерфейс...

Синтаксис Automation:

BOOL BeginChoiceMaterial();

Синтаксис COM:

BOOL BeginChoiceMaterial();

Возвращаемое значение:

TRUE	- выбрать материал,
FALSE	- выбор материала запрещен.

Примечание:

1. Индекс метода задан в перечислении событий графического документа...
2. Для вызова события необходимо в команде **Расчет МЦХ тел вращения** в диалоге **Свойства объекта** нажать кнопку **Выбрать из справочника**.

BeginChoiceProperty – Начало выбора свойства

Интерфейс...

Синтаксис Automation:

BOOL BeginChoiceProperty();

Синтаксис COM:

BOOL BeginChoiceProperty();

Возвращаемое значение:

TRUE	- выбрать свойство,
FALSE	- выбор свойства запрещен.

Примечание:

Индекс метода задан в перечислении событий графического документа...

BeginInsertFragment – Начало вставки фрагмента (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginInsertFragment();

Синтаксис COM:

BOOL BeginInsertFragment();

Возвращаемое значение:

TRUE	- вставить фрагмент,
FALSE	- вставка фрагмента запрещена.

Примечание:

Индекс метода задан в перечислении событий графического документа...

BeginRebuild – Начало перестроения ассоциативного чертежа

Интерфейс...

Синтаксис Automation:

BOOL BeginRebuild();

Синтаксис COM:

BOOL BeginRebuild();

Возвращаемое значение:

TRUE	- перестроить чертеж,
FALSE	- перестройка чертежа запрещена.

Примечание:

Индекс метода задан в перечислении событий графического документа...

ChoiceMaterial – Закончен выбор материала

Интерфейс...

Синтаксис Automation:

BOOL ChoiceMaterial (BSTR material, double density);

Синтаксис COM:

BOOL ChoiceMaterial(char * material, double density);

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс метода задан в перечислении событий графического документа...

▼ Для вызова события необходимо в команде в диалоге **Свойства объекта**, который появляется на экране после вызова команды **Расчет МЦХ тел вращения**, нажать кнопку **Выбрать из справочника**.

ChoiceProperty – Закончен выбор свойства

Интерфейс...

Синтаксис Automation:

BOOL ChoiceProperty(long objRef, double propID);

Синтаксис COM:

BOOL ChoiceProperty(long objRef, double propID);

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий графического документа...

LocalFragmentEdit – Редактирование локального фрагмента

Интерфейс...

Синтаксис Automation:

BOOL LocalFragmentEdit (LPDISPATCH newDoc, BOOL newFrw);

Синтаксис COM:

BOOL LocalFragmentEdit (long newDoc, BOOL newFrw);

Входные параметры:

newDoc	- указатель на интерфейс документа, в котором создаётся локальный фрагмент,
newFrw	- TRUE - при создании нового фрагмента, - FALSE – при редактировании.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий графического документа...

Rebuild – Ассоциативный чертеж перестроен

Интерфейс...

Синтаксис Automation:

BOOL Rebuild();

Синтаксис COM:

BOOL Rebuild();

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий графического документа...

ksDocument2DNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType - номер события в перечислении событий.

Возвращаемое значение:

TRUE - событие будет обрабатываться,
FALSE - событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий объектов графических документов ksObject2DNotify/ IObject2DNotify

ksObject2DNotify - интерфейс Automation
IObject2DNotify - интерфейс COM

Позволяют контролировать редактирование объектов графического документа.

В Automation источник событий для подписки Object2DNotify можно получить при помощи метода ksDocument2D::GetObject2DNotify.

В COM контейнером для подписки является указатель на графический документ.

Дополнительный параметр при подписке задает объекты, для которых предназначен данный интерфейс событий.

Таким параметром является либо тип объекта (ALL_OBJ...AXISLINE_OBJ, VIEW_OBJ), либо указатель на объект.

Если задан тип объекта, события генерируются для всех объектов этого типа.

Если задан указатель на объект, события генерируются только для этого объекта.

Если задан тип объектов ALL_OBJ, события генерируются для всех объектов.

В процессе обработки события можно получить информацию о редактировании объекта при помощи интерфейса IObject2DNotifyResult для COM или ksObject2DNotifyResult для Automation.

В Automation получить интерфейс результата можно при помощи метода ksDocument2D::GetObject2DNotifyResult.

В COM получить интерфейс результата можно при помощи экспортной функции ksGetObject2DNotifyResult.

IObject2DNotify – события

BeginCopy – Начало копирования объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginCopy (long objRef);

Синтаксис COM:

BOOL BeginCopy(long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- копировать объект,
FALSE	- копирование объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginDelete – Начало удаления объекта

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL BeginDestroy (long objRef);

Синтаксис COM:

BOOL BeginDelete (long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- удалить объект,
FALSE	- удаление объекта запрещено.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Для обеспечения корректной обработки события удаления вида, нужно учитывать следующее:

- при удалении системного вида, собственно сам системный вид не удаляется, удаляются только все объекты в нем;
- номер 0 может иметь служебный вид, создаваемый КОМПАС для режима визуального редактирования макроэлементов, это можно проверить по свойству IView1::EditMacroVisibleRegime.

BeginDestroyObject – Начало разрушения объекта

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL BeginDestroyObject(long objRef);

Синтаксис COM:

BOOL BeginDestroyObject(long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

TRUE - разрушить объект,
FALSE - разрушение объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginMove – Начало сдвига объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginMove (long objRef);

Синтаксис COM:

BOOL BeginMove (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

TRUE - сдвинуть объект,
FALSE - сдвиг объекта запрещен.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginProcess – Начало редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess (long pType, long objRef);

Синтаксис COM:

BOOL BeginProcess (long pType, long objRef);

Входные параметры:

pType	- тип объекта (LINESEG_OBJ...AXISLINE_OBJ, VIEW_OBJ)
objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- преобразовать объект,
FALSE	- преобразование объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginPropertyChanged – Начало изменения свойств объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginPropertyChanged(long objRef);

Синтаксис COM:

BOOL BeginPropertyChanged(long objRef);

Входной параметр:

objRef	- указатель на объект.

Возвращаемое значение:

TRUE	- Разрешить изменение свойств объекта,
FALSE	- Запретить изменение свойств объекта.

BeginRotate – Начало поворота объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginMove (long objRef);

Синтаксис COM:

BOOL BeginRotate (long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- повернуть объект,
FALSE	- поворот объекта запрещен.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginScale – Начало масштабирования объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginScale (long objRef);

Синтаксис COM:

BOOL BeginScale (long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- масштабировать объект,
FALSE	- масштабирование объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginSymmetry – Начало симметричного преобразования объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginSymmetry (long objRef);

Синтаксис COM:

BOOL BeginSymmetry (long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- преобразовать объект,
FALSE	- преобразование объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

BeginTransform – Начало трансформации объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginTransform (long objRef);

Синтаксис COM:

BOOL BeginTransform (long objRef);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

TRUE	- трансформировать объект,
FALSE	- трансформация объекта запрещена.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

ChangeActive – Переключение активности объекта (вид, слой)

Интерфейс...

Синтаксис Automation:

BOOL ChangeActive (long objRef);

Синтаксис COM:

BOOL ChangeActive (long objRef);

Входные параметры:

objRef - указатель на вид или слой.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Copy – Завершение копирования объекта

Интерфейс...

Синтаксис Automation:

BOOL Copy (long objRef);

Синтаксис COM:

BOOL Copy (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

CreateObject – Объект создан

Интерфейс...

Синтаксис Automation:

BOOL CreateObject (long objRef);

Синтаксис COM:

BOOL CreateObject (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Delete – Завершение удаления объекта

Интерфейс...

Синтаксис Automation:

VARIANT_BOOL Delete(long objRef);

Синтаксис COM:

BOOL Delete (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Метод используется при редактировании макроэлементов.
3. Для обеспечения корректной обработки события удаления вида, нужно учитывать следующее:

- при удалении системного вида собственно сам системный вид не удаляется, удаляются только все объекты в нем;
- номер 0 может иметь служебный вид, создаваемый КОМПАС для режима визуального редактирования макроэлементов, это можно проверить по свойству IView1::EditMacroVisibleRegime.

DestroyObject – Завершение удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL DestroyObject (long objRef);

Синтаксис COM:

VARIANT_BOOL DestroyObject(VARIANT Objects);

Входные параметры:

objRef	- указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.
2. Разрушить объекты можно см. описание ksDocument2D::ksDestroyObjects.

EndProcess – Завершение редактирования/создания объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (long pType);

Синтаксис COM:

BOOL EndProcess (long pType);

Входные параметры:

pType	- тип объекта (LINESEG_OBJ...AXISLINE_OBJ, VIEW_OBJ)

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Move – Завершение сдвига объекта

Интерфейс...

Синтаксис Automation:

BOOL Move (long objRef);

Синтаксис COM:

BOOL Move (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

PropertyChanged – Изменения свойств объекта

Интерфейс...

Синтаксис Automation:

BOOL PropertyChanged(long objRef);

Синтаксис COM:

BOOLPropertyChanged(long objRef);

Входной параметр:

objRef - указатель на объект.

Возвращаемое значение:

- Не используется.

Rotate – Завершение поворота объекта

Интерфейс...

Синтаксис Automation:

BOOL Rotate (long objRef);

Синтаксис COM:

BOOL Rotate (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Scale – Завершение масштабирования объекта

Интерфейс...

Синтаксис Automation:

BOOL Scale (long objRef);

Синтаксис COM:

BOOL Scale (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Symmetry – Завершение симметричного преобразования объекта

Интерфейс...

Синтаксис Automation:

BOOL Symmetry (long objRef);

Синтаксис COM:

BOOL Symmetry (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

Transform – Завершение трансформации объекта

Интерфейс...

Синтаксис Automation:

BOOL Transform (long objRef);

Синтаксис COM:

BOOL Transform (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

UpdateObject – Объект отредактирован

Интерфейс...

Синтаксис Automation:

BOOL UpdateObject (long objRef);

Синтаксис COM:

BOOL UpdateObject (long objRef);

Входные параметры:

objRef - указатель на объект или группу объектов.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов графических документов ksObject2DNotifyEnum.

ksObject2DNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий документа – модели ksDocument3DNotify/IDocument3DNotify

ksDocument3DNotify	- интерфейс Automation
IDocument3DNotify	- интерфейс COM

Позволяют контролировать редактирование документа-модели.

В Automation источник событий для подписки Document3DNotify можно получить при помощи метода ksDocument3D::GetDocument3DNotify.

В COM источником для подписки является документ-модель.

IDocument3DNotify – события

BeginChoiceMarking – Начало выбора обозначения

Интерфейс...

Синтаксис Automation:

BOOL BeginChoiceMarking();

Синтаксис COM:

BOOL BeginChoiceMarking();

Возвращаемое значение:

TRUE	- выбрать обозначение,
FALSE	- выбор обозначения запрещен.

Примечание:

1. Индекс метода задан в перечислении событий документа-модели....
2. Событие может приходить при редактировании параметров тела.

BeginLoadCombinationChange –Начало переключения типа загрузки

Интерфейс...

Синтаксис Automation:

BOOL BeginLoadCombinationChange(long index);

Синтаксис COM:

BOOL BeginLoadCombinationChange(long index);

Входные параметры:

index	- индекс стандартного или пользовательского набора загрузки.

Возвращаемое значение:

TRUE	- запретить переключение,
FALSE	- разрешить переключение.

Примечание

У сборки есть умолчательные стандартные наборы, индексы которых соответствуют перечислению ksLoadStateEnum. Кроме этого, могут быть еще и пользовательские наборы.

LoadCombinationChange – Завершение переключения типа загрузки

Интерфейс...

Синтаксис Automation :

BOOL LoadCombinationChange(long index);

Синтаксис COM :

BOOL LoadCombinationChange(long index);

Входные параметры:

index - индекс стандартного или пользовательского набора загрузки.

Возвращаемое значение:

- не используется.

Примечание

У сборки есть умолчательные стандартные наборы, индексы которых соответствуют перечислению ksLoadStateEnum. Кроме этого, могут быть еще и пользовательские наборы.

BeginChoiceMaterial – Начало выбора материала

Интерфейс...

Синтаксис Automation:

BOOL BeginChoiceMaterial();

Синтаксис COM:

BOOL BeginChoiceMaterial();

Возвращаемое значение:

TRUE - выбрать материал,
FALSE - выбор материала запрещен.

Примечание:

1. Индекс метода задан в перечислении событий документа-модели....
2. Событие может приходить при редактировании параметров тела.

BeginChoiceProperty – Начало выбора свойства

Интерфейс...

Синтаксис Automation:

BOOL BeginChoiceProperty(LPDISPATCH obj, double propID);

Синтаксис COM:

BOOL BeginChoiceProperty(LPUNKNOWN obj, double propID);

Возвращаемое значение:

TRUE - Разрешить стандартную процедуру выбора свойства,
FALSE - Запретить стандартную процедуру выбора свойства (библиотека самостоятельно обработала выбор свойства).

BeginCreatePartFromFile – Начало создания компонента в сборке (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginCreatePartFromFile (BOOL part, ksEntity * plane);

Синтаксис COM:

BOOL BeginCreatePartFromFile (BOOL part, LPENTITY plane);

Входные параметры:

part	TRUE - вставить деталь, FALSE - вставить сборку,
plane	- плоскость, на которой создается деталь.

Возвращаемое значение:

TRUE	- создать компонент в сборке,
FALSE	- создание компонента в сборке запрещено.

Примечание:

Индекс метода задан в перечислении событий документа-модели....

BeginRebuild – Начало перестроения модели

Интерфейс...

Синтаксис Automation:

BOOL BeginRebuild();

Синтаксис COM:

BOOL BeginRebuild();

Возвращаемое значение:

TRUE	- перестроить модель,
FALSE	- перестройка модели запрещена.

Примечание:

Индекс метода задан в перечислении событий документа-модели....

BeginRollbackFeatures – Начало отката дерева модели

Интерфейс...

Синтаксис Automation:

BOOL BeginRollbackFeatures();

Синтаксис COM:

BOOL BeginRollbackFeatures();

Возвращаемое значение:

FALSE	- запретить откат дерева.

BeginSetPartFromFile – Начало установки компонента в сборку (до диалога выбора имени)

Интерфейс...

Синтаксис Automation:

BOOL BeginSetPartFromFile();

Синтаксис COM:

BOOL BeginSetPartFromFile();

Возвращаемое значение:

TRUE	- установить компонент в сборку,
FALSE	- установка компонента в сборку запрещена.

Примечание:

Индекс метода задан в перечислении событий документа-модели....

ChangeCurrentEmbodiment – Исполнение установлено текущим

Интерфейс...

Синтаксис Automation:

BOOL ChangeCurrentEmbodiment(BSTR marking);

Синтаксис COM:

BOOL ChangeCurrentEmbodiment(LPCSTR marking);

Возвращаемое значение:

	- не используется.

ChoiceMarking – Закончен выбор обозначения

Интерфейс...

Синтаксис Automation:

BOOL ChoiceMarking (BSTR material);

Синтаксис COM:

BOOL ChoiceMarking (char * material);

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий документа-модели....
2. Событие может приходить при редактировании параметров тела.

ChoiceMaterial – Закончен выбор материала

Интерфейс...

Синтаксис Automation:

BOOL ChoiceMaterial (BSTR material, double density);

Синтаксис COM:

BOOL ChoiceMaterial (char * material, double density);

Возвращаемое значение:

- Не используется.

Примечание:

1. Индекс метода задан в перечислении событий документа-модели....
2. Событие может приходить при редактировании параметров тела.

ChoiceProperty – Закончен выбор свойства

Интерфейс...

Синтаксис Automation:

BOOL ChoiceProperty(LPDISPATCH obj, double propID);

Синтаксис COM:

BOOL ChoiceProperty(LPUNKNOWN obj, double propID);

Возвращаемое значение:

- Не используется.

CreateEmbodiment – Добавлено новое исполнение

Интерфейс...

Синтаксис Automation:

BOOL CreateEmbodiment(LPCSTR marking);

Синтаксис COM:

BOOL DeleteEmbodiment(LPCSTR marking);

Возвращаемое значение:

- Не используется.

DeleteEmbodiment –Удалено исполнение

Интерфейс...

Синтаксис Automation:

BOOL DeleteEmbodiment(BSTR marking);

Синтаксис COM:

BOOL DeleteEmbodiment(LPCSTR marking);

Возвращаемое значение:

- Не используется.

Rebuild – Модель перестроена

Интерфейс...

Синтаксис Automation:

BOOL Rebuild();

Синтаксис COM:

BOOL Rebuild();

Возвращаемое значение:

- Не используется.

Примечание:

Индекс метода задан в перечислении событий документа-модели....

RollbackFeatures – Завершение отката дерева модели

Интерфейс...

Синтаксис Automation:

BOOL RollbackFeatures();

Синтаксис COM:

BOOL RollbackFeatures();

Возвращаемое значение:

Не используется

IDocument3DNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий документа-модели ksObject3DNotify/IObject3DNotify

ksObject3DNotify	- интерфейс Automation
IObject3DNotify	- интерфейс COM

Позволяют контролировать редактирование объектов документа-модели.

В Automation источник событий для подписки Object3DNotify можно получить при помощи метода ksPart::GetObject3DNotify.

В COM контейнером для подписки является указатель на компонент - IPart.

Дополнительный параметр при подписке задает объекты, для которых предназначен данный интерфейс событий. Таким параметром является либо тип объекта (o3d_unknown...o3d_sketch, o3d_axis2Planes...o3d_thread, o3d_part, o3d_feature), либо указатель на объект.

- Если задан тип объекта, события генерируются для всех объектов этого типа.
- Если задан указатель на объект, события генерируются только для этого объекта.
- Если задан тип объектов o3d_unknown, события генерируются для всех объектов.

В процессе обработки события можно получить информацию о редактировании объекта при помощи интерфейса IObject3DNotifyResult для COM или ksObject3DNotifyResult для Automation.

В Automation получить интерфейс результата можно при помощи метода ksPart::GetObject3DNotifyResult.

В COM получить интерфейс результата можно при помощи экспортной функции IPart::GetObject3DNotifyResult.

ksObject3DNotify – события

BeginDelete – Начало удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginDelete (LPDISPATCH obj);

Синтаксис COM:

BOOL BeginDelete (LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

TRUE	- удалить объект,
FALSE	- удаление объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

BeginLoadStateChange – Начало изменения типа загрузки

Интерфейс...

Синтаксис Automation:

BOOL BeginLoadStateChange(LPDISPATCH obj, long loadState);

Синтаксис COM :

BOOL BeginLoadStateChange(IUnknown* obj, long loadState);

Входные параметры:

obj	- объект-вставка модели, для которой изменяется вариант загрузки,
index	- индекс стандартного или пользовательского набора загрузки.

Возвращаемое значение:

TRUE	- запретить переключение,
FALSE	- разрешить переключение.

Примечание

У сборки есть умолчательные стандартные наборы, индексы которых соответствуют перечислению ksLoadStateEnum. Кроме этого могут быть еще и пользовательские наборы

BeginPlacementChanged – Начало изменения положения объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginPlacementChanged (LPDISPATCH obj);

Синтаксис COM:

BOOL BeginPlacementChanged (LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

TRUE	- изменить положение объекта,
FALSE	- изменение положения объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

BeginProcess – Начало редактирования\создания объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess (long pType, LPDISPATCH obj);

Синтаксис COM:

BOOL BeginProcess (long pType, LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection),
pType	- тип объекта (o3d_sketch, o3d_axis2Planes...o3d_lastEntityElement, o3d_part, o3d_feature).

Возвращаемое значение:

TRUE	- редактировать\создать объект,
FALSE	- редактирование\создание объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

BeginPropertyChanged – Начало изменения свойств объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginPropertyChanged (LPDISPATCH obj);

Синтаксис COM:

BOOL BeginPropertyChanged (LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

TRUE	- изменить свойства объекта,
FALSE	- изменение свойств объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

CreateObject – Объект создан

Интерфейс...

Синтаксис Automation:

BOOL CreateObject (LPDISPATCH obj);

Синтаксис COM:

BOOL CreateObject (LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

Delete – Завершение удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL Delete (LPDISPATCH obj);

Синтаксис COM:

BOOL Delete (LPUNKNOWN obj);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

EndProcess – Редактирование\создание объекта завершено

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (long pType);

Синтаксис COM:

BOOL EndProcess (long pType);

Входные параметры:

pType	- тип объекта (o3d_sketch, o3d_axis2Planes...o3d_lastEntityElement, o3d_part, o3d_feature).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

Excluded – Объект исключен/включен в расчет

Интерфейс...

Синтаксис Automation:

BOOL Excluded (LPDISPATCH obj, BOOL excluded);

Синтаксис COM:

BOOL Excluded (LPUNKNOWN obj, BOOL excluded);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection),
Excluded	TRUE - исключен из расчета, FALSE - включен в расчет.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

Hidden – Объект скрыт/показан

Интерфейс...

Синтаксис Automation:

BOOL Hidden (LPDISPATCH obj, BOOL _hidden);

Синтаксис COM:

BOOL Hidden (LPUNKNOWN obj, BOOL _hidden);

Входные параметры:

Obj	- указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection),
_hidden	TRUE - объект скрыт FALSE - объект показан.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

LoadStateChange – Завершение изменения типа загрузки

Интерфейс...

Синтаксис Automation:

BOOL LoadStateChange(LPDISPATCH obj, long loadState);

Синтаксис COM :

BOOL LoadStateChange(IUnknown* obj, long loadState);

Входные параметры:

index	- индекс стандартного или пользовательского набора загрузки.

Возвращаемое значение:

- не используется.

Примечание

У сборки есть умолчательные стандартные наборы, индексы которых соответствуют перечислению ksLoadStateEnum. Кроме этого могут быть еще и пользовательские наборы

PlacementChanged – Изменено положения объекта

Интерфейс...

Синтаксис Automation:

BOOL PlacementChanged (LPDISPATCH obj);

Синтаксис COM:

BOOL PlacementChanged (LPUNKNOWN obj);

Входные параметры:

Obj - указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

PropertyChanged – Изменены свойства объета

Интерфейс...

Синтаксис Automation:

BOOL PropertyChanged (LPDISPATCH obj);

Синтаксис COM:

BOOL PropertyChanged (LPUNKNOWN obj);

Входные параметры:

Obj - указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей...

UpdateObject – Объект изменен

Интерфейс...

Синтаксис Automation:

BOOL UpdateObject (LPDISPATCH obj);

Синтаксис COM:

BOOL UpdateObject (LPUNKNOWN obj);

Входные параметры:

Obj - указатель на интерфейс ksFeature (IFeature) либо ksFeatureCollection (IFeatureCollection).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объектов документов-моделей....

IObject3DNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Интерфейс...

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType - номер события в перечислении событий.

Возвращаемое значение:

TRUE - событие будет обрабатываться,
FALSE - событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий основной надписи графического документа ksStampNotify/IStampNotify

ksStampNotify - интерфейс Automation
IStampNotify - интерфейс COM

Позволяют контролировать состояние основной надписи.

В Automation источником событий для подписки является указатель на интерфейс ksStamp.

В COM контейнером для подписки является указатель на документ.

IStampNotify – события

BeginEditStamp – Начало работы со штампом

Интерфейс...

Синтаксис Automation:

BOOL BeginEditStamp();

Возвращаемое значение:

TRUE	- редактировать штамп,
FALSE	- штамп редактировать нельзя.

Синтаксис COM:

BOOL BeginEditStamp();

Возвращаемое значение:

TRUE	- редактировать штамп,
FALSE	- штамп редактировать нельзя.

Примечание:

Индекс события задан в перечислении событий штампа....

EndEditStamp – Завершение работы со штампом

Интерфейс...

Синтаксис Automation:

BOOL EndEditStamp (BOOL editResult);

Входные параметры:

editResult	- результат редактирования штампа.

Возвращаемое значение:

- не используется.

Синтаксис COM:

BOOL EndEditStamp (BOOL editResult);

Входные параметры:

editResult	- результат редактирования штампа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс события задан в перечислении событий штампа....

StampBeginClearCells – Начало очистки ячейки штампа

Интерфейс...

Синтаксис Automation:

BOOL StampBeginClearCells(VARIANT numbers);

Входные параметры:

numbers	- список очищаемых ячеек.

Возвращаемое значение:

TRUE	- редактировать ячейку,
FALSE	- редактирование ячейки запрещено.

Синтаксис COM:

BOOL StampBeginClearCells(VARIANT numbers);

Входные параметры:

Number	- список очищаемых ячеек.

Возвращаемое значение:

TRUE	- редактировать ячейку,
FALSE	- редактирование ячейки запрещено.

Примечание:

- Если очищается одна ячейка, то тип параметра numbers VT_I4.
- Если очищаются несколько ячеек, то передается массив SafeArray тип VT_ARRAY | VT_I4.

StampCellBeginEdit – Начало редактирования в ячейке штампа

Интерфейс...

Синтаксис Automation:

BOOL StampCellBeginEdit (long number);

Входные параметры:

Number	- номер редактируемой ячейки.

Возвращаемое значение:

TRUE - редактировать ячейку,
FALSE - редактирование ячейки запрещено.

Синтаксис COM:

BOOL StampCellBeginEdit (long number);

Входные параметры:

Number - номер редактируемой ячейки.

Возвращаемое значение:

TRUE - редактировать ячейку,
FALSE - редактирование ячейки запрещено.

Примечание:

Индекс события задан в перечислении событий штампа....

StampCellDblClick – Двойной щелчок мышью в ячейке штампа

Интерфейс...

Синтаксис Automation:

BOOL StampCellDblClick (long number);

Входные параметры:

Number - номер ячейки.

Возвращаемое значение:

TRUE - обработать двойной щелчок в ячейке,
FALSE - обработка щелчка запрещена.

Синтаксис COM:

BOOL StampCellDblClick (long number);

Входные параметры:

Number - номер ячейки.

Возвращаемое значение:

TRUE - обработать двойной щелчок в ячейке,
FALSE - обработка щелчка запрещена.

Примечание:

Индекс события задан в перечислении событий штампа....

IStampNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий менеджера выделенных объектов ksSelectionMngNotify/ISelectionMngNotify

ksSelectionMngNotify	- интерфейс Automation
ISelectionMngNotify	- интерфейс COM

Позволяют контролировать выделение объектов документа.

В Automation источник событий для графического документа SelectionMngNotify можно получить при помощи метода ksDocument2D::GetSelectionMngNotify.

Для документа-модели источником является Менеджер выделенных объектов ksSelectionMng.

В COM контейнером для подписки на данный интерфейс является указатель на графический документ или документ-модель.

ISelectionMngNotify – события

Select – Объект выделен

Интерфейс...

Синтаксис Automation:

BOOL Select (VARIANT obj);

Синтаксис COM:

BOOL Select (VARIANT obj);

Входные параметры:

obj - выделенный объект.

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс метода задан в перечислении событий менеджера выделенных объектов....
2. Выделенный объект для графического документа: obj.vt = VT_I4, obj.lVal - reference объекта
3. Выделенный объект для документа-модели: obj.vt = VT_UNKNOWN, obj.punkVal - указатель на интерфейс объекта (объекты дерева - объекты, компоненты, сопряжения)

Unselect – Выделение с объекта снято

Интерфейс...

Синтаксис Automation:

BOOL Unselect (VARIANT obj);

Синтаксис COM:

BOOL Unselect (VARIANT obj);

Входные параметры:

obj - объект, с которого снято выделение.

Возвращаемое значение:

- не используется.

Примечание:

1. Индекс метода задан в перечислении событий менеджера выделенных объектов....
2. Объект, с которого снято выделение для графического документа: obj.vt = VT_I4, obj.lVal - reference объекта

3. Объект, с которого снято выделение для документа-модели: obj.vt = VT_UNKNOWN, obj.punkVal - указатель на интерфейс объекта (объекты дерева - объекты, компоненты, сопряжения)

UnselectAll – Выделение снято со всех объектов

Интерфейс...

Синтаксис Automation:

BOOL UnselectAll();

Синтаксис COM:

BOOL UnselectAll();

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий менеджера выделенных объектов....

ISelectionMngNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType - номер события в перечислении событий.

Возвращаемое значение:

TRUE - событие будет обрабатываться,
FALSE - событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий документа-спецификации ksSpcDocumentNotify/ISpcDocumentNotify

ksSpcDocumentNotify - интерфейс Automation
ISpcDocumentNotify - интерфейс COM

Позволяют контролировать редактирование документа-спецификации.

В Automation источник событий для документа-спецификации SpcDocumentNotify можно получить при помощи метода ksSpcDocument::GetSpcDocumentNotify.

В COM источником для подписки является документ-спецификация.

ISpcDocumentNotify – события

DocumentAdd – Добавлен документ сборочного чертежа

Интерфейс...

Синтаксис Automation:

BOOL DocumentAdd (BSTR docName);

Синтаксис COM:

BOOL DocumentAdd (char * docName);

Входные параметры:

docName - имя файла документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

DocumentBeginAdd – Начало добавления документа сборочного чертежа

Интерфейс...

Синтаксис Automation:

BOOL DocumentBeginAdd();

Синтаксис COM:

BOOL DocumentBeginAdd();

Возвращаемое значение:

TRUE - добавить документ сборочного чертежа,
FALSE - добавление документа запрещено.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

DocumentBeginRemove – Начало удаления документа сборочного чертежа

Интерфейс...

Синтаксис Automation:

BOOL DocumentBeginRemove (BSTR docName);

Синтаксис COM:

BOOL DocumentBeginRemove (char * docName);

Входные параметры:

docName	- имя файла документа.

Возвращаемое значение:

TRUE	- удалить документ сборочного чертежа,
FALSE	- удаление документа запрещено.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

DocumentRemove – Документ удален

Интерфейс...

Синтаксис Automation:

BOOL DocumentRemove (BSTR docName);

Синтаксис COM:

BOOL DocumentRemove (char * docName);

Входные параметры:

docName	- имя файла документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

SpcStyleBeginChange – Начало изменения стиля спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcStyleBeginChange (BSTR libName, long numb);

Синтаксис COM:

BOOL SpcStyleBeginChange (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

TRUE	- изменить стиль спецификации,
FALSE	- изменение стиля запрещено.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

SpcStyleChange – Стиль спецификации изменен

Интерфейс...

Синтаксис Automation:

BOOL SpcStyleChange(BSTR libName, long numb);

Синтаксис COM:

BOOL SpcStyleChange(char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий документа-спецификации....

ISpcDocumentNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий спецификации ksSpecificationNotify/ISpecificationNotify

ksSpecificationNotify	- интерфейс Automation
ISpecificationNotify	- интерфейс COM

Позволяют контролировать редактирование описания спецификации.

В Automation источником событий для подписки на данный интерфейс является объект ksSpecification.

В COM источником для подписки является является документ (графический документ, документ-модель, спецификация).

ISpecificationNotify – события

BeginCalcPositions – Начало расчета позиций

Интерфейс...

Синтаксис Automation:

BOOL BeginCalcPositions();

Синтаксис COM:

BOOL BeginCalcPositions();

Возвращаемое значение:

TRUE	- расчет позиций,
FALSE	- расчет позиций запрещен.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

BeginCreateObject – Начало расчета позиций

Интерфейс...

Синтаксис Automation:

BOOL BeginCreateObject (long typeObj);

Синтаксис COM:

BOOL BeginCreateObject (long typeObj);

Входные параметры:

0	- базовый объект,
1	- объект вспомогательный,
-1	- тип объекта не определен.

Возвращаемое значение:

TRUE	- создать объект спецификации,
FALSE	- создание объекта спецификации запрещены.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

CalcPositions – Расчет позиций завершен

Интерфейс...

Синтаксис Automation:

BOOL CalcPositions();

Синтаксис COM:

BOOL CalcPositions();

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

ChangeCurrentSpcDescription – Изменилось текущее описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL ChangeCurrentSpcDescription (BSTR libName, long numb);

Синтаксис COM:

BOOL ChangeCurrentSpcDescription (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

SpcDescriptionAdd – Добавилось описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionAdd (BSTR libName, long numb);

Синтаксис COM:

BOOL SpcDescriptionAdd (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

SpcDescriptionBeginEdit – Начало редактирования описания спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionBeginEdit (BSTR libName, long numb);

Синтаксис COM:

BOOL SpcDescriptionBeginEdit (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

TRUE	- редактировать описание спецификации,
FALSE	- редактирование описания запрещено.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

SpcDescriptionEdit – Редактирование описания спецификации завершено

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionEdit (BSTR libName, long numb, BOOL isOk);

Синтаксис COM:

BOOL SpcDescriptionEdit (char * libName, long numb, BOOL isOk);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации,
isOk	- результат изменения описания спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

SpcDescriptionRemove – Удалилось описание спецификации

Интерфейс...

Синтаксис Automation:

BOOL SpcDescriptionRemove (BSTR libName, long numb);

Синтаксис COM:

BOOL SpcDescriptionRemove (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

Synchronization – Синхронизация проведена

Интерфейс...

Синтаксис Automation:

BOOL Synchronization();

Синтаксис COM:

BOOL Synchronization();

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

SynchronizationBegin – Начало синхронизации

Интерфейс...

Синтаксис Automation:

BOOL SynchronizationBegin();

Синтаксис COM:

BOOL SynchronizationBegin();

Возвращаемое значение:

TRUE	- синхронизировать,
FALSE	- синхронизация запрещена.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

TuningSpcStyleBeginChange – Начало изменения настроек спецификации

Интерфейс...

Синтаксис Automation:

BOOL TuningSpcStyleBeginChange (BSTR libName, long numb);

Синтаксис COM:

BOOL TuningSpcStyleBeginChange (char * libName, long numb);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации.

Возвращаемое значение:

TRUE	- изменить настройки спецификации,
FALSE	- изменение настроек спецификации запрещено.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

TuningSpcStyleChange – Настройки спецификации изменились

Интерфейс...

Синтаксис Automation:

BOOL TuningSpcStyleChange (BSTR libName, long numb, BOOL isOk);

Синтаксис COM:

BOOL TuningSpcStyleChange (char * libName, long numb, BOOL isOk);

Входные параметры:

libName	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации,
isOk	- результат изменения настроек спецификации.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода установлен в перечислении событий для спецификации....

ISpecificationNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс событий объекта спецификации ksSpcObjectNotify/ISpcObjectNotify

ksSpcObjectNotify	- интерфейс Automation
ISpcObjectNotify	- интерфейс COM

Позволяют контролировать редактирование объектов спецификации.

В Automation источник событий для подписки SpcObjectNotify можно получить при помощи метода ksSpecification::GetSpcObjectNotify.

В COM источником для подписки является документ (графический, документ-модель, спецификация).

ksSpcObjectNotify – события

BeginDelete – Начало удаления объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginDelete (long objRef);

Синтаксис COM:

BOOL BeginDelete (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

TRUE	- удалить объект,
FALSE	- удаление объекта запрещено.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

BeginGeomChange – Начало изменения геометрии объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL BeginGeomChange (long objRef);

Синтаксис COM:

BOOL BeginGeomChange (long objRef);

Входные параметры:

objRef	- указатель на объект,

Возвращаемое значение:

TRUE	- изменить геометрию объекта спецификации,
FALSE	- изменение объекта спецификации запрещено.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

BeginProcess – Начало редактирования/создания объекта

Интерфейс...

Синтаксис Automation:

BOOL BeginProcess (long pType, long objRef);

Синтаксис COM:

BOOL BeginProcess (long pType, long objRef);

Входные параметры:

objRef	- указатель на объект,
pType	- тип объекта спецификации... (SPC_BASE_OBJECT, SPC_COMMENT).

Возвращаемое значение:

TRUE	- редактировать/создать объект,
FALSE	- редактирование/создание объекта спецификации запрещено.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

CellBeginEdit – Начало редактирования в ячейке

Интерфейс...

Синтаксис Automation:

BOOL CellBeginEdit (long objRef, long number);

Синтаксис COM:

BOOL CellBeginEdit (long objRef, long number);

Входные параметры:

objRef	- указатель на объект,
number	- номер ячейки.

Возвращаемое значение:

TRUE	- редактировать ячейку,
FALSE	- редактирование в ячейке запрещено.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

CellDblClick – Двойной щелчок в ячейке

Интерфейс...

Синтаксис Automation:

BOOL StampCellDblClick (long objRef, long number);

Синтаксис COM:

BOOL StampCellDblClick (long objRef, long number);

Входные параметры:

objRef	- указатель на объект,
number	- номер ячейки.

Возвращаемое значение:

TRUE	- обработать двойной щелчок,
FALSE	- обработка двойного щелчка запрещена.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

ChangeCurrent – Текущий объект изменен

Интерфейс...

Синтаксис Automation:

BOOL ChangeCurrent (long objRef);

Синтаксис COM:

BOOL ChangeCurrent (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

CreateObject – Создание объекта

Интерфейс...

Синтаксис Automation:

BOOL CreateObject (long objRef);

Синтаксис COM:

BOOL CreateObject (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации....

Delete – Объект удален

Интерфейс...

Синтаксис Automation:

BOOL Delete (long objRef);

Синтаксис COM:

BOOL Delete (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

DocumentAdd – Добавление документа в объект спецификации

Интерфейс...

Синтаксис Automation:

BOOL DocumentAdd (long objRef, BSTR docName);

Синтаксис COM:

BOOL DocumentAdd (long objRef, char * docName);

Входные параметры:

objRef	- указатель на объект,
docName	- имя файла документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

DocumentBeginAdd – Начало добавления документа

Интерфейс...

Синтаксис Automation:

BOOL DocumentBeginAdd (long objRef);

Синтаксис COM:

BOOL DocumentBeginAdd (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

TRUE	- добавить документ сборочного чертежа,
FALSE	- добавление документа сборочного чертежа запрещено.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

DocumentRemove – Удаление документа из объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL DocumentRemove (long objRef, BSTR docName);

Синтаксис COM:

BOOL DocumentRemove (long objRef, char * docName);

Входные параметры:

objRef	- указатель на объект,
docName	- имя файла документа.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

EndProcess – Конец редактирования/создания объекта

Интерфейс...

Синтаксис Automation:

BOOL EndProcess (long pType);

Синтаксис COM:

BOOL EndProcess (long pType);

Входные параметры:

pType - тип процесса (SPC_BASE_OBJECT, SPC_COMMENT).

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

GeomChange – Изменение геометрии объекта спецификации

Интерфейс...

Синтаксис Automation:

BOOL GeomChange (long objRef, BSTR docName);

Синтаксис COM:

BOOL GeomChange (long objRef, char * docName);

Входные параметры:

objRef - указатель на объект.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации...

UpdateObject – Редактирование объекта

Интерфейс...

Синтаксис Automation:

BOOL UpdateObject (long objRef);

Синтаксис COM:

BOOL UpdateObject (long objRef);

Входные параметры:

objRef	- указатель на объект.

Возвращаемое значение:

- не используется.

Примечание:

Индекс метода задан в перечислении событий объекта спецификации....

ksSpcObjectNotify – методы

IsNotiifyProcess – Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

Интерфейс дополнительных параметров для событий документа-модели ksDocument3DNotifyResult/ IDocument3DNotifyResult

Примечание:

1. Интерфейс позволяет получить указатель на объект для которого посылается данное событие. Например, выбор материала и обозначения доступны для верхнего компонента, вставки и тела.
2. Интерфейс можно получить с помощью метода IDocument3D::GetDocument3DNotifyResult или ksDocument3D::GetDocument3DNotifyResult.

ksDocument3DNotifyResult – методы

GetNotifyObject – Получить объект для которого посылается событие

Интерфейс...

Синтаксис Automation:

LPDISPATCH GetNotifyObject();

Синтаксис COM:

IUnknown* GetNotifyObject();

Возвращаемое значение:

В зависимости от типа объекта, возвращаемого методом GetNotifyObjectType возвращается:

тип объекта	Интерфейс,
o3d_part	- IPart или ksPart,
o3d_body	- IBody или ksBody,
o3d_uknown	- NULL.

GetNotifyObjectType – Получить тип объекта

Интерфейс...

Синтаксис Automation:

long GetNotifyObjectType();

Синтаксис COM:

long GetNotifyObjectType();

Возвращаемое значение:

- Тип объекта из перечисления ksObj3dTypeEnum. Допустимыми значениями являются o3d_part, o3d_body или o3d_uknown.

GetNotifyType – Получить тип события

Интерфейс...

Синтаксис Automation:

long GetNotifyType();

Синтаксис COM:

long GetNotifyType();

Возвращаемое значение:

- Идентификатор события из перечисления ksDocument3DNotifyEnum.

GetRequestFilesType – Тип процесса, запрашивающего файл

Интерфейс...

Синтаксис Automation:

long GetRequestFilesType();

Синтаксис COM:

long GetRequestFilesType();

Возвращаемое значение:

- Тип процесса из перечисления ksRequestFilesTypeEnum.

API экспортных функций

Функции работы с документом моделью

Функции данного раздела позволяют работать с документом-моделью.

ksGetActive3dDocument – Получить указатель на активный документ-модель

Аналог данной функции при использовании Automation - метод KompasObject::ActiveDocument3D.

Синтаксис:

LPDOCUMENT3D ksGetActive3dDocument();

Возвращаемое значение:

указатель на активный документ трехмерной модели	- в случае удачного завершения,
NULL	- если активного документа нет.

ksGet3dDocument – Получить указатель на объект документа-модели

Синтаксис:

LPDOCUMENT3D ksGet3dDocument();

Возвращаемое значение:

указатель на объект документа трехмерной модели	- в случае удачного завершения,
NULL	- если такого документа нет.

ksGet3dDocumentFromReference – Получить указатель на документ трехмерной модели, соответствующий присланному указателю

Аналог данной функции при использовании Automation - метод KompasObject::ksGet3dDocumentFromRef.

Синтаксис:

LPDOCUMENT3D ksGet3dDocumentFromReference (reference doc);

Входной параметр:

doc	- указатель на документ.

Возвращаемое значение:

указатель на документ трехмерной модели	- в случае удачного завершения,
NULL	- если такого документа нет.

ksGetModelLibrary – Получить указатель на интерфейс библиотеки моделей

Аналог данной функции при использовании Automation - метод KompasObject::GetModelLibrary.

Синтаксис:

LPMODELLIBRARY ksGetModelLibrary();

Возвращаемое значение:

- указатель на интерфейс ksModelLibrary.

ksGetObjectsFilter3D – Получить интерфейс фильтрации объектов 3D

Справка системы КОМПАС...

KOMPAS.chm::/683_82_7_2_Filqtry_obwektov.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetObjectsFilter3D.

Синтаксис:

IObjectsFilter3D * ksGetObjectsFilter3D();

Возвращаемое значение:

указатель на интерфейс ksObjectsFilter3D

ksGetReferenceFrom3dDocument– Получить указатель на документ трехмерной модели, соответствующий присланному интерфейсу

Синтаксис:

reference ksGetReferenceFrom3dDocument (LPDOCUMENT3D doc);

Входной параметр:

doc	- интерфейс документа трехмерной модели.

Возвращаемое значение:

указатель на документ трехмерной модели	- в случае удачного завершения,
NULL	- если такого документа нет.

Функции работы с графическим документом

Функции управления масштабом

Функции данного раздела позволяют управлять масштабом отображения объектов документа

ksGetZoomScale – Получить масштаб и центр изменения масштаба окна графического документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/62_6_1_Izmenenie_masshtaba_izob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetZoomScale.

Синтаксис:

int ksGetZoomScale (double *x, double *y, double *scale);

Выходные параметры:

x, y	- координаты (в СК вида) точки центра изменения масштаба,
scale	- коэффициент изменения масштаба изображения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksRefreshActiveWindow – Обновить окно документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/67_6_4_Obnovlenie_izobrazhenija.htm

Синтаксис:

int ksRefreshActiveWindow ();

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksZoom – Задать масштаб изображения прямоугольной рамкой

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/62_6_1_Izmenenie_masshtaba_izob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksZoom.

Синтаксис:

int ksZoom (double x1, double y1, double x2, double y2);

Входные параметры:

x1, y1	- координаты (в СК вида) первой точки диагонали рамки,
x2, y2	- координаты (в СК вида) второй точки диагонали рамки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksZoomPrevNextOrAll – Показать документ в предыдущем/последующем масштабе или документ целиком

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/62_6_1_Izmenenie_masshtaba_izob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksZoomPrevNextOrAll.

Синтаксис:

int ksZoomPrevNextOrAll (unsigned char type);

Входной параметр:

type	тип масштабирования: 0 - следующий масштаб, 1 - предыдущий масштаб, 2 - весь документ.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksZoomScale – Задать масштаб изображения по точке и коэффициенту масштабирования

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/62_6_1_Izmenenie_masshtaba_izob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksZoomScale.

Синтаксис:

int ksZoomScale (double x, double y, double scale);

Входные параметры:

x1, y1 - координаты (в СК вида) точки центра изменения масштаба,
scale_ - коэффициент изменения масштаба изображения.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Вспомогательные построения

Математические функции

Функции данного раздела позволяют выполнять математические вычисления.

AtanD – Получить арктангенс аргумента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksAtanD.

Синтаксис:

double AtanD (double tang);

Входной параметр:

tang - тангенс угла.

Возвращаемое значение:

- значение угла (в градусах).

CosD – Получить косинус аргумента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCosD.

Синтаксис:

double CosD (double ang);

Входной параметр:

ang - угол в градусах.

Возвращаемое значение:

- косинус аргумента.

ksDistanceT1T2OnCurve – Получить расстояние между двумя точками на кривой по параметрам кривой в данных точках

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistanceT1T2OnCurve.

Синтаксис:

double ksDistanceT1T2OnCurve (reference curve,double t1,double t2);

Входные параметры:

curve - указатель на кривую,
t1 - параметр кривой в точке 1,
t2 - параметр кривой в точке 2.

Возвращаемое значение:

- Расстояние между двумя точками на кривой.

ksEqualPoints – Определить эквивалентность (совпадение) двух точек

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksEqualPoints.

Синтаксис:

int ksEqualPoints(double x1, double y1, double x2, double y2);

Входные параметры:

x1, y1	- координаты первой точки,
x2, y2	- координаты второй точки.

Возвращаемое значение:

1	- точки совпадают,
0	- точки не совпадают.

ksGetCurveMinMaxParametr – Получить минимальный и максимальный параметр кривой

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurveMinMaxParametr.

Синтаксис:

int ksGetCurveMinMaxParametr (reference curve,double *tMin,double *tMax);

Входные параметры:

curve	- указатель на кривую,

Выходные параметры:

tMin	- минимальный параметр кривой,
tMax	- максимальный параметр кривой.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksGetCurvePerpendicular – Получить угол нормали к кривой в заданной точке

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurvePerpendicular.

Синтаксис:

double ksGetCurvePerpendicular (reference curve, double x, double y);

Входные параметры:

curve - указатель на кривую,
x, y - координаты точки.

Возвращаемое значение:

угол нормали к кривой в заданной точке (в градусах). - в случае успеха,
-1 - в случае неудачи.

Примечание:

Если указанная точка не находится на кривой, она проецируется на кривую.

ksGetCurvePoint – Преобразовать параметр кривой t в координаты вида

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurvePoint.

Синтаксис:

int ksGetCurvePoint (reference curve,
double t,
double *kx,
double *ky);

Входные параметры:

curve - указатель на кривую,
t - параметр кривой.

Выходные параметры:

kx, ky - координаты проекции точки на кривую,

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

См. также ksGetCurvePointProjectionEx

ksGetCurvePointProjection – Получить координаты проекции точки на кривую

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurvePointProjection.

Синтаксис:

int ksGetCurvePointProjection (reference curve,

double x, double y,

double *kx, double *ky);

Входные параметры:

curve	- указатель на кривую,
x, y	- координаты проецируемой точки.

Выходные параметры:

kx, ky	- координаты проекции точки на кривую.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. См. также ksGetCurvePointProjectionEx
2. При выполнении функции ksGetCurvePointProjection проекция на кривую из данной точки может быть не найдена, например, если проекция находится на продолжении кривой. В таком случае возвращаются координаты ближайшей к проекции начальной или конечной точки кривой.
3. В функцию ksGetCurvePointProjection нужно передавать координаты в системе координат кривой, то есть в системе координат макроэлемента, которому принадлежит кривая. Возвращаются координаты также в системе координат кривой.

ksGetCurvePointProjectionEx – Рассчитать координаты проекции точки на кривую

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurvePointProjectionEx.

Синтаксис:

int ksGetCurvePointProjectionEx (reference curve,
double x,
double y,
double *kx,
double *ky,
double *t);

Входные параметры:

curve	- указатель на кривую,
x, y	- координаты проецируемой точки.

Выходные параметры:

kx, ky	- координаты проекции точки на кривую,
t	- параметр кривой.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. См. также ksGetCurvePointProjection
2. При выполнении функции ksGetCurvePointProjectionEx проекция на кривую из данной точки может быть не найдена, например, если проекция находится на продолжении кривой. В таком случае возвращаются координаты ближайшей к проекции начальной или конечной точки кривой.
3. В функцию ksGetCurvePointProjectionEx нужно передавать координаты в системе координат кривой, то есть в системе координат макроэлемента, которому принадлежит кривая. Возвращаются координаты также в системе координат кривой.

ksIsCurveClosed – Проверить, замкнута ли указанная кривая

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsCurveClosed.

Синтаксис:

int ksIsCurveClosed (reference p);

Входной параметр:

p	- указатель на кривую.

Возвращаемое значение:

1	- кривая замкнута,
0	- кривая разомкнута,
-1	- в случае неудачи.

ksIsPointInsideContour – Проверить положение точки относительно кривой

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsPointInsideContour.

Синтаксис:

int ksIsPointInsideContour (reference p, double x, double y, double precision);

Входные параметры:

p	- указатель на контур,
x, y	- координаты точки,
precision	- точность проверки (от 1 до 1E-6).

Возвращаемое значение:

0	- в случае неудачи,
1	- точка вне контура,
2	- точка на контуре,
3	- точка внутри контура.

ksLengthFromMtr – Пересчитать длину из локальной СК в СК вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLengthFromMtr.

Синтаксис:

int ksLengthFromMtr (double *len);

Входной и Выходной параметр:

len	- длина.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Параметр len является одновременно и входным, и выходным. На входе он представляет собой длину в локальной СК, а в результате работы функции преобразуется в длину в СК вида.

ksLengthIntoMtr – Пересчитать длину из СК вида в локальную СК

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLengthIntoMtr.

Синтаксис:

int ksLengthIntoMtr (double *len);

Входной и Выходной параметр:

len - длина.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр len является одновременно и входным, и выходным. На входе он представляет собой длину в СК вида, а в результате работы функции преобразуется в длину в локальной СК.

ksLinePointTangentCurve – Функция расчета касательных к кривой, проходящих через заданную точку

Синтаксис:

```
void LIB_FUNC ksLinePointTangentCurve( reference p,      // указатель на кривую
                                       double xc,        // координаты точки
                                       double yc,
                                       reference angArr ) // массив углов касательных
```

Входные параметры:

p - указатель на кривую,
xc, yc - координаты точки.

Выходные параметры:

angArr - массив углов касательных к кривой, проходящих через заданную точку.

ksMakeEncloseContours – Определить группу объектов, охватывающих заданную точку

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMakeEncloseContours.

Синтаксис:

reference ksMakeEncloseContours (reference gr, double x, double y);

Входные параметры:

gr - временная группа
или 0, если требуются контуры в текущем виде,
x, y - координаты точки внутри охватываемых контуров.

Возвращаемое значение:

указатель на временную группу контуров - в случае удачного завершения,
0 - если контуров не нашлось
или в случае неудачи.

ksPointFromMtr – Пересчитать координаты точки из локальной СК в СК вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPointFromMtr.

Синтаксис:

int ksPointFromMtr (double x, double y, double *xn, double *yn);

Входные параметры:

x, y - координаты точки в локальной системе координат,

Выходные параметры:

xn, yn - координаты точки в СК вида.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksPointIntoMtr – Пересчитать координаты точки из СК вида в локальную СК

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPointIntoMtr.

Синтаксис:

int ksPointIntoMtr (double x, double y, double *xn, double *yn);

Входные параметры:

x, y - координаты точки в системе координат вида.

Выходные параметры:

xn, yn - координаты точки в локальной СК.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksPointsOnCurve – Получить массив равномерно расположенных по кривой точек

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1391_Postroenie_tochek.htm#point_on_curve

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksPointsOnCurve.

Синтаксис:

reference ksPointsOnCurve (reference curve, int count);

Входные параметры:

curve - указатель на кривую,
count - количество точек.

Возвращаемое значение:

- указатель на массив равномерно расположенных на указанной кривой точек. - в случае успеха,
0 - в случае неудачи.

ksTanCurvCurv – Получить координаты точек касания прямых к двум кривым

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1396_vspomogatelnye.htm#TANGENT2_LINE

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksTanCurvCurv.

Синтаксис:

int ksTanCurvCurv(reference p1, reference p2,reference pointArr1,reference pointArr2);

Входные параметры:

p1 - указатель на первую кривую,
p2 - указатель на вторую кривую.

Выходные параметры:

pointArr1 - динамический массив математических точек POINT_ARR на кривой p1,
pointArr2 - динамический массив математических точек POINT_ARR на кривой p2.

Возвращаемое значение:

1 - успешное завершение,
0 - построить касательную нельзя (кривые совпадают или одна кривая вложена в другую),
-1 - первый объект не существует,
-2 - второй объект не существует,
-3 - кривые расположены в разных видах,
-4 - не совпадают СК определения кривых (геометрическая и аннотационная),
-5 - первый объект не является кривой,
-6 - второй объект не является кривой,
-7 - ошибка.

MovePoint – Сдвинуть точку по вектору

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMovePoint.

Синтаксис:

void MovePoint (double *x, double *y, double ang, double len);

Входные параметры:

x, y - координаты точки,
ang - угол вектора сдвига в градусах,
len - длина вектора сдвига.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

Параметры x и y являются одновременно и входными, и выходными. На входе они представляют собой начальные координаты точки, а в результате работы метода они преобразуются в координаты точки после сдвига.

Rotate – Повернуть точку относительно центра

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksRotate.

Синтаксис:

void Rotate (double x, double y, double xc, double yc, double ang, double *xr, double *yr);

Входные параметры:

x, y	- координаты базовой точки,
xc, yc	- координаты центра поворота,
ang	- угол поворота.

Выходные параметры:

xr, yr	- координаты точки после поворота.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

SheetToView – Пересчитать координаты точки из СК листа в СК текущего вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSheetToView.

Синтаксис:

int (double x, double y, double * outX, double * outY);

Входные параметры:

x, y	- координаты точки в системе координат листа.

Выходные параметры:

outX, outY	- координаты точки в системе координат вида.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

SinD – Получить синус аргумента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksSinD.

Синтаксис:

double SinD (double ang);

Входной параметр:

ang - угол в градусах.

Возвращаемое значение:

синус аргумента.

Symmetry – Получить координаты точки, симметричной относительно заданной оси

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksSymmetry.

Синтаксис:

void Symmetry (double x, double y,double x1, double y1, double x2, double y2, double *xs, double *ys);

Входные параметры:

x, y - координаты базовой точки,
x1, y1 - координаты первой точки на оси симметрии,
x2, y2 - координаты второй точки на оси симметрии.

Выходные параметры:

xs, ys - координаты симметричной точки.

Возвращаемое значение:

TRUE - в случае удачного завершения,

FALSE - в случае неудачи.

TanD – Получить тангенс аргумента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/74_8_1_3_Vvod_znachenij_v_polja.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::TanD.

Синтаксис:

double TanD (double ang);

Входной параметр:

ang - угол в градусах.

Возвращаемое значение:

тангенс аргумента.

ViewToSheet – Пересчитать координаты точки из СК текущего вида в СК листа

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksViewToSheet.

Синтаксис:

int ViewToSheet (double x, double y, double * outX, double * outY);

Входные параметры:

x, y - координаты точки в системе координат вида.

Выходные параметры:

outX, outY - координаты точки в системе координат листа.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Функции вычисления пересечений

Функции данного раздела обеспечивают вычисление пересечений геометрических объектов друг с другом.

IntersectArcArc – Получить координаты точек пересечения двух дуг окружностей

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectArcArc.

Синтаксис:

void IntersectArcArc (double xac, double yac,

double rada,

double fa1, double fa2,

short directa,

double xbc, double ybc,

double radb,

double fb1, double fb2,

short directb,

int * kp,

double xp[], double yp[]);

Входные параметры:

xac, yac	- координаты центра первой дуги,
rada	- радиус первой дуги,
fa1, fa2	- углы начальной и конечной точек первой дуги,
directa	- направление первой дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке,
xbc, ybc	- координаты центра второй дуги,
radb	- радиус второй дуги,
fb1, fb2	- углы начальной и конечной точек второй дуги,
directb	- направление второй дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке.

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp[], yp[]	- массивы координат точки пересечения.

IntersectArcLine – Получить координаты точек пересечения дуги окружности и прямой

Пример...

Синтаксис:

void IntersectArcLine (double xc,
double yc,
double rad,
double f1,
double f2,
int n,
double x,
double y,
double ang,
int *kp,
double xp[],
double yp[]);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус дуги,
f1, f2	- углы начальной и конечной точек дуги,
n	- направление дуги,
x, y	- координаты точки на прямой,
ang	- угол наклона прямой.

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp, yp	- координаты точки пересечения.

IntersectCirArc – Получить координаты точек пересечения окружности и дуги

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectCirArc.

Синтаксис:

void IntersectCirArc (double xcc, double ycc,

double radc,

double xac, double yac,

double rada,

double fa1, double fa2,

short directa,

int * kp,

double xp, double yp);

Входные параметры:

xcc, ycc	- координаты центра окружности,

radc	- радиус окружности,
xac, yac	- координаты центра дуги,
rada	- радиус дуги,
fa1, fa2	- углы начальной и конечной точек дуги,
directa	- направление дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке.

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp, yp	- координаты точки пересечения.

IntersectCirCir – Получить координаты точек пересечения двух окружностей

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectCirCir.

Синтаксис:

void IntersectCirCir (double xc1,double yc1,
double rad1, double xc2,
double yc2, double rad2,
int *kp, double xp, double yp);

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
rad1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
rad2	- радиус второй окружности,

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp, yp	- координаты точки пересечения.

IntersectCirLin – Получить координаты точек пересечения окружности и прямой

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectCirLin.

Синтаксис:

void IntersectCirLin (double xc,
double yc,
double rad,
double x,
double y,
double ang,
int *kp,
double xp,
double yp);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
x, y	- координаты точки на прямой,
ang	- угол наклона прямой.

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp, yp	- координаты точки пересечения.

IntersectCurvCurv – Получить координаты точек пересечения двух кривых

Пример...

Синтаксис:

void IntersectCurvCurv (reference p1, reference p2, int *kp, double xp, double yp, int MaxCount);

Входные параметры:

p1	- указатель на первую кривую,
p2	- указатель на вторую кривую,
MaxCount	- максимальное число выдаваемых точек пересечений.

Выходные параметры:

kp	- число пересечений,
xp, yp	- координаты точки пересечения.

IntersectCurvCurvEx – Получить координаты точек пересечения двух кривых

Синтаксис:

void IntersectCurvCurv (reference p1, reference p2, int *kp, double xp, double yp, int MaxCount, int touchInclude);

Входные параметры:

p1 - указатель на первую кривую,
p2 - указатель на вторую кривую,
MaxCount - максимальное число выдаваемых точек пересечений,
touchInclude - включать точки касания.

Выходные параметры:

kp - число пересечений,
xp, yp - координаты точки пересечения.

IntersectLinLin – Получить координаты точки пересечения двух прямых

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectLinLin.

Синтаксис:

```
void IntersectLinLin (double x1, double y1,
double angle1,
double x2, double y2,
double angle2,
int *kp,
double *xp, double *yp);
```

Входные параметры:

x1, y1 - координаты точки на первой прямой,
angle1 - угол наклона первой прямой,
x2, y2 - координаты точки на второй прямой,
angle2 - угол наклона второй прямой,

Выходной параметр:

kp - число пересечений (0 или 1),
xp, yp - координаты точки пересечения.

IntersectLinSArc – Получить координаты точек пересечения отрезка и дуги

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectLinSArc.

Синтаксис:

void IntersectLinSArc (double x1, double y1,

double x2, double y2,

double xc, double yc,

double rad,

double f1, double f2,

short direct,

int * kp,

double xp, double yp);

Входные параметры:

x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка,
xc, yc	- координаты центра дуги,
rad	- радиус дуги,
f1, f2	- углы начальной и конечной точек дуги,
direct	- направление дуги:
	- 1 - против часовой стрелки,
	- 1 - по часовой стрелке.

Выходные параметры:

kp	- число пересечений (от 0 до 2),
xp, yp	- координаты точки пересечения.

IntersectLinSCir – Получить координаты точек пересечения отрезка и окружности

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectLinSCir.

Синтаксис:

void IntersectLinSCir (double x1, double y1,

double x2, double y2,

double xc, double yc,

double rad,

int * kp,

double xp, double yp);

Входные параметры:

x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка,

xc, yc - координаты центра окружности,
rad - радиус окружности.

Выходные параметры:

kp - число пересечений (от 0 до 2),
xp, yp - координаты точки пересечения.

IntersectLinSLine – Получить координаты точки пересечения отрезка и прямой

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectLinSLine.

Синтаксис:

```
void IntersectLinSLine (double x1,
double y1,
double x2,
double y2,
double x,
double y,
double ang,
int *kp,
double *xp,
double *yp);
```

Входные параметры:

x1, y1 - координаты начальной точки отрезка,
x2, y2 - координаты конечной точки отрезка,
x, y - координаты точки на прямой,
ang - угол наклона прямой.

Выходные параметры:

kp - число пересечений (0 или 1),
xp, yp - координаты точки пересечения.

IntersectLinSLinS – Получить координаты точки пересечения двух отрезков

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectLinSLinS.

Синтаксис:

```
Void IntersectLinSLinS (double x11, double y11,
```

double x12, double y12,

double x21, double y21,

double x22, double y22,

int * kp,

double * xp, double *yp);

Входные параметры:

x11, y11 - координаты начальной точки первого отрезка,
x12, y12 - координаты конечной точки первого отрезка,
x21, y21 - координаты начальной точки второго отрезка,
x22, y22 - координаты конечной точки второго отрезка.

Выходные параметры:

kp - число пересечений (0 или 1),
xp, yp - координаты точки пересечения.

ksIntersectCurvCurv – Получить координаты точек пересечения двух кривых

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksIntersectCurvCurv.

Синтаксис:

int ksIntersectCurvCurv (reference p1, reference p2, reference array);

Входные параметры:

p1 - указатель на первую кривую,
p2 - указатель на вторую кривую.

Выходной параметр:

array - динамический массив точек пересечения MathPointParam.

Возвращаемое значение:

1 - пересечение данных объектов обнаружено,
0 - пересечение не обнаружено (кривые не пересекаются или совпадают),
-1 - первый объект не существует,
-2 - второй объект не существует,
-3 - кривые расположены в разных видах,
-4 - не совпадают СК определения кривых,
-5 - первый объект не является кривой,
-6 - второй объект не является кривой,

-7	- ошибка: массив типа POINT_ARR предварительно не создан.

Примечание :

Динамический массив точек пересечения array должен быть создан до передачи его в функцию.

ksIntersectCurvCurvEx – Получить координаты точек пересечения двух кривых

Синтаксис:

void IntersectCurvCurv (reference p1, reference p2, reference array, int touchInclude);

Входные параметры:

p1	- указатель на первую кривую,
p2	- указатель на вторую кривую,
touchInclude	- включать точки касания.

Выходной параметр:

array	- динамический массив точек пересечения MathPointParam.

Возвращаемое значение:

1	- пересечение данных объектов обнаружено,
0	- пересечение не обнаружено (кривые не пересекаются или совпадают),
-1	- первый объект не существует,
-2	- второй объект не существует,
-3	- кривые расположены в разных видах,
-4	- не совпадают СК определения кривых,
-5	- первый объект не является кривой,
-6	- второй объект не является кривой,
-7	- ошибка: массив типа POINT_ARR предварительно не создан.

Примечание:

Динамический массив точек пересечения array должен быть создан до передачи его в функцию.

Perpendicular – Получить координаты точки пересечения отрезка и перпендикуляра к нему, проходящего через заданную точку

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksPerpendicular.

Синтаксис:

void Perpendicular (double x, double y, double x1, double y1,

double x2, double y2, double *xp, double *yp);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка.

Выходные параметры:

xp, yp	- координаты точки пересечения отрезка и перпендикуляра к нему, проходящего через заданную точку.

Функции вычисления касаний и сопряжений

Функции данного раздела обеспечивают вычисление касаний и сопряжений геометрических объектов.

CouplingLineLine – Получить параметры окружностей, касательных к двум прямым

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCouplingLineLine.

Синтаксис:

void CouplingLineLine (double x1,
double y1,
double ang1,
double x2,
double y2,
double ang2,
double rad,
int *kp,
CON *con);

Входные параметры:

x1, y1	- координаты точки на первой прямой,
angl1	- угол наклона первой прямой,
x2, y2	- координаты точки на второй прямой,
angl2	- угол наклона второй прямой.

Выходные параметры:

rad	- радиус окружности сопряжения,
kp	-количество сопрягающих окружностей,
con	- Структура параметров сопрягающей окружности... (описана в LTDEFINE.H).

Примечание:

Возможное число сопряжений - от 0 до 4.

ksCouplingCircleCircle – Получить параметры сопрягающих окружностей определенного радиуса и точки сопряжения для двух окружностей

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCouplingCircleCircle.

Синтаксис:

```
void ksCouplingCircleCircle (double xc1,
double yc1,
double radc1,
double xc2,
double yc2,
double radc2,
double rad,
int *kp,
CON *con);
```

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
radc1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
radc2	- радиус второй окружности,
rad	- радиус окружности сопряжения.

Выходные параметры:

kp	-количество сопрягающих окружностей,
con	Структура параметров сопрягающей окружности... (описана в LTDEFINE.H).

Примечание:

Заполняются до 8 структур CON, если сопряжение существует.

ksCouplingLineCircle – Получить параметры сопрягающих окружностей определенного радиуса и точки касания при сопряжении окружности и прямой

Пример...

Аналог данной функции при использовании Automation - метод Mathematic2D::ksCouplingLineCircle.

Синтаксис:

void ksCouplingLineCircle (double xc,
double yc,
double radc,
double x1,
double y1,
double angle1,
double rad,
int *kp,
CON *con);

Входные параметры:

xc, yc	- координаты центра окружности,
radc	- радиус окружности,
x1, y1	- координаты точки на прямой,
angle1	- угол наклона прямой.

Выходные параметры:

rad	- радиус окружности сопряжения,
kp	-количество сопрягающих окружностей,
con	- Структура параметров сопрягающей окружности... (описана в LTDEFINE.H).

Примечание:

Заполняются до 8 структур CON, если сопряжение существует.

ksCouplingLineLine – Получить параметры сопряжения двух прямых

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCouplingLineLine.

Синтаксис:

void CouplingLineLine (double x1,
double y1,
double angle1,
double x2,
double y2,

double angle2,
double rad,
int *kp,
CON *con);

Входные параметры:

x1, y1	- координаты точки на первой прямой,
angle1	- угол наклона первой прямой,
x2, y2	- координаты точки на второй прямой,
angle2	- угол наклона второй прямой.

Выходные параметры:

rad	- радиус окружности сопряжения,
kp	-количество сопрягающих окружностей,
con	- Структура параметров сопрягающей окружности... (описана в LTDEFINE.H).

Примечание:

Заполняются 4 структуры con, если сопряжение есть.

ksTanLinePointCurve – Получить точки касания кривой и прямой, проведенной из заданной точки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1396_vspomogatelnye.htm#TANLINE_BY_OUTSIDE_PNT

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksTanLinePointCurve.

Синтаксис:

int ksTanLinePointCurve (double x,
double y,
reference pCur,
reference array);

Входные параметры:

pCur	- указатель на кривую,
x, y	- координаты точки.

Выходной параметр:

array - указатель на динамический массив точек касания MathPointParam.

Возвращаемое значение:

количество точек касания	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Динамический массив точек касания array должен быть создан до передачи его в функцию.

TanCircleCircle – Получить координаты точек касания прямых к двум окружностям

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksTanCircleCircle.

Синтаксис:

```
int TanCircleCircle (double xc1,
double yc1,
double radius1,
double xc2,
double yc2,
double radius2,
struct TAN tang);
```

Входные параметры:

xc1, yc1	- координаты центра первой окружности,
radius1	- радиус первой окружности,
xc2, yc2	- координаты центра второй окружности,
radius2	- радиус второй окружности.

Выходной параметр:

tang	- указатель на массив структур параметров касательных отрезков.

Описание:

Функция определяет точки касания прямых к двум окружностям. Структура типа TAN (описана в LIBTOOL.H) содержит координаты двух точек касания и определяет одну касательную прямую. Касательных прямых может быть от 0 до 4.

TanLineAngCircle – Получить точки касания окружности и прямой, проходящей под заданным углом

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksTanLineAngCircle.

Синтаксис:

void TanLineAngCircle (double xc,
double yc,
double rad,
double ang,
double xp[],
double yp[]);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
ang	- угол касательной прямой.

Выходные параметры:

xp, yp	- координаты точки касания.

TanLinePointCircle – Получить точки касания окружности и прямой, проходящей через заданную точку

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksTanLinePointCircle.

Синтаксис:

void TanLinePointCircle (double x,
double y,
double xc,
double yc,
double rad,
int *kp,
double xp,
double yp);

Входные параметры:

x, y	- координаты точки,
xc, yc	- координаты центра окружности,
rad	- радиус окружности.

Выходной параметр:

kp	- число касательных (от 0 до 2),
xp, yp	- координаты точки касания.

Примечание:

Построение касательной невозможно в случае, если точка лежит внутри окружности. В противном случае точек касания или две (точка вне окружности), или одна (если точка принадлежит окружности).

Функции вычисления длин, расстояний, углов и МЦХ

Функции данного раздела позволяют вычислять расстояния от точки до объекта, длины объектов, углы между объектами или между объектом и осью X и массово-центровочные характеристики.

Angle – Получить угол (в градусах) между осью OX и вектором, заданным двумя точками

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksAngle.

Синтаксис:

double Angle (double x1, double y1, double x2, double y2);

Входные параметры:

x1, y1	- координаты начальной точки вектора,
x2, y2	- координаты конечной точки вектора.

Возвращаемое значение:

угол в градусах.	- в случае успеха,
0	- в случае неудачи.

DistancePntPnt – Получить расстояние между двумя точками

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntPnt.

Синтаксис:

double DistancePntPnt (double x1, double y1, double x2, double y2);

Входные параметры:

x1, y1	- координаты первой точки,
x2, y2	- координаты второй точки.

Возвращаемое значение:

расстояние между двумя точками. - в случае успеха,
0 - в случае неудачи.

ksCalcInertiaProperties – Получить плоские массово-центровочные характеристики кривой или группы кривых

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MIX.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCalcInertiaProperties.

Синтаксис:

int ksCalcInertiaProperties (reference p,
InertiaParam * properties,
unsigned char dimension);

Входные параметры:

p - указатель на кривую или группу кривых,
dimension - размерность длины.

Размерности длины...

Выходные параметры:

properties - указатель на структуру параметров для расчета плоских МЦХ.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Кривые должны быть замкнутыми и представлять собой наружные и внутренние контура.

ksCalcMassInertiaProperties – Получить объемные массово-центровочные характеристики тел вращения или выдавливания, заданных кривой или группой кривых

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_MIX3D.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksCalcMassInertiaProperties.

Синтаксис:

int ksCalcMassInertiaProperties (reference p,
MassInertiaParam * properties,
unsigned int bitVector,
double density,
double param);

Входные параметры:

p	- указатель на кривую или группу кривых,
density	- плотность тела (г/куб.мм),
param	- параметр тела (для тела вращения - угол раствора в градусах, для тела выдавливания - толщина),
bitVector	- структура параметров, определяющая размерности длины, массы и типы деталей.

Выходные параметры:

properties	- указатель на структуру параметров для расчета МЦХ объемных тел.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

Кривые должны быть замкнутыми и представлять собой наружные и внутренние контура.

ksDistancePntArc – Получить расстояние между точкой и дугой

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntArc.

Синтаксис:

double ksDistancePntArc (double x, double y,
double xac, double yac,
double rada,

double fa1, double fa2,
short directa);

Входные параметры:

x, y	- координаты точки,
xac, yac	- координаты центра дуги,
rada	- радиус дуги,
fa1, fa2	- начальный и конечный углы дуги,
directa	- направление дуги:
	1 - против часовой стрелки,
	-1 – по часовой стрелке.

Возвращаемое значение:

расстояние между точкой и дугой.	- в случае успеха,
0	- в случае неудачи.

ksDistancePntCircle – Получить расстояние между точкой и окружностью

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntCircle.

Синтаксис:

double ksDistancePntCircle (double x,
double y,
double xc, double yc,
double rad);

Входные параметры:

x, y	- координаты точки,
xc, yc	- координаты центра окружности,
rad	- радиус окружности.

Возвращаемое значение:

расстояние между точкой и окружностью.	- в случае успеха,
0	- в случае неудачи.

ksDistancePntLine – Получить расстояние между точкой и прямой, заданной точкой и углом

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntLine.

Синтаксис:

double ksDistancePntLine (double x,
double y,
double x1,
double y1,
double angle);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты точки на прямой,
angle	- угол наклона прямой.

Возвращаемое значение:

расстояние между точкой и прямой.	- в случае успеха,
0	- в случае неудачи.

ksDistancePntLineForPoint – Получить расстояние между точкой и прямой, заданной двумя точками

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntLineForPoint.

Синтаксис:

double ksDistancePntLineForPoint (double x,
double y,
double x1,
double y1,
double x2,
double y2);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты первой точки на прямой,
x2, y2	- координаты второй точки на прямой.

Возвращаемое значение:

расстояние между точкой и прямой.	- в случае успеха,
0	- в случае неудачи.

ksDistancePntLineSeg – Получить расстояние между точкой и отрезком

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntLineSeg.

Синтаксис:

double ksDistancePntLineSeg (double x, double y,

double x1, double y1,

double x2, double y2);

Входные параметры:

x, y	- координаты точки,
x1, y1	- координаты начальной точки отрезка,
x2, y2	- координаты конечной точки отрезка.

Возвращаемое значение:

расстояние между точкой и отрезком.	- в случае успеха,
0	- в случае неудачи.

ksDistancePntPntOnCurve – Получить расстояние между двумя точками на кривой

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MEASURE_DISTANCE_2POINTS_BYCURVE.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistancePntPntOnCurve.

Синтаксис:

double ksDistancePntPntOnCurve (reference curve,

double x1, double y1,

double x2, double y2);

Входные параметры:

curve	- указатель на кривую,
x1, y1	- координаты первой точки,
x2, y2	- координаты второй точки.

Возвращаемое значение:

расстояние между двумя точками на кривой	- в случае успеха,

0 - в случае неудачи.

Примечание:

Если заданные точки не находятся на кривой, то определяется расстояние между их проекциями на кривую.

ksGetCurvePerimeter – Получить длину периметра кривой

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_PERIMETER.htm

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksGetCurvePerimeter.

Синтаксис:

double ksGetCurvePerimeter (reference curve,

unsigned char dimension);

Входные параметры:

curve - указатель на кривую,
dimension - размерность длины.

Размерности длины...

Возвращаемое значение:

длина периметра кривой - в случае успеха,
0 - в случае неудачи.

ksViewGetDensity – Выбрать из диалога значение плотности

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/dlg_mix_density.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksViewGetDensity.

Синтаксис:

double ksViewGetDensity (void * HWindow);

Входной параметр:

HWindow - дескриптор "родительского" окна или NULL.

Возвращаемое значение:

плотность (г/куб.мм)	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Плотность выбирается из системного файла плотностей graphic.dns.

ksViewGetDensityAndMaterial – Выбрать из диалога плотность и наименование материала

Справка системы КОМПАС...

`KOMPAS.chm::/dlg_mix_density.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksViewGetDensityAndMaterial.

Синтаксис:

double ksViewGetDensityAndMaterial (char * material, int sizeStr, void *HWindow);

Входные параметры:

HWindow	- дескриптор "родительского" окна,
sizeStr	- размер строки material.

Выходные параметры:

material	- наименование материала.

Возвращаемое значение:

плотность (г/куб.мм)	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Плотность выбирается из системного файла плотностей graphic.dns.

При использовании Unicode следует использовать функцию ksViewGetDensityAndMaterialW.

ksViewGetDensityAndMaterialW – Выбрать из диалога плотность и наименование материала (Unicode)

Справка системы КОМПАС...

`KOMPAS.chm::/dlg_mix_density.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksViewGetDensityAndMaterial.

Синтаксис:

double LIB_FUNC ksViewGetDensityAndMaterial (LPWSTR material, int sizeStr, void *HWindow);

Входные параметры:

HWindow	- дескриптор "родительского" окна,
sizeStr	- размер строки material.

Выходные параметры:

material	- наименование материала.

Возвращаемое значение:

плотность (г/куб.мм)	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Плотность выбирается из системного файла плотностей graphic.dns.
2. При использовании ANSI следует использовать функцию ksViewGetDensityAndMaterial.

ReadFragment – Прочитать фрагмент в текущий вид

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_INSERTFRAGMENT.htm`

Аналог данной функции при использовании Automation - метод ksFragment::ksReadFragment.

Синтаксис:

int ReadFragment (char *fileName,
unsigned char currentLayer,
PlacementParam * par);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на структуру параметров привязки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец",
где
c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

2. При использовании Unicode следует использовать функцию ReadFragmentW.

ReadFragmentW – Прочитать фрагмент в текущий вид (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksReadFragment.

Синтаксис:

int LIB_FUNC ReadFragmentW (LPWSTR fileName,
unsigned char currentLayer,
PlacementParam * par);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на структуру параметров привязки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где
c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

2. При использовании ANSI следует использовать функцию ReadFragment.

Функции работы с фрагментами и библиотеками фрагментов

Функции работы с фрагментами

Справка системы КОМПАС: Фрагменты...

KOMPAS.chm::/454_Glava52_Obshchie_svedenija_.htm

Библиотеки фрагментов...

KOMPAS.chm::/617_Glava75_Biblioteka_fragment.htm

Функции данного раздела позволяют работать с фрагментами и библиотеками фрагментов.

Фрагменты

FragmentDefinition – Определить данные для вставки фрагмента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/456_Glava53_Fragmenty_v_grafich.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksFragmentDefinition.

Синтаксис:

reference FragmentDefinition (char * fileName,
char * comment,
unsigned char insertType);

Входные параметры:

fileName	- имя файла фрагмента,
comment	- имя вставки,
insertType	- тип вставки (действителен для внешнего фрагмента): 0 - взять в документ, 1 - внешней ссылкой.

Возвращаемое значение:

указатель на определение фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция определяет данные для вставки фрагмента (внешнего или находящегося непосредственно в документе) с указанным именем по ссылке.
2. Непосредственно вставка фрагмента (вставка ссылки) производится при помощи метода ksFragment::ksInsertFragment.
3. Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где:

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

В этом случае определение создается для фрагмента из библиотеки фрагментов.

4. Определение для данного фрагмента одно; вставок может быть сколько угодно.
5. При использовании Unicode следует использовать функцию FragmentDefinitionW.

FragmentDefinitionW – Определить данные для вставки фрагмента (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/456_Glava53_Fragmenty_v_grafich.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksFragmentDefinition.

Синтаксис:

reference LIB_FUNC FragmentDefinitionW (LPWSTR fileName,
LPWSTR comment,
unsigned char insertType);

Входные параметры:

fileName	- имя файла фрагмента,
comment	- имя вставки,
insertType	- тип вставки (действителен для внешнего фрагмента): 0 - взять в документ, 1 - внешней ссылкой.

Возвращаемое значение:

указатель на определение фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция определяет данные для вставки фрагмента (внешнего или находящегося непосредственно в документе) с указанным именем по ссылке.
2. Непосредственно вставка фрагмента (вставка ссылки) производится при помощи метода ksFragment::ksInsertFragment.

3. Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где:

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

В этом случае определение создается для фрагмента из библиотеки фрагментов.

4. Определение для данного фрагмента одно; вставок может быть сколько угодно.
5. При использовании ANSI следует использовать функцию FragmentDefinition.

InsertFragment – Вставить фрагмент в документ

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/456_Glava53_Fragmenty_v_grafich.htm

Замечание:

Функция устарела, рекомендуется использовать вместо нее функцию ksInsertFragmentEx.

Аналог данной функции при использовании Automation - метод ksFragment::ksInsertFragment.

Синтаксис:

reference InsertFragment (reference p,

unsigned char currentLayer,

PlacementParam * par);

Входные параметры:

p	- указатель на фрагмент,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на структуру параметров привязки

Возвращаемое значение:

указатель на вставку фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

ksInsertFragmentEx – Вставить фрагмент в документ

Справка системы КОМПАС...

KOMPAS.chm::/456_Glava53_Fragmenty_v_grafich.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksInsertFragmentEx.

Синтаксис:

reference ksInsertFragmentEx (reference p,

unsigned char currentLayer,

PlacementParam * par,

unsigned char scaleProjLinesSize);

Входные параметры:

p	- указатель на фрагмент,
curentLayer	- тип размещения объектов фрагмента по слоям:
	0 - на "свои" слои,
	1- на текущий слой,
par	- указатель на структуру параметров привязки
scaleProjLinesSize	- признак масштабирования длины выносных линий размеров:
	1 - выносные линии масштабируются,
	0 - выносные линии не масштабируются.

Возвращаемое значение:

указатель на вставку фрагмента	- в случае успешного завершения,
0	- в случае неудачи.

ksReadFragmentToGroup – Вставить фрагмент россыпью во временную группу

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Замечание:

Функция устарела, рекомендуется использовать вместо нее функцию ksReadFragmentToGroupEx.

Аналог данной функции при использовании Automation - метод ksFragment::ksReadFragmentToGroup.

Синтаксис:

reference ksReadFragmentToGroup (char * fileName,

unsigned char curentLayer,

PlacementParam * par);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям:
	0 - на "свои" слои,
	1- на текущий слой,
par	- указатель на структуру параметров привязки.

Возвращаемое значение:

указатель на временную группу	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где:
c:\gr\lib1.lfr - имя файла библиотеки фрагментов,

|детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

ksReadFragmentToGroupEx – Вставить фрагмент россыпью во временную группу

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksReadFragmentToGroupEx.

Синтаксис:

reference ksReadFragmentToGroupEx (char * fileName,

unsigned char curentLayer,

PlacementParam * par.

unsigned char scaleProjLinesSize);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям:
	0 - на "свои" слои,
	1- на текущий слой,
par	- указатель на структуру параметров привязки,
scaleProjLinesSize	- признак масштабирования длины выносных линий размеров:
	1 - выносные линии масштабируются,
	0 - выносные линии не масштабируются.

Возвращаемое значение:

указатель на временную группу	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

В этом случае фрагмент берется из библиотеки фрагментов.

2. При использовании Unicode следует использовать функцию ksReadFragmentToGroupExW.

ksReadFragmentToGroupExW – Вставить фрагмент россыпью во временную группу (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksReadFragmentToGroupEx.

Синтаксис:

reference LIB_FUNC ksReadFragmentToGroupExW (LPWSTR fileName,

unsigned char curentLayer,

PlacementParam * par.

unsigned char scaleProjLinesSize);

Входные параметры:

fileName	- имя фрагмента,
curentLayer	- тип размещения объектов фрагмента по слоям: 0 - на "свои" слои, 1- на текущий слой,
par	- указатель на структуру параметров привязки
scaleProjLinesSize	- признак масштабирования длины выносных линий размеров: 1 - выносные линии масштабируются, 0 - выносные линии не масштабируются.

Возвращаемое значение:

указатель на временную группу	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается задание имени файла следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,

- фланец - имя фрагмента.

  В этом случае фрагмент берется из библиотеки фрагментов.

2. При использовании ANSI следует использовать функцию ksReadFragmentToGroupEx.

LocalFragmentDefinition – Начать определение локального фрагмента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksLocalFragmentDefinition.

Синтаксис:

int LocalFragmentDefinition(char * comment);

Входной параметр:

comment - имя вставки.

Возвращаемое значение:

указатель на определение фрагмента - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Вставка ссылки на фрагмент производится с помощью функции InsertFragment. Указатель на определение фрагмента возвращается при окончании описания - функцией CloseLocalFragmentDefinition. Отдельного файла фрагмента нет; "тело" фрагмента хранится в документе.
2. Все объекты вида, вводимые между функциями LocalFragmentDefinition и CloseLocalFragmentDefinition, принадлежат локальному фрагменту.
3. При использовании Unicode следует использовать функцию LocalFragmentDefinitionW.

LocalFragmentDefinitionW – Начать определение локального фрагмента (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksLocalFragmentDefinition.

Синтаксис:

int LIB_FUNC LocalFragmentDefinitionW(LPWSTR comment);

Входной параметр:

comment - имя вставки.

Возвращаемое значение:

указатель на определение фрагмента - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Вставка ссылки на фрагмент производится с помощью функции InsertFragment. Указатель на определение фрагмента возвращается при окончании описания функцией CloseLocalFragmentDefinition. Отдельного файла фрагмента нет; "тело" фрагмента хранится в документе.
2. Все объекты вида, вводимые между функциями LocalFragmentDefinition и CloseLocalFragmentDefinition, принадлежат локальному фрагменту.
3. При использовании ANSI следует использовать функцию LocalFragmentDefinition.

WriteFragment – Записать группу во фрагмент

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksWriteFragment.

Синтаксис:

int WriteFragment (reference gr,

char *filename,

char *comment,

double xb,

double yb);

Входные параметры:

gr - указатель на группу
(0 - записывается группа выделения),
filename - имя файла фрагмента,
comment - комментарий для фрагмента,
xb, yb - координаты точки привязки.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию WriteFragmentW.

WriteFragmentW – Записать группу во фрагмент (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTFRAGMENT.htm

Аналог данной функции при использовании Automation - метод ksFragment::ksWriteFragment.

Синтаксис:

int LIB_FUNC WriteFragment (reference gr,

LPWSTR filename,

LPWSTR comment,

double xb,

double yb);

Входные параметры:

gr	- указатель на группу (0 - записывается группа выделения),
filename	- имя файла фрагмента,
comment	- комментарий для фрагмента,
xb, yb	- координаты точки привязки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию WriteFragment.

Библитеки фрагментов

ksAddFragmentToLibrary – Добавить фрагмент в библиотеку фрагментов

Функция не поддерживается

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/618_75_1_Sozdanie_biblioteki_fr.htm

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksAddFragmentToLibrary

Синтаксис:

int ksAddFragmentToLibrary (char * libName, char * frwName);

Входные параметры:

libName	- имя фрагмента в библиотеке,
frwName	- имя файла фрагмента.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец", где:

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

2. При использовании Unicode следует использовать функцию ksAddFragmentToLibraryW.

ksAddFragmentToLibraryW – Добавить фрагмент в библиотеку фрагментов (Unicode)

Функция не поддерживается

Справка системы КОМПАС...

KOMPAS.chm::/618_75_1_Sozdanie_biblioteki_fr.htm

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksAddFragmentToLibrary.

Синтаксис:

int LIB_FUNC ksAddFragmentToLibraryW (LPWSTR libName, LPWSTR frwName);

Входные параметры:

libName	- имя фрагмента в библиотеке,
frwName	- имя файла фрагмента.

Возвращаемое значение:

1	- в случае успешного завершения,

0 - в случае неудачи.

Примечание:

1. Допускается задание имени фрагмента следующего вида: "c:\gr\lib1.lfr|детали|литье|фланец",где:

- c:\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье| - разделы, подразделы внутри библиотеки фрагментов,
- фланец - имя фрагмента.

2. При использовании ANSI следует использовать функцию ksAddFragmentToLibrary.

ksCheckFragmentLibrary – Проверить, открыта ли библиотека фрагментов

Функция не поддерживается

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/617_Glava75_Biblioteka_fragment.htm

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksCheckFragmentLibrary.

Синтаксис:

int ksCheckFragmentLibrary (char * libName, unsigned char possibleMessage);

Входные параметры:

libName - имя файла библиотеки фрагментов,
possibleMessage - признак выдачи сообщения:
- 1 - сообщать, если библиотека уже открыта,
- 0 - не сообщать.

Возвращаемое значение:

1 - если библиотека открыта,
0 - если библиотека не открыта.

Примечание.

При использовании Unicode следует использовать функцию ksCheckFragmentLibraryW.

ksCheckFragmentLibraryW – Проверить, открыта ли библиотека фрагментов (Unicode)

Функция не поддерживается

Справка системы КОМПАС...

KOMPAS.chm::/617_Glava75_Biblioteka_fragment.htm

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksCheckFragmentLibrary.

Синтаксис:

int LIB_FUNC ksCheckFragmentLibraryW (LPWSTR libName, unsigned char possibleMessage);

Входные параметры:

libName	- имя файла библиотеки фрагментов,
possibleMessage	- признак выдачи сообщения: - 1 - сообщать, если библиотека уже открыта, - 0 - не сообщать.

Возвращаемое значение:

1	- если библиотека открыта,
0	- если библиотека не открыта.

Примечание.

При использовании ANSI следует использовать функцию ksCheckFragmentLibrary.

ksChoiceFragmentFromLib – Выбрать имя фрагмента, хранящегося в библиотеке фрагментов

Пример...

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksChoiceFragmentFromLib

Синтаксис:

int ksChoiceFragmentFromLib (char * fileLibFrw,

char * nameFrw,

unsigned int size);

Входные параметры:

fileLibFrw	- полное имя библиотеки фрагментов,
size	- размер буфера фрагмента.

Выходные параметры:

nameFrw	- имя фрагмента.

Возвращаемое значение:

3	- имя фрагмента,
2	- имя папки,
1	- корень библиотеки фрагментов,
0	- в случае неудачи.

Примечание:

1. Библиотека фрагментов открывается в режиме диалога. Ее редактирование запрещено. Нажатие <Enter> - выбор имени фрагмента. <Esc> - отказ.
2. При использовании Unicode следует использовать функцию ksChoiceFragmentFromLibW.

ksChoiceFragmentFromLibW – Выбрать имя фрагмента, хранящегося в библиотеке фрагментов (Unicode)

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksChoiceFragmentFromLib

Синтаксис:

int LIB_FUNC ksChoiceFragmentFromLib (LPWSTR fileLibFrw,

LPWSTR nameFrw,

unsigned int size);

Входные параметры:

fileLibFrw	- полное имя библиотеки фрагментов,
size	- размер буфера фрагмента.

Выходные параметры:

nameFrw	- имя фрагмента.

Возвращаемое значение:

3	- имя фрагмента,
2	- имя папки,
1	- корень библиотеки фрагментов,
0	- в случае неудачи.

Примечание:

1. Библиотека фрагментов открывается в режиме диалога. Ее редактирование запрещено. Нажатие <Enter> - выбор имени фрагмента. <Esc> - отказ.
2. При использовании ANSI следует использовать функцию ksChoiceFragmentFromLib.

ksCreateInsertionFragment – Создать вставку фрагмента

Синтаксис:

int LIB_FUNC ksCreateInsertionFragment(char * fileName);

Входной параметр:

fileName - полное имя файла фрагмента.

Возвращаемое значение:

1 - в случае удачи.

Примечание:

Допускается fileName следующего вида "c:_f\gr\lib1.lfr|детали|литье|фланец", где:

- c:_f\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье|- разделы, подразделы внутри библиотеки,
- фланец - имя фрагмента или имя подраздела).

Функция запускает процесс вставки фрагмента из файла. Функция не ждет завершения процесса вставки. Достаточно передать только имя библиотеки фрагментов. В этом случае процесс запустится с начальным каталогом библиотеки.

ksCreateInsertionFragmentW – Создать вставку фрагмента (Unicode)

Синтаксис:

int LIB_FUNC ksCreateInsertionFragmentW(LPWSTR * fileName);

Входной параметр:

fileName - полное имя файла фрагмента.

Возвращаемое значение:

1 - в случае удачи.

Примечание:

Допускается fileName следующего вида "c:_f\gr\lib1.lfr|детали|литье|фланец", где:

- c:_f\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье|- разделы, подразделы внутри библиотеки,
- фланец - имя фрагмента или имя подраздела).

Функция запускает процесс вставки фрагмента из файла. Функция не ждет завершения процесса вставки. Достаточно передать только имя библиотеки фрагментов. В этом случае процесс запустится с начальным каталогом библиотеки.

ksExistFragmentInLibrary – Проверить наличие указанного фрагмента или раздела в библиотеке фрагментов

Пример...

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksExistFragmentInLibrary.

Синтаксис:

int ksExistFragmentInLibrary (char *frwName);

Входной параметр:

frwName - имя фрагмента.

Возвращаемое значение:

1 - указанный фрагмент или папка есть в библиотеке,
0 - указанного фрагмента или папки нет,
-1 - нет указанной библиотеки.

Примечание:

Пример формирования имени фрагмента:

- "c:_f\gr\lib1.lfr|детали|литье|фланец", где
- c:_f\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье|- разделы, подразделы внутри библиотеки,
- фланец - имя фрагмента или имя подраздела).

ksExistFragmentInLibraryW – Проверить наличие указанного фрагмента или раздела в библиотеке фрагментов (Unicode)

Пример...

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksExistFragmentInLibrary.

Синтаксис:

int LIB_FUNC ksExistFragmentInLibraryW (LPWSTR frwName);

Входной параметр:

frwName - имя фрагмента.

Возвращаемое значение:

1 - указанный фрагмент или папка есть в библиотеке,
0 - указанного фрагмента или папки нет,

-1 - нет указанной библиотеки.

Примечание:

Пример формирования имени фрагмента:

- "c:_f\gr\lib1.lfr|детали|литье|фланец", где
- c:_f\gr\lib1.lfr - имя файла библиотеки фрагментов,
- |детали|литье|- разделы, подразделы внутри библиотеки,
- фланец - имя фрагмента или имя подраздела).

ksFragmentLibrary – Открыть библиотеку фрагментов

Функция не поддерживается

Пример...

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksFragmentLibraryOperation.

Синтаксис:

int ksFragmentLibrary (char * libName, int type);

Входные параметры:

libName	- имя файла библиотеки фрагментов,
type	- тип действия с библиотекой фрагментов.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. При выполнении операций типа -1, 0, 1, 4, 5 libName - полное имя файла библиотеки фрагментов.

   При выполнении операций типа 2, 3 libName - полное имя файла библиотеки фрагментов путь внутри библиотеки с именем редактируемого фрагмента.
2. При использовании Unicode следует использовать функцию ksFragmentLibraryW.

ksFragmentLibraryW – Открыть библиотеку фрагментов (Unicode)

Функция не поддерживается

Аналог данной функции при использовании Automation - метод ksFragmentLibrary::ksFragmentLibraryOperation.

Синтаксис:

int LIB_FUNC ksFragmentLibraryW (LPWSTR libName, int type);

Входные параметры:

libName	- имя файла библиотеки фрагментов,
type	- тип действия с библиотекой фрагментов.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. При выполнении операций типа -1, 0, 1, 4, 5 libName - полное имя файла библиотеки фрагментов.
2. При выполнении операций типа 2, 3 libName - полное имя файла библиотеки фрагментов путь внутри библиотеки с именем редактируемого фрагмента.
3. При использовании ANSI следует использовать функцию ksFragmentLibrary.

ksIsLibraryEnabled – Проверить защиту библиотеки фрагментов и моделей

Аналог данной функции при использовании Automation - метод KompasObject::ksIsLibraryEnabled.

Синтаксис:

int ksIsLibraryEnabled (char * libName);

Входные параметры:

libName	- имя библиотеки либо полное, либо относительно папки LIBS.

Возвращаемое значение:

1	- выполнять библиотеку можно,
0	- выполнять библиотеку нельзя.

Примечание:

1. Функция применима для прикладных библиотек (*.rtw), библиотек фрагментов (*.lfr), библиотек моделей (*.l3d).
2. При использовании Unicode следует использовать функцию ksIsLibraryEnabledW.

ksIsLibraryEnabledW – Проверить защиту библиотеки фрагментов и моделей (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksIsLibraryEnabled.

Синтаксис:

int LIB_FUNC ksIsLibraryEnabledW (LPWSTR libName);

Входные параметры:

libName	- имя библиотеки либо полное, либо относительно папки LIBS.

Возвращаемое значение:

1	- выполнять библиотеку можно,
0	- выполнять библиотеку нельзя.

Примечание:

Функция применима для прикладных библиотек (*.rtw), библиотек фрагментов (*.lfr), библиотек моделей (*.l3d).

При использовании ANSI следует использовать функцию ksIsLibraryEnabled.

Создание графических объектов

Графические примитивы

Функции данного радела обеспечивают построение геометрических примитивов:

точек, прямых, отрезков, окружностей, дуг, кривых, штриховки.

ArcByAngle – Создать дугу по двум точкам и углу раствора

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ARC_2_PNT_ANGLE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksArcByAngle.

Синтаксис:

reference ArcByAngle (double xc,

double yc,

double rad,

double f1,

double f2,

int n,

int type);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
f1, f2	- начальный и конечный угол дуги в градусах,

n	- направление отрисовки дуги: 1 - против часовой стрелки, -1 - по часовой стрелке,
type	- стиль линии.

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

ArcBy3Points – Создать дугу окружности по трем точкам

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ARC_3_PNT.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksArcBy3Points.

Синтаксис:

```
reference ArcBy3Points (double x1,
 double y1,
 double x2,
 double y2,
 double x3,
 double y3,
 int style);
```

Входные параметры:

x1, y1	- координаты начальной точки на дуге,
x2, y2	- координаты средней точки на дуге,
x3, y3	- координаты конечной точки на дуге,
style	- стиль линии.

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

ArcByPoint – Создать дугу по центру и конечным точкам

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CIRCLEARC.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksArcByPoint.

Синтаксис:

ArcByPoint (double xc,

double yc,

double rad,

double x1,

double y1,

double x2,

double y2,

short direction,

unsigned short style);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
x1, y1	- координаты начальной точки дуги,
x2, y2	- координаты конечной точки дуги,
direction	- направление отрисовки дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке,
type	- стиль линии.

Возвращаемое значение:

указатель на дугу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

Circle – Создать окружность

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CIRCLE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCircle.

Синтаксис:

reference Circle (double xc, double yc,

double rad,

int type);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
type	- стиль линии.

Возвращаемое значение:

указатель на окружность	- в случае удачного завершения,
0	- в случае неудачи.

Equidistant – Построить эквидистанту

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_EQUID_TO_OBJ.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEquidistant.

Синтаксис:

reference Equidistant (EquidistantParam *par);

Выходной параметр:

par	- указатель на структуру параметров эквидистанты.

Возвращаемое значение:

указатель на эквидистанту	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

Hatch – Создать штриховку

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_HATCH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksHatch.

Синтаксис:

int Hatch (unsigned int style,

double ang,

double step,

double width,

double x0, double y0);

Входные параметры:

style	- стиль штриховки,
angle	- угол штриховки в градусах,
step	- шаг штриховки,
width	- ширина полосы штрихования вдоль границы штриховки,
x0, y0	- координаты начальной точки штриховки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Все следующие далее до вызова функции EndObj геометрические примитивы определяют границы штриховки (внешние и внутренние). Порядок определения элементов границы является произвольным. Функция EndObj возвращает указатель на объект Штриховка.
2. Функция устарела, рекомендуется вместо нее использовать функцию HatchEx.

HatchEx – Создать штриховку

Справка системы КОМПАС...

`KOMPAS.chm::/CM_HATCH.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksHatch.

Синтаксис:

int HatchEx(unsigned short style,

double angle,

double step,

double width,

double x0, double y0,

unsigned long color,

unsigned char sheetAng);

Входные параметры:

style	- стиль штриховки,
angle	- угол штриховки в градусах,
step	- шаг штриховки,
width	- ширина полосы штрихования вдоль границы штриховки,

x0, y0 - координаты начальной точки штриховки,
color - цвет штриховки, по умолчанию FREE_COLOR (0xff000000l),
sheetAng - тип угла штриховки:
0 - угол штриховки относительно ее границ сохраняется при повороте границ (используется при изображении накатки на деталях),
1 - обычная штриховка (угол штриховки - постоянный).

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Все, следующие далее до вызова функции EndObj, геометрические примитивы определяют границы штриховки (внешние и внутренние). Порядок определения элементов границы является произвольным. Функция EndObj возвращает указатель на объект Штриховка.

ksColouring – Создать фоновую заливку цветом

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_COLOURING.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksColouring.

Синтаксис:

int ksColouring (unsigned long color);

Входной параметр:

color - цвет заливки.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Заливка - составной объект. Объекты вида, вводимые между операторами ksColouring и EndObj, принадлежат заливке и образуют ее границу. EndObj возвращает указатель на заливку.
2. Если color = -1 или 0xFFFFFFFF - создается заливка цветом фона документа.

ksColouringEx – Создать фоновую заливку цветом

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_COLOURING.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksColouringEx.

Синтаксис:

reference ksColouringEx (unsigned long color, reference group);

Входные параметры:

color	- цвет заливки,
group	- группа, образующая границу заливки.

Возвращаемое значение:

указатель на заливку	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если color = -1 или 0xFFFFFFFF, то создается заливка цветом фона документа.

ksConicArc – Построить коническое сечение

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksConicArc.

Синтаксис:

reference ksConicArc (ConicArcParam * par);

Входной параметр:

par	- указатель на структуру параметров конического сечения ConicArcParam

Возвращаемое значение:

указатель на коническое сечение	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Коническим сечением может быть дуга окружности, дуга эллипса или кривая NURBS.

ksCreateViewObject – Создать объект, используя визуальный процесс

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateViewObject.

Синтаксис:

reference ksCreateViewObject (unsigned short int type);

Входной параметр:

type	- тип объекта (см. ltdefine.h LINESEG_OBJ...ORDINATEDIMENSION_OBJ).

Возвращаемое значение:

	- указатель на созданный объект.

Примечание:

Функция распространяется на все объекты вида, кроме:

LAYER_OBJ	-слой,
CONTOUR_OBJ	-контур,
MACRO_OBJ	-нетипизированный макроэлемент,
FRAGMENT_OBJ	-вставной фрагмент,
ELLIPSE_ARC_OBJ	-дуга эллипса.

ksEllipse – Создать эллипс с заданными параметрами

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ELLIPSE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEllipse.

Синтаксис:

reference ksEllipse (EllipseParam *par);

Выходной параметр:

par	- указатель на Структуру параметров эллипса.

Возвращаемое значение:

указатель на эллипс	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

ksEllipseArc – Построить дугу эллипса

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEllipseArc.

Синтаксис:

reference ksEllipseArc (EllipseArcParam *par);

Входной параметр:

par - указатель на Структуру параметров дуги эллипса.

Возвращаемое значение:

указатель на дугу эллипса - в случае удачного завершения,
0 - в случае неудачи.

ksHatch – Создать штриховку с заданными параметрами

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_HATCH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksHatchByParam.

Синтаксис:

reference ksHatch (HatchParam * par);

Входной параметр:

par - указатель на структуру параметров штриховки HatchParam.

Возвращаемое значение:

указатель на штриховку - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Объекты границы штриховки лежат во временной группе pBoundaries. После выполнения функции ksHatch временная группа прекращает существование.

ksHatchEx – Создать штриховку с заданными параметрами

Справка системы КОМПАС...

KOMPAS.chm::/CM_HATCH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksHatchByParam.

Синтаксис:

reference ksHatchEx (HatchParamEx * par);

Входные параметры:

par - указатель на структуру параметров штриховки HatchParamEx.

Возвращаемое значение:

указатель на штриховку - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Объекты границы штриховки лежат во временной группе pBoundaries.
2. После выполнения функции ksHatchEx временная группа прекращает существование.

ksInsertRaster – Вставить растровый объект

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTRASTER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksInsertRaster.

Синтаксис:

reference ksInsertRaster (RasterParam *par);

Входной параметр:

par - Структура параметров растрового объекта RasterParam.

Возвращаемое значение:

указатель на растровый объект - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksInsertRasterW.

ksInsertRasterW – Вставить растровый объект (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_INSERTRASTER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksInsertRaster.

Синтаксис:

reference ksInsertRasterW (RasterParamW *par);

Входной параметр:

par - Структура параметров растрового объекта RasterParamW.

Возвращаемое значение:

указатель на растровый объект - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksInsertRaster.

ksParEllipseArc – Построить дугу эллипса

Синтаксис:

reference ksParEllipseArc (EllipseArcParam1 *par);

Входной параметр:

par - указатель на Структуру параметров дуги эллипса.

Возвращаемое значение:

указатель на дугу эллипса - в случае удачного завершения,
0 - в случае неудачи.

ksPointsOnCurveByStep – Получить массив точек, расположенных на кривой с заданным шагом

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1391_Postroenie_tochek.htm#point_on_distance

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksPointsOnCurveByStep.

Синтаксис:

reference ksPointsOnCurveByStep (reference curve,

double step);

Входные параметры:

curve	- указатель на кривую,
step	- шаг точек вдоль кривой.

Возвращаемое значение:

указатель на динамический массив математических точек ksDynamicArray типа POINT_ARR.	- в случае успеха,
0	- в случае неудачи.

ksRectangle – Построить прямоугольник

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRectangle.

Синтаксис:

reference ksRectangle (RectangleParam * par,

unsigned char centre);

Входные параметры:

par	- указатель на структуру параметров прямоугольника RectangleParam,
centre	- признак построения обозначения центра: 0 - нет осей, 1 - значок осей (маленький "крестик"), 2 - горизонтальная ось, 3 - обе оси.

Возвращаемое значение:

указатель на прямоугольник	- в случае удачного завершения,
0	- в случае неудачи.

ksRegularPolygon – Создать правильный многоугольник с заданными параметрами

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Mnogougolqnik.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRegularPolygon.

Синтаксис:

reference ksRegularPolygon (RegularPolygonParam * par,

unsigned char centre);

Входные параметры:

centre	- признак построения обозначения центра: 0 - нет осей, 1 - значок осей (маленький "крестик"), 2 - горизонтальная ось, 3 - обе оси.

Выходные параметры:

par	- указатель на структуру параметров правильного многоугольника.

Возвращаемое значение:

указатель на многоугольник	- в случае удачного завершения,
0	- в случае неудачи.

Line – Создать прямую линию через указанную точку под заданным углом

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1396_vspomogatelnye.htm#simple_line

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLine.

Синтаксис:

reference Line (double x, double y,

double ang);

Входные параметры:

x, y	- координаты точки,
ang	- угол наклона прямой.

Возвращаемое значение:

указатель на графический объект Линия.	- в случае успеха,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида и используется для вспомогательных построений.

LineSeg – Создать отрезок прямой линии

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1418_Postroenie_otrezkov_.htm#otr_dve_tochki

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLineSeg.

Синтаксис:

reference LineSeg (double x1,

double y1,

double x2,

double y2,

unsigned int style);

Входные параметры:

x1, y1	- координаты первой точки отрезка,
x2, y2	- координаты второй точки отрезка,
style	- стиль линии.

Возвращаемое значение:

указатель на отрезок	- в случае удачного завершения,
0	- в случае неудачи.

Point – Проставить точку

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1391_Postroenie_tochek.htm#simple_point

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPoint

Синтаксис:

reference Point (double x, double y,

unsigned int style);

Входные параметры:

x, y	- координаты точки,
style	- стиль отрисовки точек.

Возвращаемое значение:

указатель на графический объект Точка	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Объект создается в текущем слое текущего вида.

PointArraw – Проставить значок в графическом документе

Пример...

Справка системы КОМПАС: стрелка линии-выноски...

`KOMPAS.chm::/CM_LEADER.htm`

Тип значка...

`KOMPAS.chm::/CM_LEADER.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPointArraw

Синтаксис:

reference PointArraw (double x, double y,

double ang,

unsigned char term);

Входные параметры:

x, y	- координаты точки привязки значка,
ang	- угол поворота значка,
term	- тип отрисовки значка.

Аннотационные объекты

Функции данного радела обеспечивают построение аннотационных объектов.

AnnArcByPoint – Создать аннотационную дугу по точкам

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnArcByPoint.

Синтаксис:

reference AnnArcByPoint (double xc, double yc, double rad,

double x2, double y2,

short direction,

unsigned char term1, unsigned char term2,

unsigned short style);

Входные параметры:

xc, yc	- координаты центра дуги,
rad	- радиус дуги,
x1, y1	- координаты начальной точки дуги,
x2, y2	- координаты конечной точки дуги,
direction	- направление отрисовки дуги:
	1 - против часовой стрелки,
	-1 - по часовой стрелке,
term1, term2	- типы значков на концах дуги,
style	- стиль линии.

Возвращаемое значение:

указатель на аннотационную дугу	- в случае удачного завершения,
0	- в случае неудачи.

AnnLineSeg – Построить аннотационный отрезок

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnLineSeg.

Синтаксис:

reference AnnLineSeg (double x1, double y1,

double x2, double y2,

unsigned char term1, unsigned char term2,

unsigned short style);

Входные параметры:

x1, y1	- координаты первой точки отрезка,
x2, y2	- координаты второй точки отрезка,
term1, term2	- типы значков на концах линии,
style	- стиль линии.

Возвращаемое значение:

указатель на аннотационный отрезок	- в случае удачного завершения,
0	- в случае неудачи.

ksAnnCircle – Создать объект "Аннотационная окружность"

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnCircle.

Синтаксис:

reference ksAnnCircle (double xc,
double yc,
double rad,
unsigned short style);

Входные параметры:

xc, yc	- координаты центра окружности,
rad	- радиус окружности,
style	- стиль линии.

Возвращаемое значение:

указатель на аннотационную окружность	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро аннотационные объекты превращаются в геометрические.

ksAnnEllipse – Создать объект "Аннотационный эллипс"

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnEllipse

Синтаксис:

reference ksAnnEllipse (EllipseParam * par);

Входные параметры:

par	- указатель на структуру параметров эллипса EllipseParam,

Возвращаемое значение:

указатель на аннотационный эллипс	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnParEllipseArc – Создать объект "Аннотационная дуга эллипса"

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnParEllipseArc.

Синтаксис:

reference LIB_FUNC ksAnnParEllipseArc (EllipseArcParam1 * par,
unsigned char term1,
unsigned char term2);

Входные параметры:

par	- указатель на структуру параметров дуги эллипса EllipseArcParam1,
term1, term2	- типы значков на концах ломаной.

Возвращаемое значение:

указатель на аннотационную дугу эллипса	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро аннотационные объекты превращаются в геометрические.

ksAnnPoint – Создать объект "точка" с аннотационной точкой привязки

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnPolyline.

Синтаксис:

reference LIB_FUNC ksAnnPoint(double x,
double y,
unsigned short style);

Входные параметры:

x, y	- координаты точки,

style	- стиль отрисовки точки: 0 - точка, 1 - крестик, 2 - х-точка, 3 - квадрат, 4 - треугольник, 5 - окружность, 6 - звезда, 7 - перечеркнутый квадрат, 8 - утолщенный плюс.

Возвращаемое значение:

указатель на точку	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.
3. Координаты точки привязки аннотационные не зависят от масштаба вида.

ksAnnPolyline – Создать аннотационную ломаную линию

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnPolyline.

Синтаксис:

int ksAnnPolyline(unsigned short style,
unsigned char term1,
unsigned char term2);

Входные параметры:

style	- стиль линии,
term1, term2	- типы значков на концах ломаной.

Возвращаемое значение:

указатель на аннотационную ломаную линию	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. ksAnnPolyline - составной объект. Объекты Point, вводимые между операторами ksAnnPolyline и EndObj, принадлежат аннотационной полилинии. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnPolylineEx – Создать объект "Аннотационная ломаная линия" по структуре параметров

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnPolylineEx.

Синтаксис:

reference ksAnnPolylineEx(PolylineParamEx * par,
unsigned char term1,
unsigned char term2);

Входные параметры:

par	- указатель на структуру параметров ломаной PolylineParamEx,
term1, term2	- типы значков на концах ломаной.

Возвращаемое значение:

указатель на аннотационную ломаную	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.

ksAnnTextEx– Создать многострочный текст по структуре параметров с аннотационной точкой привязки

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAnnTextEx.

Синтаксис:

reference ksAnnTextEx(TextParam * txtParam,
int align);

Входные параметры:

txtParam	- параметры текста,
align	- выравнивание текста: -1 - установить выравнивание как у стиля текста, 0 - слева, 1 - по центру, 2 - справа, 3 - на всю ширину.

Возвращаемое значение:

указатель на текст	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Аннотационные объекты можно создавать только в составе макро. Допускается создание аннотационных объектов во временной группе для последующего добавления в макро.
2. При разрушении макро-аннотационные объекты превращаются в геометрические.
3. Координаты точки привязки не зависят от масштаба вида.

Кривые Безье, Nurbs, ломаные

Функции данного радела обеспечивают построение NURBS и кривых Безье.

Bezier – Создать кривую Безье

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_BEZIER.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksBezier.

Синтаксис:

int Bezier (int closed, unsigned int style);

Входные параметры:

closed	- признак замыкания сплайна: 0 - незамкнутый, 1 - замкнутый,
style	стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Характерные точки кривой Безье задаются функциями Point (если не надо задавать производные в узлах) или BezierPoint (при необходимости задания производных) до вызова EndObj.
2. Объект создается в текущем слое текущего вида.
3. В качестве стиля линии можно задавать номер системного или пользовательского стиля линии.
4. Функция EndObj возвращает указатель на созданный объект Кривая Безье.

_Bezier – Создать кривую Безье по массиву точек (узлов кривой Безье)

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BEZIER.htm

Синтаксис:

```
reference _Bezier (BezierPointParam *par,
  int CountPoints,
  int closed,
  unsigned int style);
```

Входные параметры:

par	- массив точек (узлов кривой Безье - BezierPointParam), определяющих кривую.
CountPoints	- количество точек в массиве,
closed	- признак замыкания сплайна: 0 - незамкнутый, 1 - замкнутый,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на кривую Безье	- в случае успешного завершения,
0	- в случае неудачи.

BezierPoint – Построить узел кривой Безье

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BEZIER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksBezierPoint.

Синтаксис:

int BezierPoint (BezierPointParam *par);

Входной параметр:

par	- указатель на структуру параметров узла кривой Безье....

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksAddPowerForm – Ввести параметр для построения NURBS кусочно-степенным способом

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAddPowerForm.

Синтаксис:

int ksAddPowerForm (double x, double y);

Входные параметры:

x, y	- параметры степенной функции (см. Примечание).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Кусочно-степенная форма состоит из последовательности стыкующихся полиномов:

P0 [0, 1), P1 [1, 2), ... Pk [k, k1),

где Pi - полином, [i, i1) - интервал определения,

Pi(t).x = a0.x a1.x * (t - i) a2.x * (t - i) * (t - i) ... an.x * (t - i)^n

Pi(t).y = a0.y a1.y * (t - i) a2.y * (t - i) * (t - i) ... an.y * (t - i)^n

Функция ksAddPowerForm должна вызываться последовательно n1 раз для пар параметров (a0.x, a0.y), (a1.x, a1.y), ... (an.x, an.y).

Затем вызывается функция ksCreatePowerArc, которая создает из переданных параметров дугу NURBS степени n 1 и присоединяет ее к существующей кривой NURBS.

ksCreatePowerArc – Построить дугу NURBS кусочно-степенным способом и присоединить ее к существующей кривой NURBS

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreatePowerArc.

Синтаксис:

int ksCreatePowerArc();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание :

Параметры для построения дуги определяются функцией ksAddPowerForm.

ksNurbsKnot – Создать узел NURBS-кривой

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_NURBS.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNurbsKnot.

Синтаксис:

int ksNurbsKnot (double knot);

Входной параметр:

knot	- узел кривой NURBS.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция предназначена для считывания форматов обмена, в частности, DXF.

Для создания NURBS-кривой массив узлов (узловой вектор) задавать не обязательно. Но если он задан, то должен подчиняться следующим правилам:

1. Узловой вектор не должен быть убывающим.
2. Количество узлов (knotCount) должно быть:

- для разомкнутого сплайна knotCount= degree + pointCount;
- для замкнутого сплайна knotCount= degree + pointCount + (degree -1);

где:

- degree - степень сплайна,
- pointCount - количество точек.

ksPolyline – Создать ломаную

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#POLYLINE

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPolyline.

Синтаксис:

int ksPolyline (unsigned short style);

Входной параметр:

style	стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Узлы определяются следующими далее функциями Point. Описание завершается функцией EndObj, возвращающей указатель на созданный объект. Объект создается в текущем слое текущего вида. В качестве стиля линии можно задавать номер предопределенного (системного) или пользовательского стиля линии.

_ksPolyline – Создать ломаную линию

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#POLYLINE

Синтаксис:

reference _ksPolyline (PolylineParam *par);

Выходной параметр:

par - указатель на структуру параметров ломаной PolylineParam...

Возвращаемое значение:

указатель на ломаную линию - в случае удачного завершения,
0 - в случае неудачи.

Описание:

1. Объект создается в текущем слое текущего вида. В стилях линии можно задавать номер предопределенного (системного) или пользовательского стиля линии.
2. В функции GetObjParam на указатель полилинии выдаются параметры полилинии в виде структуры PolylineParam (если значение par равно ALLPARAM), если значение par другое - выдается только стиль полилинии. В функции SetObjParam можно изменить параметры полилинии.

ksPolylineEx – Создать ломаную линию

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#POLYLINE

Синтаксис:

reference ksPolylineEx(PolylineParamEx * par);

Входной параметр:

par - указатель на структуру параметров ломаной PolylineParamEx...

Возвращаемое значение:

указатель на ломаную линию - в случае удачного завершения,
0 - в случае неудачи.

Nurbs – Создать NURBS

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/spline_postroenie.htm#SPLINE_NURBS

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNurbs.

Синтаксис:

int Nurbs (unsigned char degree,

unsigned char close,

unsigned int style);

Входные параметры:

degree	- порядок NURBS (степень полинома + 1), от 3 до 10,
close	- признак замыкания сплайна: FALSE - незамкнутый, TRUE - замкнутый,
style	стиль линии.

Системные стили линий...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Узлы кривой определяются следующими далее функциями Point или NurbsPoint. Описание кривой завершается функцией EndObj, возвращающей указатель на созданную кривую.
2. Объект создается в текущем слое текущего вида.
3. В качестве стиля линии можно задавать номер предопределенного (системного LStyles) или пользовательского стиля линии.

NurbsForConicCurve – Создать NURBS по характеристическим точкам конического сечения

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNurbsForConicCurve.

Синтаксис:

reference NurbsForConicCurve (double x[4],

double y[4],

unsigned short style);

Входные параметры:

x[4], y[4]	- массивы координат характеристических точек конического сечения; точки не должны совпадать, а касательные к кривой в этих точках не должны быть параллельными,
style	- стиль линии.

Системные стили линий...

Возвращаемое значение:

указатель на NURBS	- в случае удачного завершения,
0	- в случае неудачи.

NurbsPoint – Создать узел NURBS

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_NURBS.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNurbsPoint.

Синтаксис:

int NurbsPoint (NurbsPointParam *par);

Входной параметр:

par - указатель на структуру параметров узла NURBS....

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Узлы кривой определяются следующими далее функциями Point или NurbsPoint. Описание кривой завершается функцией EndObj, возвращающей указатель на созданную кривую.

TanLineAngCurve – Функция расчета касательной к кривой

Синтаксис:

```
void LIB_FUNC TanLineAngCurve( reference p,       // указатель на кривую,
                               double ang,        // угол касательной,
                               reference pointArr ); // массив точек касания.
```

Входные параметры:

p - указатель на кривую,
ang - угол касательной.

Выходные параметры:

pointArr - массив точек касания.

Составные объекты

Contour – Создать контур

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ASSEMBLYCONTOUR.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksContour.

Синтаксис:

int Contour (unsigned short style);

Входной параметр:

style	- стиль линии.

Примечание:

Все, определяемые далее до вызова функции EndObj, геометрические примитивы чертежа (отрезки, дуги, кривые) будут включены в контур. Контур должен быть непрерывным - начальная точка очередного объекта обязана совпадать с конечной точкой предыдущего. Полученный контур может быть использован для построения эквидистанты. Функция EndObj возвращает указатель на созданный макроэлемент.

EndObj – Завершить создание комплексного объекта

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_MAKEMACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEndObj

Синтаксис:

reference EndObj (void);

Возвращаемое значение:

указатель на завершенный объект (контур, макроэлемент и т.п.)	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Комплексным считается объект, определяемый несколькими функциями.

ksDuplicateBoundaries – Получить временную группу контуров, задающих границы штриховки или заливки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/164_20_1_Zadanie_granic_shtrikh.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDuplicateBoundaries.

Синтаксис:

reference ksDuplicateBoundaries (reference p);

Входной параметр:

p	- указатель на штриховку или заливку.

Возвращаемое значение:

указатель на временную группу контуров, задающих границы штриховки или заливки	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Функция устарела, рекомендуется вместо нее использовать функцию ksDuplicateBoundariesEx

ksDuplicateBoundariesEx – Получить копию границы штриховки или заливки во временной группе

Справка системы КОМПАС...

KOMPAS.chm::/164_20_1_Zadanie_granic_shtrikh.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDuplicateBoundaries.

Синтаксис:

reference ksDuplicateBoundariesEx (reference p, unsigned char systemCoord);

Входной параметр:

p	- указатель на штриховку или заливку.
systemCoord	1 - в системе координат листа, 0 - в системе координат текущего вида.

Возвращаемое значение:

указатель на временную группу контуров, задающих границы штриховки или заливки	- в случае удачного завершения,
0	- в случае неудачи.

Работа с макроэлементами и библиотечными макроэлементами

Функции работы с макроэлементами и библиотечными макроэлементами

Библиотечный макроэлемент - это макроэлемент, имеющий пользовательские параметры.

Пользовательские параметры могут быть записаны в макроэлемент при помощи функции ksSetMacroParam.

Также при помощи функции ksSetMacroParam может быть включен режим управления характерными точками макроэлемента (поддержка макроэлементом интерфейса ILibHPObject), отработка событий внешних воздействий на макроэлемент (поддержка макроэлементом интерфейса ILibExternalObject), выключен\включен режим редактирования макроэлемента по двойному щелчку мыши.

Получить пользовательские параметры можно при помощи функции GetMacroParam.

EditMacroMode – Получить режим работы функции библиотеки (создание нового или редактирование существующего макроэлемента)

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/Z_MACRO.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEditMacroMode.

Синтаксис:

reference EditMacroMode();

Возвращаемое значение:

указатель на редактируемый макроэлемент	- при редактировании макроэлемента,
0	- при создании макроэлемента.

EndObj – Завершить создание комплексного объекта

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/351_39_1_Sozdanie_novogo_makroe.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEndObj

Синтаксис:

reference EndObj (void);

Возвращаемое значение:

указатель на завершенный объект (контур, макроэлемент и т.п.)	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Комплексным считается объект, определяемый несколькими функциями.

GetMacroParam – Получить параметры макроэлемента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroParam.

Синтаксис:

int GetMacroParam (reference pMacro,

void * value,

unsigned int size);

Входные параметры:

pMacro	- указатель на макроэлемент.

Выходные параметры:

valu e	-указатель на область памяти размером size, в которую будут скопированы пользовательские данные из макроэлемента,
size	-размер области памяти с пользовательскими данными для макроэлемента.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Пользовательские данные могут быть сохранены в макроэлементе при помощи ksSetMacroParam.
2. При pMacro == 0 выдаются параметры макроэлемента, редактирование которого производится в данный момент (если таковой существует).

3. При необходимости значение параметра size можно получить при помощи функции GetMacroParamSize.

GetMacroParamSize – Получить размер памяти параметров указанного макроэлемента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroParamSize.

Синтаксис:

int GetMacroParamSize (reference ref);

Входной параметр:

ref - указатель на макроэлемент.

Возвращаемое значение:

размер памяти - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

При ref = 0 выдается размер памяти параметров макроэлемента, редактирование которого производится в данный момент (если таковой имеется).

GetMacroPlacement – Получить точку привязки и угол поворота макроэлемента

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroPlacement.

Синтаксис:

int GetMacroPlacement(reference macro,

double *x,

double *y,

double *angl);

Входные параметры:

macro - указатель на макроэлемент (0 - редактируемый макроэлемент),

Выходные параметры:

x, y - координаты точки привязки макроэлемента
angl - угол поворота макроэлемента от оси OX против часовой стрелки (в градусах)

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи,
-1 - если у макроэлемента нет СК (не вызывалась функция SetMacroPlacement.)

Смотрите также:

Функция SetMacroPlacement.

Примечание:

Функция устарела, рекомендуется вместо нее использовать функцию ksGetMacroPlacement.

ksAddObjectToMacro – Добавить объект, слой, вид или группу объектов в макроэлемент

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAddObjectToMacro.

Синтаксис:

int ksAddObjectToMacro (reference macro,reference obj);

Входные параметры:

macro - указатель на макроэлемент,
obj - указатель на добавляемый объект.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Слой и вид добавляются россыпью.
2. Добавляемые объекты и макроэлемент должны принадлежать текущему документу и одному виду.
3. Добавляемые объекты перестают быть самостоятельными.

ksGetMacroEditParam – Получить параметры редактирования указанного макроэлемента

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Синтаксис:

int ksGetMacroEditParam (reference ref,

char * fileName,

unsigned int fileNameSize,

char * libName,

unsigned int libNameSize,

int * number);

Входные параметры:

ref	- указатель на макроэлемент,
fileName	- указатель на буфер имени файла библиотеки,
fileNameSize	- размер буфера имени файла библиотеки,
libName	- указатель на буфер имени библиотеки,
libNameSize	- размер буфера имени библиотеки,
number	- указатель на буфер номера функции редактирования.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция возвращает для указанного макроэлемента параметры редактирования: имя файла прикладной библиотеки, имя библиотеки и номер функции, предназначенной для редактирования данного макроэлемента.
2. При ref = 0 выдаются параметры для макроэлемента, редактирование которого производится в данный момент (если таковой имеется).
3. При значении NULL параметров fileName, libName, number или, если значение fileNameSize, libNameSize меньше требуемой длины, параметры не возвращаются.
4. При использовании Unicode следует использовать функцию ksGetMacroEditParamW.

ksGetMacroEditParamW – Получить параметры редактирования указанного макроэлемента (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Синтаксис:

int LIB_FUNC ksGetMacroEditParamW (reference ref,

LPWSTR fileName,

unsigned int fileNameSize,

LPWSTR libName,

unsigned int libNameSize,

int * number);

Входные параметры:

ref	- указатель на макроэлемент,
fileName	- указатель на буфер имени файла библиотеки,
fileNameSize	- размер буфера имени файла библиотеки,
libName	- указатель на буфер имени библиотеки,
libNameSize	- размер буфера имени библиотеки,
number	- указатель на буфер номера функции редактирования.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция возвращает для указанного макроэлемента параметры редактирования: имя файла прикладной библиотеки, имя библиотеки и номер функции, предназначенной для редактирования данного макроэлемента.
2. При ref = 0 выдаются параметры для макроэлемента, редактирование которого производится в данный момент (если таковой имеется).
3. При значении NULL параметров fileName, libName, number или если значение fileNameSize, libNameSize меньше требуемой длины - параметры не возвращаются.
4. При использовании ANSI следует использовать функцию ksGetMacroEditParam.

ksGetMacroPlacement – Получить точку привязки и угол поворота макроэлемента

Справка системы КОМПАС...

KOMPAS.chm::/619_75_2_Vstavka_fragmentov_iz_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroPlacement.

Синтаксис:

int ksGetMacroPlacement (reference macro,

double *x,

double *y,

double *angl);

unsigned char sheetParam);

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
sheetParam	- признак системы координат: 1 - координаты и угол заданы в СК макроэлемента, 0 - в СК вида.

Выходные параметры:

x, y	- координаты точки привязки макроэлемента,
angl	- угол поворота макроэлемента от оси OX против часовой стрелки (в градусах).

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи,
-1	- если у макроэлемента нет СК (не вызывалась функция SetMacroPlacement),

Смотрите также:

Функция SetMacroPlacement.

ksGetMacroPlacementEx – Получить точку привязки и угол поворота макроэлемента

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroPlacementEx.

Описание:

Развитие функции ksGetMacroPlacement. В дополнение можно получить флаг зеркальной симметрии объекта.

mirrorSymmetry = 0 - нормальный исходный объект,

mirrorSymmetry = 1 - макроэлемент получен операцией симметрии из исходного,

mirrorSymmetry = NULL - не заполняется.

Синтаксис:

int ksGetMacroPlacementEx(reference macro,
double *x,
double *y,
double *angl,
unsigned char sheetParam,
unsigned char * mirrorSymmetry);//флаг зеркальной симметрии объекта

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
sheetParam	- признак системы координат: 1 - координаты и угол заданы в СК листа, 0 - в СК вида.

Выходные параметры:

x, y	- координаты точки привязки макроэлемента,
angl	- угол поворота макроэлемента от оси OX против часовой стрелки (в градусах),
mirrorSymmetry	- флаг зеркальной симметрии объекта; 0 - нормальный исходный объект, 1 - макроэлемент получен операцией симметрии из исходного.

Возвращаемое значение:

1	- в случае успеха,
0	- ошибка выполнения функции (у макрообъекта нет СК).

Смотрите также:

1. Функции GetMacroPlacement, ksGetMacroPlacement.
2. Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена, относительно исходного ее построения..

Примечание:

Функция позволяет получить СК макрообъекта в СК вида или листа, угол проворота, флаг зеркальной симметрии объекта. Если СК макро не имеет, функция вернет 0.

Если macro = 0, функция позволяет получить СК редактируемого макроэлемента.

ksGetMacroWaitDblClickEdit – Получить режим ожидания DblClick при редактировании макроэлемента

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetMacroWaitDblClickEdit.

Синтаксис:

long ksGetMacroWaitDblClickEdit (reference ref);

Входные параметры:

ref	- указатель на макроэлемент.

Возвращаемое значение:

1	- режим ожидания DblClick включен,
0	- режим ожидания DblClick выключен.

Примечание:

Макроэлемент, созданный библиотекой, может редактироваться двумя способами:

- по первому щелчку мыши - через характерные точки и\или интерфейс внешних воздействий;
- по двойному щелчку - через команду библиотеки.

Если время инициализации характерных точек сравнимо с двойным щелчком, то редактирование через команду библиотеки становится невозможно. Во время редактирования макроэлемента, при включенном режиме ожидания DblClick, после первого щелчка по макроэлементу КОМПАС будет ожидать повторного нажатия, в течении времени, установленного в настройках "Скорость выполнения двойного щелчка".

При выключенном режиме ожидания DblClick пауза отсутствует и нажатие всегда интерпретируется как одинарное нажатие.

ksOpenMacro – Открыть макроэлемент для редактирования

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenMacro.

Синтаксис:

int ksOpenMacro(reference macro);

Входной параметр:

macro	- указатель на макроэлемент.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. После вызова функции можно добавлять в macro новые объекты вида, использовать функцию FindObj; режим закрывается функцией EndObj();
2. Открыт доступ к внутренним объектам макроэлементов. Теперь при создании макроэлемента или при добавлении к макроэлементу новых объектов вида, на внутренние объекты выдается указатель (reference), как и на самостоятельные объекты вида.

   Пример....
3. Допускается вводить пустой макроэлемент. Пустой макроэлемент является вырожденным объектом. Он живет только в период редактирования документа. После записи документа, в котором находился пустой макро, этот макро будет удален. Макроэлемент считается нормальным, если в нем находится хотя бы один внутренний объект.

Пример…

4. Функции работы с указателями на объекты допускают работу с внутренними объектами макроэлемента. К ним относятся:

- CopyObj
- DecomposeObj
- DeleteObj
- GetObjGabaritRect
- GetObjParam
- ksEditViewObject
- LightObj
- MoveObj
- RotateObj
- SetObjParam
- SymmetryObj
- TransformObj

ksSetMacroParam – Записать параметры макроэлемента

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroParam.

Синтаксис:

```
int ksSetMacroParam( reference m,
void * value,
unsigned int size,
char * fileName,
char * libName,
int number,
unsigned char paramType );
```

Входные параметры:

m	- указатель на макроэлемент,
value	- указатель на область памяти размером size, содержащую пользовательские данные для макроэлемента (см. описание ниже),
size	- размер области памяти с пользовательскими данными для макроэлемента;
fileName	- имя файла библиотеки;
libName	- имя библиотеки;
number	- номер функции в библиотеке,
paramType	- битовый флаг, указание какой тип редактирования поддерживает макроэлемент.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Описание:

Запись в макроэлемент параметров, создающей его библиотечной функции, для дальнейшего редактирования. Если в сеансе работы системы КОМПАС-ГРАФИК такой макроэлемент указывается двойным щелчком левой кнопки мыши, то автоматически будет вызвана библиотечная функция для его редактирования. Она не обязательно должна совпадать с родительской функцией.

При значении имен библиотеки и ее файла NULL и номера функции -1 редактирование будет осуществляться родительской функцией.

При обработке событий это правило не действует, и в макроэлемент прописываются имя той библиотеки и ее файла, которая была использована непосредственно перед этим событием. Поэтому при создании макроэлемента по событию необходимо явно задавать имя библиотеки, имя файла и номер функции.

При необходимости в параметрах макроэлемента можно дополнительно сохранить пользовательские данные (допустимы все типы данных, кроме указателей), задав указатель на область памяти, в которой эти данные хранятся и размер этой области памяти (параметры value и size).

Примечание.

1. При использовании Unicode следует использовать функцию ksSetMacroParamW.
2. Параметры макроэлемента можно получить, используя функцию GetMacroParam.

ksSetMacroParamW – Записать параметры макроэлемента (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroParam.

Синтаксис:

```
extern "C" int LIB_FUNC ksSetMacroParamW( reference m,
void * value,
unsigned int size,
LPWSTR fileName,
LPWSTR libName,
int number,
unsigned char paramType );
```

Входные параметры:

m - указатель на макроэлемент,

value - указатель на структуру, содержащую сохраняемые параметры;
size - размер структуры параметров редактирования;
fileName - имя файла библиотеки;
libName - имя библиотеки;
number - номер функции в библиотеке,
paramType - битовый флаг, указание какой тип редактирования поддерживает макро.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Описание:

Запись в макроэлемент параметров, создающей его библиотечной функции, для дальнейшего редактирования. Если в сеансе работы системы КОМПАС-ГРАФИК такой макроэлемент указывается двойным щелчком левой кнопки мыши, то автоматически будет вызвана библиотечная функция для его редактирования. Она не обязательно должна совпадать с родительской функцией. При значении имен библиотеки и ее файла NULL и номера функции -1 редактирование будет осуществляться родительской функцией.

При обработке событий это правило не действует, и в макроэлемент прописываются имя той библиотеки и ее файла, которая была использована непосредственно перед этим событием. Поэтому при создании макроэлемента по событию необходимо явно задавать имя библиотеки, имя файла и номер функции.

Примечание.

1. При использовании ANSI следует использовать функцию ksSetMacroParam.
2. Можно сохранить в макроэлементе дополнительные параметры, распределив блок памяти userPars и заполнив его нужными значениями (допустимы все типы данных, кроме указателей).
3. Параметры макроэлемента можно получить, используя функцию GetMacroParam.

ksSetMacroPlacementEx – Установить точку привязки и угол поворота макроэлемента

Справка системы КОМПАС...

`KOMPAS.chm::/Z_MACRO.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroPlacementEx.

Синтаксис:

```
int  ksSetMacroPlacementEx( reference macro,
double x,
double y,
double angl,
int relative,
unsigned char mirrorSymmetry );
```

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
x, y	- точка начала координат макроэлемента,
angl	- угол поворота оси X,
relative	- признак системы координат: 1 - координаты и угол заданы в СК макроэлемента, 0 - в СК вида,
mirrorSymmetry	- флаг зеркальной симметрии объекта.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Смотрите также:

1. Функции GetMacroPlacement, ksGetMacroPlacement.
2. Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена относительно исходного ее построения.

Примечание:

Функция позволяет установить СК макроэлемента либо в СК вида, либо смещением относительно СК макроэлемента.

ksSetMacroWaitDblClickEdit – Изменить режим ожидания DblClick при редактировании макроэлемента

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroWaitDblClickEdit.

Синтаксис:

long ksSetMacroWaitDblClickEdit (reference ref, long waitDblClick);

Входные параметры:

ref	- указатель на макроэлемент,
waitDblClick	- режим ожидания DblClick (1 - включить, 0 - выключить).

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Макроэлемент, созданный библиотекой, может редактироваться двумя способами:

- по первому щелчку мыши - через характерные точки и\или интерфейс внешних воздействий;
- по двойному щелчку - через команду библиотеки.

Если время инициализации характерных точек сравнимо с двойным щелчком, то редактирование через команду библиотеки становится невозможно. Во время редактирования макроэлемента при включенном режиме ожидания DblClick после первого щелчка по макроэлементу КОМПАС будет ожидать повторного нажатия, в течении времени, установленного в настройках "Скорость выполнения двойного щелчка".

При выключенном режиме ожидания DblClick пауза отсутствует и нажатие всегда интерпретируется как одинарное нажатие.

ksUpdateMacro – Очистить макроэлемент и положить в него группу

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksUpdateMacro.

Синтаксис:

int ksUpdateMacro (reference macro, reference gr);

Входные параметры:

macro	- указатель на макроэлемент,
gr	- указатель на группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Macro – Создать новый макроэлемент

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMacro.

Синтаксис:

int Macro (unsigned char type);

Входной параметр:

type	- тип макроэлемента: 0 - объединяет объекты текущего слоя, 1- включаемые объекты могут принадлежать разным слоям.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Макроэлемент - составной объект.
2. Объекты вида, вводимые между методами ksDocument2D::ksMacro и EndObj, принадлежат макроэлементу. EndObj возвращает указатель на макроэлемент.
3. Допускается вложенность макроэлементов друг в друга (максимальное число вложений не ограничено).

SetMacroParam – Установить параметры макроэлемента

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/Z_MACRO.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroParam.

Синтаксис:

int SetMacroParam (reference m,

void * value,

unsigned int size,

char *fileName,

char * libName ,

int number);

Входные параметры:

m	- указатель на макроэлемент,
value	- указатель на структуру, содержащую сохраняемые параметры;
size	- размер структуры параметров редактирования;
fileName	- имя файла библиотеки;
libName	- имя библиотеки;
number	- номер функции в библиотеке.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Описание:

1. Запись в макроэлемент параметров, создающей его библиотечной функции, для дальнейшего редактирования. Если в сеансе работы системы КОМПАС-ГРАФИК такой макроэлемент указывается двойным щелчком левой кнопки мыши, то автоматически будет вызвана библиотечная функция для его редактирования. Она не обязательно должна совпадать с родительской функцией. При значении имен библиотеки и ее файла NULL и номера функции -1 редактирование будет осуществляться родительской функцией.
2. Можно сохранить в макроэлементе дополнительные параметры, распределив блок памяти userPars и заполнив его нужными значениями (допустимы все типы данных, кроме указателей).
3. Параметры макроэлемента можно получить, используя функцию GetMacroParam.

Примечание:

Функция устарела, рекомендуется вместо нее использовать функцию ksSetMacroParam .

SetMacroPlacement – Установить точку привязки и угол поворота макроэлемента

Справка системы КОМПАС..

`KOMPAS.chm::/Z_MACRO.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMacroPlacement.

Синтаксис:

int SetMacroPlacement(reference macro,

double x,

double y,

double angl,

int relativ);

Входные параметры:

macro	- указатель на макроэлемент (0 - редактируемый макроэлемент),
x, y	- координаты точки привязки макроэлемента,
angl	- угол поворота макроэлемента от оси OX против часовой стрелки (в градусах),
relativ	- признак системы координат: 1 - координаты и угол заданы в СК макроэлемента, 0 - в СК вида.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Смотрите также:

- Функции GetMacroPlacement, ksGetMacroPlacement, ksSetMacroPlacementEx

▼ Рекомендации по обеспечению корректного редактирования библиотекой макроэлементов, геометрия которых зеркально отражена, относительно исходного ее построения.

Внешние воздействия на библиотечный элемент

Интерфейс ILibExternalObject

Интерфейс внешних воздействий на библиотечный элемент 2D.

Иерархия:

IUnknown

ILibExternalObject

Описание.

Интерфейс должен быть реализован на стороне библиотеки разработчиком приложения, если необходимо отработать события, возникающие при редактировании библиотечного элемента 2D.

Режим поддержки интерфейса управления характерными точками включается в функции ksSetMacroParam.

Вызывается КОМПАС при редактировании библиотечного элемента 2D через функцию LibObjInterfaceEntry.

Является обработчиком событий, генерирующихся КОМПАС при редактировании библиотечного элемента 2D.

Интерфейс позволяет обработать события сдвига, поворота, деформации, удаления, восстановления из Undo для библиотечного элемента 2D, а также добавление элемента в модель.

Интерфейс реализован еще до реализации механизма событий в КОМПАС API.

Практически полностью дублирует события интерфейса IObject2DNotify.

Отличие состоит в том, что передача интерфейса КОМПАС осуществляется не через подписку а посредством предопределенной функции LibObjInterfaceEntry. Передача интерфейса КОМПАС может быть осуществлена из библиотеки, установленной в функции ksSetMacroParam, в то время как подписка на интерфейс IObject2DNotify может быть осуществлена с любой библиотеки.

ILibExternalObject – методы

Lib_AddToModel – Библиотечный элемент 2D добавлен в модель

Интерфейс...

Синтаксис:

BOOL Lib_AddToModel();

Возвращаемое значение:

TRUE - в случае успешного завершения,

FALSE - в случае неудачи.

Примечания:

Этот метод позволяет отработать событие добавления библиотечного элемента в модель.

Lib_Deform – Деформация библиотечного элемента 2D

Интерфейс...

Синтаксис:

BOOL Lib_Deform();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Этот метод позволяет отработать событие деформации библиотечного элемента.

Lib_Delete – Удаление библиотечного элемента 2D

Интерфейс...

Синтаксис:

BOOL Lib_Delete();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Этот метод позволяет отработать событие удаления библиотечного элемента.

Lib_Move – Сдвиг библиотечного элемента 2D

Интерфейс...

Синтаксис:

BOOL Lib_Move();

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечания:

Этот метод позволяет отработать событие сдвига библиотечного элемента.

Lib_Restore – Восстановление библиотечного элемента 2D из Undo

Интерфейс...

Синтаксис:

BOOL Lib_Restore();

Выходной параметр:

text	- текст для отображения рядом с курсором.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Этот метод позволяет отработать событие восстановления библиотечного элемента.

Lib_Rotate – Поворот библиотечного элемента 2D

Интерфейс...

Синтаксис:

BOOL Lib_Rotate();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Этот метод позволяет отработать событие поворота библиотечного элемента.

Lib_Transform – Трансформация библиотечного элемента 2D

Интерфейс...

Синтаксис:

BOOL Lib_Transform();

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечания:

Этот метод позволяет отработать событие трансформации библиотечного элемента.

Интерфейс ILibPropertyObject

Интерфейс объекта со свойствами отображаемыми в окне свойств.

Описание:

Интерфейс должен быть реализован на стороне библиотеки разработчиком приложения, если необходимо обеспечить управление характерными точками библиотечного элемента 2D.

Режим поддержки интерфейса управления характерными точками включается в функции ksSetMacroParam.

Вызывается КОМПАС при селектировании библиотечного элемента 2d через функцию LibObjInterfaceEntry.

Интерфейс является обработчиком событий, генерирующихся КОМПАС при селектировании библиотечного элемента.

Интерфейс позволяет выполнить следующие действия:

- Добавить дополнительные свойства в окно свойств для библиотечных элементов.
- Выполнить изменение свойств через окно свойств.
- Выполнить копирование свойств в процессе копирования свойств между библиотечными элементами.
- Выполнить поиск и селектирование объектов по свойствам.

Интерфейс реализован по аналогии с интерфейсом хот-точек.

Передача интерфейса в КОМПАС осуществляется посредством предопределенной функции LibObjInterfaceEntry.

Передача интерфейса КОМПАС может быть осуществлена из библиотеки, установленной флагом MP_PROPERTY_OBJECT в функции ksSetMacroParam, или установившей свойство IMacroObject::PropertyObjectEditable.

Группировка и сортировка свойств осуществляется по идентификатору. Свойства могут быть заданы в любой последовательности. Они будут отсортированы по возрастанию идентификатора. Принадлежность к группе определяется также по идентификатору.

Свойство добавится в группу с ближайшим меньшим идентификатором.

Применение интерфейса приведено в следующих примерах:

Gayka1 (Демонстрационные примеры КОМПАС-МАСТЕР);

Пример описания ресурсов...

ILibPropertyObject – методы

ApplyProperty – Установить новое значение для свойства

Интерфейс...

Синтаксис COM:

int ApplyProperty(PropertyParam * param);

Входные и выходные параметры:

param - структура параметров свойства PropertyParam,

Возвращаемое значение:

1 - новое значение свойства установлено,
0 - новое значение не установлено.

Примечание:

Идентификация свойства осуществляется по параметру propertyId в PropertyParam.

EndEditProperty – Завершение редактирования свойств объекта

Интерфейс...

Синтаксис COM:

BOOL EndEditProperty( );

Возвращаемое значение:

Не используется.

Примечание:

Предназначено для освобождения каких либо данных, созданных на время редактирования свойств.

GetGroupName – Имя группы объектов

Интерфейс...

Синтаксис COM:

LPOLESTR GetGroupName();

Возвращаемое значение:

- имя группы объектов, отображаемой в списке типов выделенных объектов окна свойств.

Примечание:

Группы объектов также отображаются в окне поиска объектов по свойствам.

GetMouseEnterLeavePoint – Запрос параметров точек для визуального определения места применения параметра

Интерфейс...

Синтаксис:

BOOL GetMouseEnterLeavePoint(int propertyId, long pointIndex, LPUNKNOWN parameters);

Входные параметры:

propertyId	- идентификатор свойства,
pointIndex	- индекс точки,
parameters	- интерфейс параметров отображения точки IMouseEnterLeaveParameters.

Примечание:

Функция вызывается при наведении на редактор свойства, если установлен признак PropertyParam .needMouseEnterLeaveMessage.

Функция позволяет подсветить в чертеже одну или несколько точек применения контрола.

Функция вызывается в цикле с увеличением индекса точки pointIndex пока библиотека не вернет FALSE, как признак завершения получения параметров точек.

GetProperty – Добавить свойство в список свойств

Интерфейс...

Синтаксис COM:

BOOL GetProperty(int index, PropertyParam * param);

Входной параметр:

index	- индекс свойства,

Выходной параметр:

param	- структура параметров свойства PropertyParam,

Возвращаемое значение:

TRUE	- если требуется добавить свойство с заданным индексом,
FALSE	- формирование списка свойств закончено.

Примечание:

1. Метод позволяет сформировать список внешних свойств объекта.
2. Метод вызывается при первом селектировании объекта.
3. Для получения текущих значений свойства используется метод ILibPropertyObject::UpdateProperty.

4. В отличие от работы с окном свойств, для организации полноценной работы поиска по свойствам библиотечных макроэлементов необходимо:

- Задать для вещественных и целых значений минимальное и максимальное значение свойства (для создания правильного валидатора).
- Заполнить список всеми возможными значениями свойства без учета зависимостей от других свойств (т.к. в диалоге поиска это список всегда должен показываться полностью). Это требуется для полноценной работы динамически задаваемых списочных свойств (PropertyParam::additionData).

OnChoiceProperty – Событие запуска внешнего редактирования для пользовательского свойства

Интерфейс...

Синтаксис COM:

BOOL OnChoiceProperty(PropertyParam * param);

Входные и выходные параметры:

param - структура параметров свойства PropertyParam,

Возвращаемое значение:

TRUE - значение изменилось,
FALSE - значение не изменилось.

Примечание:

1. Идентификация свойства осуществляется по параметру propertyId в PropertyParam.
2. Событие вызывается для свойств с типом ksOPControlExternalEdit и ksOPControlExternalStringEdit из перечисления ksObjectPropertyControlTypeEnum.

UpdateProperty – Обновить параметры свойства

Интерфейс...

Синтаксис COM:

BOOL UpdateProperty(PropertyParam * param);

Входные и выходные параметры:

param - структура параметров свойства PropertyParam,

Возвращаемое значение:

TRUE - если параметры свойства обновились,
FALSE - если параметры свойства не изменились.

Примечание:

1. Идентификация свойства осуществляется по параметру propertyId в PropertyParam.
2. При выделении нескольких макроэлементов отображаемый в окне "Свойства" список значений будет сформирован с учетом значения флага PropertyParam:

   summList - // TRUE - объединять списки, FALSE - пересекать списки.

Работа с характерными точками

Интерфейс ILibHPObject1

Интерфейс для работы с характерными точками.

Иерархия:

IKompasAPIObject

    ILibHPObject1

Описание:

Дополнительный интерфейс управления характерными точками библиотечного элемента 2d. Является обработчиком событий, которые генерируются системой КОМПАС при селектировании библиотечного элемента 2d.

Интерфейс должен быть реализован на стороне библиотеки разработчиком приложения в том же классе, который реализует интерфейс ILibHPObject, если необходимо обеспечить управление характерными точками библиотечного элемента.

Данный интерфейс можно получить у объекта ILibHPObject посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif).

Интерфейс позволяет выполнить следующие действия.

- Сформировать отображение характерных точек, задав идентификаторы битмапов.
- Задать тексты характерных точек в расширенном представлении.
- Изменить отображение характерной точки и текст у курсора при попадании указателя мыши в габарит точки.
- Получить выпадающее меню, ассоциированное с данным библиотечным элементом или с данной hot-точкой.
- Установить состояния команд из библиотечного меню.
- Отработать события селектирования\расселектирования характерных точек.

ILibHPObject1 – методы

LibHotPnt_GetEx – Получить текущее описание характерной точки

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_GetEx(HotPointDescription1* point, int index);

Входной параметр:

index - номер характерной точки.

Выходной параметр:

point - структура параметров характерной точки HotPointDescription1.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Примечания:

1. Метод вызывается в цикле для каждой точки (увеличивая номер точки), пока не вернет FALSE.
2. Значения координат характерной точки необходимо устанавливать в собственной системе координат объекта.

LibHotPnt_GetCursorTextEx – Задать текст для отображения рядом с курсором

Интерфейс...

Синтаксис COM:

LPOLESTR LibHotPnt_GetCursorTextEx(int index);

Входной параметр:

index - номер характерной точки.

Выходной параметр:

text - текст для отображения рядом с курсором.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Метод вызывается при попадании мыши в габарит hot point. При помощи метода можно указать текст, который будет выдан рядом с курсором. Текст выдается для UNICODE вместо ILibHPObject::LibHotPnt_GetCursorText.
2. С присланной строки снимается копия, сама исходная строка не удаляется.

LibHotPnt_GetMenuEx – Получить popup-меню, ассоциированное с данным библиотечным элементом и ассоциированное с данной hot-точкой, если index > -1

Интерфейс...

Синтаксис COM:

int LibHotPnt_GetMenuEx(int index);

Входной параметр:

index - номер характерной точки.

Возвращаемое значение:

Идентификатор меню (HMENU) или 0, если меню не определено. - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. Метод будет вызван при нажатии правой кнопки мыши на выделенном библиотечном элементе или его характерной точке.
2. Метод позволяет получить выпадающее меню, ассоциированное с данным библиотечным элементом, если index = -1 и ассоциированное с данной hot-точкой, если index > -1.
3. Присланное меню будет модифицировано в системе.
4. Идентификаторы команд должны быть уникальны на уровне приложения.
5. При модификации меню идентификаторы команд становятся уникальными на уровне КОМПАС.
6. Меню будет добавлено в конец системного выпадающего меню для библиотечного элемента.
7. При выборе команды из меню будет вызван метод ILibHPObject::LibHotPnt_ExecuteCommand.
8. Метод является расширением возможностей метода ILibHPObject::LibHotPnt_GetMenu.
9. Присланное меню будет уничтожено вызовом функции DestroyMenu().

LibHotPnt_Select – Выделить характерную точку

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_Select(int index);

Входной параметр:

index - номер характерной точки.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Выделение характерной точки происходит при нажатии левой кнопки мыши на характерной точке. При этом генерируется событие.

Этот метод является обработчиком события выделения характерной точки библиотечного элемента 2d.

Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

LibHotPnt_Unselect – Отменить выделение характерной точки

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_Unselect(int index);

Входной параметр:

index - номер характерной точки.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Метод вызывается при снятии выделения характерной точки. Метод является обработчиком события снятия выделения (расселектирования) характерной точки библиотечного элемента 2d. Это событие происходит при нажатии левой кнопки мыши вне выделенной характерной точки или после окончания перетаскивания характерной точки с нажатой левой кнопкой мыши.

Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

LibHotPnt_UpdateCommand – Установить состояния команд из меню

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_UpdateCommand(int commandId);

Входной параметр:

commandId - идентификатор команды.

Возвращаемое значение:

TRUE - команда доступна,
FALSE - команда недоступна.

Примечание:

Метод позволяет установить состояния команд из меню, присланного по ILibHPObject::LibHotPnt_GetMenu или ILibHPObject1::LibHotPnt_GetMenuEx.

Интерфейс ILibHPObject

Интерфейс управления характерными точками библиотечного объекта 2D.

Иерархия:

IUnknown

ILibHPObject

ILibHPObject1

Описание.

Посредством вызова метода IUnknown::QueryInterface (const GUID far& IID, void** pif) у данного интерфейса можно получить дополнительный интерфейс ILibHPObject1 - интерфейс для работы с характерными точками.

Интерфейс должен быть реализован на стороне библиотеки разработчиком приложения, если необходимо обеспечить управление характерными точками библиотечного элемента 2D.

Режим поддержки интерфейса управления характерными точками включается в функции ksSetMacroParam.

Вызывается КОМПАС при селектировании библиотечного элемента 2d через функцию LibObjInterfaceEntry.

Является обработчиком событий, генерирующихся КОМПАС при селектировании библиотечного элемента.

Интерфейс позволяет выполнить следующие действия.

- Сформировать отображение характерных точек, задав координаты характерных точек.
- Задать тексты характерных точек.
- Задав идентификаторы курсора, изменить отображение курсора и его текст при попадании указателя мыши в габарит точки.
- Получить выпадающее меню, ассоциированное с данным библиотечным элементом.
- Отработать события начала смещения, смещения и конца смещения характерных точек.
- Выполнить команду библиотечного меню.

Интерфейс реализован еще до реализации механизма событий в КОМПАС API.

Отличие состоит в том, что передача интерфейса КОМПАС осуществляется не через подписку а посредством предопределенной функции LibObjInterfaceEntry. Передача интерфейса КОМПАС может быть осуществлена из библиотеки, установленной в функции ksSetMacroParam.

ILibHPObject– методы

LibHotPnt_Complete – Завершить редактирование методом перетаскивания характерных точек

Интерфейс...

Синтаксис:

BOOL LibHotPnt_Complete(int index, BOOL success);

Входные параметры:

index	- номер характерной точки,
success	- признак завершения сдвига характерной точки: TRUE - нормальное завершение, FALSE - во время сдвига нажата <Esc>.

Возвращаемое значение:

TRUE	- в случае удачного завершения.

Описание:

Этот метод позволяет отработать событие окончания перетаскивания характерной точки с нажатой левой кнопкой мыши.

Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

LibHotPnt_ExecuteCommand – Выполнить команду из меню, присланного по LibHotPnt_GetMenu()

Интерфейс...

Синтаксис:

BOOL LibHotPnt_ExecuteCommand(int id);

Входной параметр:

id	- идентификатор команды контекстного меню.

Возвращаемое значение:

TRUE	- что-то изменилось и требуется передача в UNDO и пересчет положения характерных точек,
FALSE	- команда не обработана.

Описание:

Этот метод позволяет отработать событие выбора команды контекстного меню, полученного методами ILibHPObject::LibHotPnt_GetMenu и ILibHPObject1::LibHotPnt_GetMenuEx, и выполнить команду с присланным идентификатором.

Выбор команды контекстного меню осуществляется нажатием левой кнопки мыши в габарите команды меню.

Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

LibHotPnt_Get – Установить текущее описание характерной точки

Интерфейс...

Синтаксис:

BOOL LibHotPnt_Get(HotPointDescription* point, int index);

Входной параметр:

index - номер характерной точки,

Выходной параметр:

point - структура параметров характерной точки HotPointDescription,

Возвращаемое значение:

TRUE - в случае удачного завершения.

Описание:

Метод позволяет задать отображение характерных точек, задав координаты характерных точек, тексты характерных точек, идентификаторы курсора, благодаря чему, можно изменить отображение курсора при попадании указателя мыши в габарит точки.

Примечания:

1. Метод вызывается в цикле для каждой точки (увеличивая номер точки), пока не вернет FALSE.
2. Значения координат характерной точки необходимо устанавливать в системе координат макроэлемента.

   Координаты характерных точек определены в системе координат макроэлемента.

LibHotPnt_GetCursorText – Задать текст для отображения рядом с курсором

Интерфейс...

Синтаксис:

BOOL LibHotPnt_GetCursorText(int index,

char** text);

Входной параметр:

index - номер характерной точки.

Выходной параметр:

text - текст для отображения рядом с курсором.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. При движении мыши над характерной точкой, можно задать текст у курсора. Метод вызывается при попадании мыши в габарит hot point.
2. С присланной строки снимается копия, сама исходная строка не удаляется.

LibHotPnt_GetMenu – Задать контекстное меню, ассоциированное с данным библиотечным элементом

Интерфейс...

Синтаксис:

int LibHotPnt_GetMenu();

Возвращаемое значение:

контекстное меню (HMENU).

Описание:

Метод вызовется при нажатии правой кнопки мыши на селектированном библиотечном элементе. Метод позволяет получить выпадающее меню, ассоциированное с данным библиотечным элементом. Присланное меню будет модифицировано в системе.

Идентификаторы команд должны быть уникальны на уровне приложения.

При модификации меню идентификаторы команд становятся уникальными на уровне КОМПАС. Меню будет добавлено в конец системного выпадающего меню для библиотечного элемента.

При выборе команды из меню будет вызван метод ILibHPObject::LibHotPnt_ExecuteCommand.

Присланное меню будет уничтожено вызовом функции DestroyMenu().

LibHotPnt_GetMenuEx – Получить popup-меню, ассоциированное с данным библиотечным элементом и с данной hot-точкой, если index > -1

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_ViewData(HotPointDescription1 * point, int index);

Входной параметр:

index - номер характерной точки.

Выходной параметр:

point - указатель структуры параметров характерной точки HotPointDescription1.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. При движении мыши над характерной точкой можно задать новое изображение (подсветку) точки и выдать текст у курсора.
2. Метод вызывается при попадании мыши в габарит hot point.
3. При помощи метода можно указать текст, который будет выдан рядом с курсором. Текст выдается для UNICODE вместо ILibHPObject::LibHotPnt_GetCursorText.
4. С присланной строки снимается копия, сама исходная строка не удаляется.
5. Идентификатор нового битмапа характерной точки. Если идентификатор битмапа 0, битмап не меняется.

LibHotPnt_Prepare – Подготовиться к редактированию методом перетаскивания характерных точек

Интерфейс...

Синтаксис:

BOOL LibHotPnt_Prepare(int index);

Входной параметр:

index - номер характерной точки.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Описание:

Этот метод позволяет отработать событие начала перетаскивания характерной точки с нажатой левой кнопкой мыши.

Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

LibHotPnt_Set – Получить положение характерной точки

Интерфейс...

Синтаксис:

BOOL LibHotPnt_Set(HotPointDescription* point, int index);

Входные параметры:

index - номер характерной точки,
point - структура параметров характерной точки HotPointDescription.

Возвращаемое значение:

TRUE - в случае удачного завершения.

Описание:

При перетаскивании характерной точки библиотечного элемента 2D с нажатой левой кнопкой мыши генерируется событие.

Этот метод позволяет отработать событие изменения положения характерной точки. Библиотека может изменить отображение библиотечного элемента или выполнить другие действия, связанные с этим событием.

Примечания:

1. Актуальны только координаты (x, y), остальные поля должны игнорироваться.
2. Значения координат приходят в собственной системе координат объекта.

LibHotPnt_ViewData – Получить описание характерной точки при попадании указателя мыши в габарит точки

Интерфейс...

Синтаксис COM:

BOOL LibHotPnt_ViewData(HotPointDescription1 * point, int index);

Входной параметр:

index - номер характерной точки.

Выходной параметр:

point - указатель структуры параметров характерной точки HotPointDescription1.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

Примечание:

1. При движении мыши над характерной точкой можно задать новое изображение (подсветку) точки и выдать текст у курсора.
2. Метод вызывается при попадании мыши в габарит hot point.
3. При помощи метода можно указать текст, который будет выдан рядом с курсором.
4. Текст выдается для UNICODE вместо ILibHPObject::LibHotPnt_GetCursorText
5. С присланной строки снимается копия, сама исходная строка не удаляется.
6. Идентификатор нового битмапа характерной точки. Если идентификатор битмапа 0, битмап не меняется.

Матрицы преобразования

Функции данного радела обеспечивают трансформацию объектов с использованием матриц преобразования координат.

DeleteMtr– Отменить матрицу преобразования координат

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDeleteMtr.

Синтаксис:

int DeleteMtr (void);

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Описание:

Отменяет режим преобразования координат, линейных и угловых параметров, введенный функцией Mtr.

ksMtr – Создать матрицу преобразования координат

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMtr.

Синтаксис:

int ksMtr (double x, double y,

double angle,

double scaleX,

double scaleY);

Входные параметры:

x, y	- координаты начала локальной системы координат,
angle	- угол наклона системы координат в градусах,
scaleX	- масштаб локальной системы координат по оси X,
scaleY	- масштаб локальной системы координат по оси Y.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Допускается вложение матриц трансформации. В результате вложения действует суммарная матрица, полученная произведением накопленных матриц.
2. Объекты вида, вводимые между операторами ksMtr и DeleteMtr, подвергаются преобразованию по суммарной матрице.

Mtr – Создать матрицу преобразования координат

Пример...

Синтаксис:

int Mtr (double x, double y,

double ang,

double scale);

Входные параметры:

x, y	- координаты начала локальной системы координат,
ang	- угол наклона системы координат в градусах,
scale	- коэффициент масштабирования.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Введенный режим преобразования отменяется функцией DeleteMtr.

MtrForIGES – Создать матрицу преобразования координат

Пример...

Синтаксис:

int MtrForIGES (double rotateMtr[2][2],

double moveArr[2]);

Входные параметры:

rotateMtr - матрица поворота,
moveArr - вектор сдвига.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Все объекты вида, вводимые между операторами MtrForIGES и DeleteMtr, подвергаются трансформации по суммарной матрице.

Проверка условий

IsGeomObject – Проверить геометрический объект или нет

Пример...

Синтаксис:

int IsGeomObject (reference obj);

Входные параметры:

obj - указатель на объект.

Возвращаемое значение:

1 - объект геометрический,
0 - объект не геометрический.

IsObjFromAssociativeView – Проверить, принадлежит ли объект ассоциативному виду

Пример...

Синтаксис:

int IsObjFromAssociativeView (reference obj);

Входные параметры:

obj	- указатель на объект.

Возвращаемое значение:

1	- объект принадлежит ассоциативному виду,
0	- объект не принадлежит ассоциативному виду,

IsVisibleOrHiddenArraysInObject– Проверить наличие видимых или невидимых участков на кривой

Синтаксис:

int IsVisibleOrHiddenArraysInObject (reference obj);

Входные параметры:

obj	- указатель на объект кривой.

Возвращаемое значение:

1	- кривая содержит видимые и невидимых участки,
0	- кривая не содержит видимые и невидимых участки или в случае ошибки.

Примечание:

Объект может состоять из видимых и невидимых участков, если он принадлежит ассоциативному виду.

Простановка текстовых надписей

Функции работы с текстовыми надписями и таблицами

Функции данного раздела обеспечивают построение и обработку текстовых надписей (простых, сложноструктурированных, форматированных).

GetTextLength – Получить длину текста

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTextLength.

Синтаксис:

double GetTextLength (char * text, unsigned short style);

Входные параметры:

style - стиль текста.

Системные стили текстов...

Входные параметры:

text - указатель на строку текста.

Возвращаемое значение:

вычисленное значение длины текста (в миллиметрах).

Примечание:

1. Функция может принимать строку в синтаксисе версии КОМПАС 4 (спецзнаки, дроби, отклонения).
2. При использовании Unicode следует использовать функцию GetTextLengthW.

GetTextLengthW – Получить длину текста (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTextLength.

Синтаксис:

double LIB_FUNC GetTextLengthW (LPWSTR text, unsigned short style);

Входные параметры:

style - стиль текста.

Системные стили текстов...

Входные параметры:

text - указатель на строку текста.

Возвращаемое значение:

вычисленное значение длины текста (в миллиметрах).

Примечание:

1. Функция может принимать строку в синтаксисе версии КОМПАС 4 (спецзнаки, дроби, отклонения).
2. При использовании ANSI следует использовать функцию GetTextLength.

GetTextLengthFromReference – Получить длину текста, заданного указателем

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTextLengthFromReference.

Синтаксис:

double GetTextLengthFromReference (reference pText);

Входной параметр:

pText - указатель на текст.

Возвращаемое значение:

вычисленное значение длины текста (в миллиметрах).

Примечание:

Функция может принимать строку в синтаксисе версии КОМПАС 4 (спецзнаки, дроби, отклонения).

ksConvertTextToCurve – Преобразовать указанный текст в кривые

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksConvertTextToCurve.

Синтаксис:

reference ksConvertTextToCurve (reference pText);

Входной параметр:

pText - указатель на текст.

Возвращаемое значение:

указатель на временную группу кривых - в случае удачного завершения,
0 - в случае неудачи.

ksEditTextLine – Вызвать диалог редактирования сложноструктурированного текста

Аналог данной функции при использовании Automation - метод KompasObject::ksEditTextLine.

Синтаксис:

int ksEditTextLine (void *HWindow,

char *str,

int sizeStr);

Входные параметры:

HWindow	- HWND окна,
sizeStr	- размер строки str.

Выходной параметр:

str	- строка текста.

Возвращаемое значение:

1	- выход из диалога по кнопке **OK**,
0	- выход из диалога по кнопке **Отмена**.

Примечание.

При использовании Unicode следует использовать функцию ksEditTextLineW.

ksEditTextLineW – Вызвать диалог редактирования сложноструктурированного текста (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksEditTextLine.

Синтаксис:

int LIB_FUNC ksEditTextLine (void *HWindow,

LPWSTRstr,

int sizeStr);

Входные параметры:

HWindow	- HWND окна,
sizeStr	- размер строки str.

Выходной параметр:

str	- строка текста.

Возвращаемое значение:

1 - выход из диалога по кнопке **ОК**,
0 - выход из диалога по кнопке **Отмена**.

Примечание.

При использовании ANSI следует использовать функцию ksEditTextLine.

ksGetTextAlign – Получить тип привязки текста

Пример...

Справка системы КОМПАС...

kompas.chm::/528_65_5_1_Izmenenie_parametrov.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTextAlign.

Синтаксис:

int ksGetTextAlign (reference pText);

Входной параметр:

pText - указатель на объект "текст".

Возвращаемое значение:

тип привязки текста - в случае успешного завершения,
-1 - в случае неудачи.

Типы привязки текста...

ksSetTableColumnText – Задать текст ячейки таблицы

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetTableColumnText.

Синтаксис:

int ksSetTableColumnText (unsigned int numb, TextParam *par);

Входные параметры:

numb - номер ячейки,

Выходные параметры:

par - указатель на структуру параметров строки текста TextParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования таблицы:

ksOpenTable ();

ksSetTableColumnText ();

EndObj ();

ksSetTextAlign – Установить тип привязки текста

Пример...

Справка системы КОМПАС...

kompas.chm::/528_65_5_1_Izmenenie_parametrov.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetTextAlign.

Синтаксис:

int ksSetTextAlign (reference pText, unsigned int align);

Входные параметры:

pText	- указатель на объект "текст",
align	- тип привязки текста.

Типы привязки текста...

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSetTextLineAlign – Установить выравнивание текста

Пример...

Справка системы КОМПАС...

kompas.chm::/528_65_5_1_Izmenenie_parametrov.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetTextLineAlign.

Синтаксис:

int ksSetTextLineAlign (unsigned char align);

Входной параметр:

align	- признак выравнивания: 0 - по левому краю, 1 - по центру, 2 - по правому краю, 3 - по ширине.

Возвращаемое значение:

предыдущий признак выравнивания	- в случае успешного завершения,
-1	- в случае неудачи.

Примечание.

Функция ksSetTextLineAlign должна использоваться внутри блока, т.е. необходимо, чтобы был открытый на редактирование текст. Иначе функция не работает.

Paragraph – Начать параграф

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksParagraph.

Синтаксис:

int Paragraph (ParagraphParam *par);

Выходной параметр:

par	- указатель на структуру параметров параграфа ParagraphParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Описание:

1. Параграфом называется автоматически форматируемый блок текста (левосторонний, правосторонний, центрированный, по двум сторонам).
2. При работе в текстовом процессоре он завершается символом <ENTER>. Описание параграфа заканчивается функцией EndObj, возвращающей указатель на созданный объект.

Text – Создать строку текста в графическом документе

Пример...

Справка системы КОМПАС...

kompas.chm::/562_Glava67_Tekst_v_grafichesko.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksText.

Синтаксис:

reference Text (double x,
double y,
double ang,
double hStr,
double ksuStr,
unsigned int bitVector,
char *s);

Входные параметры:

x, y	- координаты точки привязки текста,
ang	- угол наклона текста,
hStr	- высота символов,
ksuStr	- сужение текста,
bitVector	- битовый вектор, задающий признаки начертания текста,
s	- строка символов.

Признаки начертания текста...

Возвращаемое значение:

указатель на текст	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Строка символов может включать спецсимвол. Например, чтобы задать 10 градусов, необходимо указать значение "10&01". Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
2. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ #, кроме случаев, когда эти символы используются в строке именно как управляющие.
3. Использование управляющих символов:

- отклонение: $ верхнее отклонение; нижнее отклонение $
- дробь: $d числитель ; знаменатель $
- спецсимвол: &nn номер спецсимвола 0...99.

4. bitVector формируется с помощью логической операции |. Поддерживаются определения:

- ITALIC_ON (включить наклон),
- BOLD_ON (включить утолщение),
- UNDERLINE_ON (включить подчеркивание). См. ltdefine.h.

5. При использовании Unicode следует использовать функцию TextW.

6. При использовании внутри таблицы функция Text не возвращает указатель на текст— в случае успешного завершения возвращается 1.

TextW – Создать строку текста в графическом документе (Unicode)

Справка системы КОМПАС...

`kompas.chm::/562_Glava67_Tekst_v_grafichesko.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksText.

Синтаксис:

```
reference TextW (double x,
double y,
double ang,
double hStr,
double ksuStr,
unsigned int bitVector,
LPWSTR s);
```

Входные параметры:

x, y	- координаты точки привязки текста,
ang	- угол наклона текста,
hStr	- высота символов,
ksuStr	- сужение текста,
bitVector	- битовый вектор, задающий признаки начертания текста,
s	- строка символов.

Признаки начертания текста...

Возвращаемое значение:

указатель на текст	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Строка символов может включать спецсимвол. Например, чтобы задать 10 градусов, необходимо указать значение "10&01". Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
2. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ #, кроме случаев, когда эти символы используются в строке именно как управляющие."
3. Использование управляющих символов:

- отклонение: $ верхнее отклонение ; нижнее отклонение $
- дробь: $d числитель ; знаменатель $
- спецсимвол: &nn номер спецсимвола 0...99.

4. bitVector формируется с помощью логической операции |. Поддерживаются определения:

- ITALIC_ON (включить наклон),
- BOLD_ON (включить утолщение),
- UNDERLINE_ON (включить подчеркивание). См. ltdefine.h.

5. При использовании ANSI следует использовать функцию Text.
6. При использовании внутри таблицы функция TextW не возвращает указатель на текст — в случае успешного завершения возвращается 1.

TextLine – Задать подстроку параграфа текста

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTextLine.

Синтаксис:

int TextLine (unsigned int bitVector, int type, void *value, char * s);

Входной параметр:

bitVector	- значения свойств, устанавливаемые двоичными флагами,
type	- тип свойства.

Выходной параметр:

value	- указатель на значение свойства,
s	- создаваемый компонент текста.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция предназначена для задания сложноструктурированного текста, отдельные компоненты которого имеют различные параметры.
2. Спецзнак задается отдельной компонентой. Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
3. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ # кроме случаев, когда эти символы используются в строке именно как управляющие."
4. Использование управляющих символов:

- отклонение: $ верхнее отклонение ; нижнее отклонение $
- дробь: $d числитель ; знаменатель $
- спецсимвол: &nnn номер спецсимвола

5. bitVector формируется с помощью логической операции | из набора определений [INVARIABLE ...NEW_LINE] (см. ltdefine.h).
6. При использовании Unicode следует использовать функцию TextLineW.

TextLineW – Задать подстроку параграфа текста (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTextLine.

Синтаксис:

int TextLineW (unsigned int bitVector, int type, void *value, LPWSTR s);

Входной параметр:

bitVector	- значения свойств, устанавливаемые двоичными флагами,
type	- тип свойства.

Выходной параметр:

value	- указатель на значение свойства,
s	- создаваемый компонент текста.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция предназначена для задания сложноструктурированного текста, отдельные компоненты которого имеют различные параметры.
2. Спецзнак задается отдельной компонентой. Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
3. Не рекомендуется использовать в строке символы, которые могут идентифицироваться как управляющие символы: @ $ & ; ~ ^ #, кроме случаев, когда эти символы используются в строке именно как управляющие."
4. Использование управляющих символов:

- отклонение: $ верхнее отклонение ; нижнее отклонение $
- дробь: $d числитель ; знаменатель $
- спецсимвол: &nnn номер спецсимвола

5. bitVector формируется с помощью логической операции | из набора определений [INVARIABLE ...NEW_LINE] (см. ltdefine.h).
6. При использовании ANSI следует использовать функцию TextLine.

Работа с таблицей

Функции данного раздела обеспечивают оформление таблиц.

ksClearTableColumnText – Очистить ячейку таблицы или допуска формы

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksClearTableColumnText.

Синтаксис:

int ksClearTableColumnText (unsigned int numb);

Входной параметр:

numb - номер ячейки,

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если numb = 0, очищается вся таблица.
2. Нумерация ячеек начинается с 1.
3. Функция используется в режиме редактирования таблицы или допуска формы.

ksCombineTwoTableItems – Объединить две ячейки таблицы, если они имеют общее ребро

Пример...

Справка системы КОМПАС...

`kompas.chm::/584_69_1_3_Obwedinenie_jacheek.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCombineTwoTableItems.

Синтаксис:

int ksCombineTwoTableItems (unsigned int index1, unsigned int index2);

Входные параметры:

index1 - номер первой ячейки,
index2 - номер второй ячейки.

Возвращаемое значение:

1 - в случае успешного завершения,

0 - в случае неудачи.

Примечание:

1. Нумерация ячеек начинается с 1.
2. Функция используется в режиме редактирования таблицы или допуска формы.

ksDivideTableItem – Разделить ячейку таблицы

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDivideTableItem.

Синтаксис:

int ksDivideTableItem (unsigned int index, unsigned char vertical, unsigned short style);

Входные параметры:

index	- номер ячейки,
vertical	- направление разделения ячейки:
	1 - вертикально,
	0 - горизонтально,
style	- стиль линии получившейся границы:
	0 - невидимая,
	1 - основная,
	2 - тонкая,
	7 - утолщенная.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Нумерация ячеек начинается с 1.
2. Функция используется в режиме редактирования таблицы или допуска формы.

ksGetPointOnToleranceTable – Получить координаты точки на таблице допуска формы

Пример,,,

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetPointOnToleranceTable.

Синтаксис:

int ksGetPointOnToleranceTable (reference tolerance, unsigned char entry, MathPointParam * point);

Входные параметры:

tolerance	- указатель на допуск формы,
entry	- положение базовой точки.

Выходной параметр:

point - указатель на структуру параметров математической точки MathPointParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksGetTableBorderStyle – Получить стиль границы ячейки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/587_69_1_7_Granicy_jacheek.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTableBorderStyle.

Синтаксис:

int ksGetTableBorderStyle (unsigned int index, unsigned char typeBorder);

Входные параметры:

index	- номер ячейки,
typeBorder	- тип границы: 0 - левая, 1 - правая, 2 - верхняя, 3 - нижняя.

Возвращаемое значение:

- стиль линии границы: 0 - невидимая, 1 - основная, 2 - тонкая, 7 - утолщенная.	- в случае успешного завершения,
-1	- в случае неудачи.

Примечание:

1. Нумерация ячеек начинается с 1.
2. Функция используется в режиме редактирования таблицы или допуска формы.

ksGetTableColumnText – Получить текст ячейки

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTableColumnText.

Синтаксис:

int ksGetTableColumnText (unsigned int *numb, TextParam *par);

Выходные параметры:

numb	- номер ячейки,
par	- указатель на структуру параметров текста TextParam.

Возвращаемое значение:

указатель на динамический массив строк текста - TEXT_LINE_ARR	- в случае успешного завершения,
0	- все графы пройдены,
номер графы	- в случае ошибки.

Примечание:

1. Функция используется в режиме редактирования таблицы.
2. Нумерация ячеек начинается с единицы.
3. Если не определен номер ячейки с помощью функции ColumnNumber, функция начинает работу с первой ячейки.
4. После выполнения метода происходит смещение на следующую ячейку.

ksGetTableItemsCount – Получить количество ячеек в таблице

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetTableItemsCount.

Синтаксис:

int ksGetTableItemsCount (unsigned char type);

Входной параметр:

type - признак, какое число требуется получить:

0 - общее число ячеек,

1 - число ячеек в строке,

2 - число ячеек в колонке.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Функция используется в режиме редактирования таблицы или допуска формы.

ksGetToleranceColumnText – Получить текст ячейки

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetToleranceColumnText.

Синтаксис:

int ksGetToleranceColumnText (unsigned int *numb, TextLineParam *par);

Выходные параметры:

numb - номер ячейки,
par - указатель на структуру параметров строки текста TextLineParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция используется в режиме редактирования допуска формы.
2. Нумерация ячеек начинается с единицы.
3. Если не определен номер ячейки с помощью функции ColumnNumber, функция начинает работу с первой ячейки.
4. После выполнения функции происходит смещение на следующую ячейку.
5. Если par=0, все графы пройдены.
6. После использования массив par->pTextItem желательно удалить.
7. Если numb = NULL, функция возвращает номер графы.

ksOpenTable – Открыть составной объект-таблицу для редактирования

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenTable.

Синтаксис:

int ksOpenTable (reference table);

Входной параметр:

table - указатель на таблицу.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Чтобы закрыть редактирование таблицы, нужно вызвать функцию EndObj.

ksOpenTolerance – Открыть допуск формы для редактирования

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenTolerance.

Синтаксис:

int ksOpenTolerance (reference tolerance);

Входной параметр:

tolerance - указатель на допуск формы.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Чтобы завершить редактирование допуска формы, нужно вызвать функцию EndObj.

ksReadTableFromFile – Создать таблицу, используя файл таблицы *.tbl

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_GENERATE_TBL.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksReadTableFromFile.

Синтаксис:

reference ksReadTableFromFile (char * tblFileName);

Выходной параметр:

tblFileName	- полное имя файла таблицы.

Возвращаемое значение:

указатель на таблицу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksReadTableFromFileW.

ksReadTableFromFileW – Создать таблицу, используя файл таблицы *.tbl (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_GENERATE_TBL.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksReadTableFromFile.

Синтаксис:

reference LIB_FUNC ksReadTableFromFile (LPWSTR tblFileName);

Выходной параметр:

tblFileName	- полное имя файла таблицы.

Возвращаемое значение:

указатель на таблицу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksReadTableFromFile.

ksRebuildTableVirtualGrid – Перестроить виртуальную сетку таблицы

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRebuildTableVirtualGrid.

Синтаксис:

int ksRebuildTableVirtualGrid();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция используется в режиме редактирования таблицы или допуска формы. Таблица работает с виртуальной сеткой - регулярная таблица, наложенная на редактируемую таблицу, отображающая все ячейки. Если редактируемая таблица регулярная, то ее виртуальная сетка полностью совпадает с таблицей.
2. Нумерация ячеек начинается с левого верхнего угла и с единицы по строкам.
3. Перестраивать сетку нужно после объединения или разделения ячеек.

ksSetTableBorderStyle – Изменить стиль границы ячейки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/587_69_1_7_Granicy_jacheek.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetTableBorderStyle.

Синтаксис:

int ksSetTableBorderStyle (unsigned int index, unsigned char typeBorder, unsigned short style);

Входные параметры:

index	- номер ячейки,
typeBorder	- тип границы:
	0 - левая,
	1 - правая,
	2 - верхняя,
	3 - нижняя,
style	- стиль линии границы:
	0 - невидимая,
	1 - основная,
	2 - тонкая,
	7 - утолщенная.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Нумерация ячеек начинается с 1.
2. Функция используется в режиме редактирования таблицы или допуска формы.

ksSetToleranceColumnText – Задать текст ячейки допуска формы

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetToleranceColumnText.

Синтаксис:

int ksSetToleranceColumnText (unsigned int numb, TextLineParam *par);

Входные параметры:

numb - номер ячейки,

Выходные параметры:

par - указатель на структуру параметров строки текста TextLineParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Функция используется в режиме редактирования допуска формы.

Table – Создать таблицу в графическом документе

Пример...

Справка системы КОМПАС...

kompas.chm::/582_Glava69_Obshchie_svedenija.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTable.

Синтаксис:

reference Table (void);

Возвращаемое значение:

указатель на таблицу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. После вызова функции и далее до вызова функции EndObj объекты чертежа (включая вертикальные и горизонтальные отрезки и тексты) будут объединены в таблицу.
2. Таблица может быть регулярной и нерегулярной.
3. Допустимые стили линии для отрезков таблицы:

- 1 - основная,
- 2 - тонкая,
- 7 - утолщенная,
- 0 - невидимая.

Оформление чертежа

Функции оформления чертежа

Функции данного радела обеспечивают общую работу с документами (открытие, закрытие, запись документа), работу с видами а также общее оформление и компоновку документа - задание технических требований, основной надписи штампа, использование типовых шаблонов.

ClearStamp – Очистить графу штампа чертежа/ текстового документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksStamp::ksClearStamp.

Синтаксис:

int ClearStamp (unsigned int numb);

Входной параметр:

numb	- номер очищаемой ячейки основной надписи (0 - очищается вся основная надпись).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Метод действителен для чертежа и спецификации.
2. Функция устарела, рекомендуется вместо нее использовать функцию ClearStampEx.

ClearStampEx – Очистить графу штампа чертежа/текстового документа по номеру листа

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Синтаксис:

int ClearStampEx (int sheetNumb, unsigned int numb);

Входной параметр:

sheetNumb	- номер листа, начиная с 1,
numb	- номер очищаемой ячейки основной надписи (0 - очищается вся основная надпись).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод действителен для чертежа и спецификации.

CloseStamp – Закрыть штамп чертежа/текстового документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksStamp::ksCloseStamp.

Синтаксис:

reference CloseStamp();

Возвращаемое значение:

указатель на штамп	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод действителен для чертежа и спецификации.

ColumnNumber – Определить номер графы штампа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksColumnNumber.

Синтаксис:

int ColumnNumber (unsigned int numb);

Входной параметр:

numb	- номер ячейки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция используется в режиме редактирования штампа, таблицы, допуска формы, в режиме создания допуска формы.

GetStampColumnText – Получить текст графы штампа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksStamp::ksGetStampColumnText.

Синтаксис:

reference GetStampColumnText (unsigned int *numb);

Выходной параметр:

numb	- номер ячейки основной надписи.

Возвращаемое значение:

указатель на динамический массив строк текста - TEXT_LINE_ARR и номер графы (если numb!= NULL)..	- в случае успеха,
0	- когда все графы пройдены или в случае ошибки.

Примечание:

1. Функция используется в режиме редактирования штампа.
2. Функция выдает текст графы с указанным номером и смещается на следующую графу. Если не определен номер графы с помощью функции ColumnNumber или при предыдущем вызове самой этой функции, начинает с первой графы.
3. После использования полученный массив желательно удалить.

GetReferenceDocumentPart – Получить указатель на основную надпись, или технические требования, или неуказанную шероховатость, или текущий вид, или спецификацию на листе

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetReferenceDocumentPart.

Синтаксис:

reference GetReferenceDocumentPart (unsigned char t);

Входной параметр:

t - Тип объекта оформления чертежа.

Возвращаемое значение:

указатель на объект выбранного типа - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Метод предназначен для использования в листе чертежа.

GetReferenceDocumentPartEx – Получить указатель на объект оформления чертежа

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetReferenceDocumentPartEx.

Синтаксис:

reference GetReferenceDocumentPartEx (unsigned char t, int sheetNumb);

Входные параметры:

t	- Тип объекта оформления чертежа.
sheetNumb	- зависит от типа получаемого объекта оформления чертежа:
	для t=0 sheetNumb - номер листа, начиная с 1,
	для t=1 sheetNumb не используется,
	для t=2 sheetNumb не используется,
	для t=3 sheetNumb не используется,
	для t=4 sheetNumb - номер спецификации, начиная с 1; 0 - текущая спецификация,
	для t=5 sheetNumb не используется,
	для t=6 sheetNumb - номер листа, начиная с 1.

Возвращаемое значение:

указатель на объект выбранного типа	- в случае удачного завершения,
0	- в случае неудачи.

ksGetZona – Получить зону текущего чертежа по заданной точке

Пример...

Справка системы КОМПАС: определение зоны источника...

KOMPAS.chm::/568_67_3_1_Sozdanie.htm

Разбиение на зоны...

KOMPAS.chm::/DLG_ZONE_SETUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetZona.

Синтаксис:

```
int ksGetZona (double x,
double y,
char * zona,
int size);
```

Входные параметры:

x, y	- координаты точки в текущем документе.
size	- размер буфера.

Выходной параметр:

zona	- буфер для возврата зоны,

Возвращаемое значение:

1 - успешное завершение,
0 - ошибка (например, точка вне документа или документ - не чертеж),
-1 - в текущем документе нет разбиения на зоны.

Примечание.

При использовании Unicode следует использовать функцию ksGetZonaW.

ksGetZonaW – Получить зону текущего чертежа по заданной точке (Unicode)

Справка системы КОМПАС: определение зоны источника...

KOMPAS.chm::/568_67_3_1_Sozdanie.htm

Разбиение на зоны...

KOMPAS.chm::/DLG_ZONE_SETUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetZona.

Синтаксис:

int LIB_FUNC ksGetZonaW (double x,
double y,
LPWSTR zona,
int size);

Входные параметры:

x, y - координаты точки в текущем документе,
size - размер буфера.

Выходной параметр:

zona - буфер для возврата зоны.

Возвращаемое значение:

1 - успешное завершение,
0 - ошибка (например, точка вне документа или документ - не чертеж),
-1 - в текущем документе нет разбиения на зоны.

Примечание.

При использовании ANSI следует использовать функцию ksGetZona.

ksRebuildDocument – Перестроить графический документ

Справка системы КОМПАС...

KOMPAS.chm::/CM_REBUILD_SHEET.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRebuildDocument.

Синтаксис:

int ksRebuildDocument (reference sheet);

Входные параметры:

sheet	- указатель на графический документ.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Позволяет перестроить все ассоциативные виды чертежа. Если в чертеже нет ни одного ассоциативного вида, перестраиваться ничего не будет. После вызова функции ассоциативные виды перерисовываются в соответствии с деталями, изображение которых в них содержится.
2. Если sheet = 0, перестраивается текущий документ.

OpenStamp – Открыть штамп чертежа/текстового документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Аналог данной функции при использовании Automation - метод ksStamp::ksOpenStamp.

Синтаксис:

int OpenStamp();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Следующие далее функции до вызова CloseStamp определяют текст граф основной надписи штампа.
2. Функция устарела, рекомендуется вместо нее использовать функцию OpenStampEx.

OpenStampEx – Открыть составной объект "штамп" по номеру листа

Справка системы КОМПАС...

KOMPAS.chm::/574_68_2_Osnovnaja_nadpisq_i_fo.htm

Синтаксис:

int OpenStampEx (int sheetNumb);

Входные параметры:

sheetNumb	- номер листа, начиная с 1.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Операторы ColumnNumber и TextLine, вводимые между операторами OpenStampEx и CloseStamp, принадлежат штампу листа с номером sheetNumb.
2. ColumnNumber определяет номер ячейки, куда помещать текст.
3. Номера определены в соответствии с ГОСТ на данный штамп.
4. CloseStamp возвращает указатель на штамп.

SetStampColumnText – Задать текст графы штампа

Пример...

Аналог данной функции при использовании Automation - метод ksStamp::ksSetStampColumnText.

Синтаксис:

int SetStampColumnText (unsigned int numb,

reference textArr);

Входные параметры:

numb	- номер ячейки основной надписи,
textArr	- указатель на динамический массив строк текста - TEXT_LINE_ARR.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Метод используется в режиме редактирования штампа.

Работа со слоями

Функции работы со слоями

Функции данного радела обеспечивают работу со слоями вида или фрагмента чертежа (создание и обработка слоя, слияние слоев и редактирование их параметров).

ChangeObjectLayer – Изменить слой объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksChangeObjectLayer.

Синтаксис:

int ChangeObjectLayer (reference obj, int number);

Входные параметры:

obj - указатель на объект,
number - номер слоя, на который переносится объект.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Слой, на который переносится объект, должен существовать и быть доступным для редактирования (не фоновым и не выключенным).

GetLayerNumber – Получить номер слоя

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetLayerNumber.

Синтаксис:

int GetLayerNumber (reference p);

Входной параметр:

p - указатель на слой.

Возвращаемое значение:

номер слоя - в случае удачного завершения,
-1 - в случае неудачи.

Примечания:

1. Системный слой имеет номер 0.
2. Если p - объект слоя, возвращается номер слоя этого объекта.
3. Если p = 0, то возвращается номер текущего слоя.

GetLayerReference – Получить указатель на слой по номеру слоя текущего вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetLayerReference.

Синтаксис:

reference GetLayerReference (int number);

Входной параметр:

number - номер слоя.

Возвращаемое значение:

указатель на слой - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Возврат нулевого значения означает, что слой с таким номером в документе отсутствует.

Layer – Сделать слой текущим

Пример...

Справка системы КОМПАС...

`kompas.chm::/401_46_4_Perekljuchenie_mezhdu_.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLayer.

Синтаксис:

reference Layer (int n);

Входной параметр:

n - номер слоя.

Возвращаемое значение:

указатель на слой - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Если слоя с заданным номером нет, он создается.

Работа с видами

Функции работы с видами

Функции данного радела обеспечивают работу с видами чертежа (создание и изменение их параметров).

CreateSheetView – Создать новый вид в чертеже

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATESHEETVIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetView.

Синтаксис:

reference CreateSheetView(ViewParam *par,

int *number);

Входные параметры:

number - номер вида.

Выходные параметры:

par - указатель на структуру параметров вида,

Возвращаемое значение:

указатель на вид - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Созданный вид становится текущим.
2. Если указать *number = 0, то создается вид с номером по возрастанию.
3. Если *number = n, создается вид с номером n. Если вид n существует, ничего не создается (reference = 0) - ошибка.
4. При использовании Unicode следует использовать функцию CreateSheetViewW.

CreateSheetViewW – Создать новый вид в чертеже (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATESHEETVIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetView.

Синтаксис:

reference LIB_FUNC CreateSheetView(ViewParamW *par,

int *number);

Входные параметры:

number - номер вида.

Выходные параметры:

par - указатель на структуру параметров вида,

Возвращаемое значение:

указатель на вид - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Созданный вид становится текущим.
2. Если указать *number = 0, то создается вид с номером по возрастанию.
3. Если *number = n, создается вид с номером n. Если вид n существует, ничего не создается (reference = 0) - ошибка.
4. При использовании ANSI следует использовать функцию CreateSheetView.

GetViewNumber – Получить номер вида по указателю на вид или объект вида

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/387_45_1_Perekljuchenie_mezhdu_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetViewNumber.

Синтаксис:

int GetViewNumber (reference p);

Входной параметр:

p	- указатель на вид или объект вида.

Возвращаемое значение:

номер вида	- в случае удачного завершения,
-1	- в случае неудачи.

Примечание:

1. Если p - указатель на объект вида, возвращается номер вида этого объекта.
2. Если p = 0, то возвращается номер текущего вида.

GetViewReference – Получить указатель на вид по номеру вида

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/387_45_1_Perekljuchenie_mezhdu_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetViewReference.

Синтаксис:

reference GetViewReference(int number);

Входной параметр:

number	- номер вида.

Возвращаемое значение:

указатель на вид	- в случае удачного завершения,
0	- в случае неудачи.

ksAssociationViewMatrix3D – Создать матрицу ассоциативного вида

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_VIEWS_SCHEMA.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAssociationViewMatrix3D.

Синтаксис:

VARIANT ksAssociationViewMatrix3D (reference view);

Входные параметры:

view - указатель на ассоциативный вид.

Возвращаемое значение:

SAFEARRAY double (VT_ARRAY | VT_R8)

В массиве будут лежать 16 элементов, которые представляют собой матрицу размера 4x4.

Примечание:

Если указатель на вид не задан (view = 0), то функция будет выполняться для текущего вида.

ksCreateSheetArbitraryView – Создать произвольный ассоциативный вид

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_ARBITRARY_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetArbitraryView.

Синтаксис:

reference ksCreateSheetArbitraryView (AssociationViewParam * par, int *number);

Входные параметры:

number - номер создаваемого вида.
par - структура параметров ассоциативного вида AssociationViewParam.

Возвращаемое значение:

указатель на вид - в случае успешного завершения,
0 - в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Если значение параметра number = 0, то создается вид с номером по возрастанию.
3. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
4. При использовании Unicode следует использовать функцию ksCreateSheetArbitraryViewW.

ksCreateSheetArbitraryViewW – Создать произвольный ассоциативный вид (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_ARBITRARY_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetArbitraryView.

Синтаксис:

reference LIB_FUNC ksCreateSheetArbitraryView (AssociationViewParamW * par, int *number);

Входные параметры:

number	- номер создаваемого вида,
par	- структура параметров ассоциативного вида AssociationViewParam.

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Если значение параметра number = 0, то создается вид с номером по возрастанию.
3. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
4. При использовании ANSI следует использовать функцию ksCreateSheetArbitraryView.

ksCreateSheetArrowView – Создать ассоциативный вид по стрелке

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_ARROW_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetArrowView.

Синтаксис:

reference ksCreateSheetArrowView (AssociationViewParam *par,
int *number,
reference obj);

Входные параметры:

par	- структура параметров ассоциативного вида AssociationViewParam,
number	- номер создаваемого вида,
obj	- указатель на объект "стрелка вида".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "стрелка вида" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании Unicode следует использовать функцию ksCreateSheetArrowViewW.

ksCreateSheetArrowViewW – Создать ассоциативный вид по стрелке (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_ARROW_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetArrowView.

Синтаксис:

reference LIB_FUNC ksCreateSheetArrowViewW (AssociationViewParamW *par,
int *number,
reference obj);

Входные параметры:

par - структура параметров ассоциативного вида AssociationViewParam,
number - номер создаваемого вида,
obj - указатель на объект "стрелка вида".

Возвращаемое значение:

указатель на вид - в случае успешного завершения,
0 - в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "стрелка вида" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании ANSI следует использовать функцию ksCreateSheetArrowView.

ksCreateSheetProjectionView – Создать проекционный ассоциативный вид

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_PROJECTION_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetProjectionView.

Синтаксис:

reference ksCreateSheetProjectionView(AssociationViewParam * par,
int * number,
reference view);

Входные параметры:

par - структура параметров ассоциативного вида AssociationViewParam,
number - номер создаваемого вида,
view - указатель на базовый вид.

Возвращаемое значение:

указатель на вид - в случае успешного завершения,
0 - в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Базовый вид должен быть ассоциативным.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании Unicode следует использовать функцию ksCreateSheetProjectionViewW.

ksCreateSheetProjectionViewW – Создать проекционный ассоциативный вид (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_PROJECTION_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetProjectionView.

Синтаксис:

```
reference LIB_FUNC ksCreateSheetProjectionViewW( AssociationViewParamW * par,
int * number,
reference view);
```

Входные параметры:

par	- структура параметров ассоциативного вида AssociationViewParam,
number	- номер создаваемого вида,
view	- указатель на базовый вид.

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Базовый вид должен быть ассоциативным.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании ANSI следует использовать функцию ksCreateSheetProjectionView.

ksCreateSheetSectionView – Создать ассоциативный вид разреза/сечения

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_SECTION_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetSectionView.

Синтаксис:

reference ksCreateSheetSectionView (AssociationViewParam *par,
int *number,
reference obj);

Входные параметры:

par - структура параметров ассоциативного вида AssociationViewParam,
number - номер создаваемого вида,
obj - указатель на объект "линия разреза".

Возвращаемое значение:

указатель на вид - в случае успешного завершения,
0 - в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "линия разреза" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании Unicode следует использовать функцию ksCreateSheetSectionViewW.

ksCreateSheetSectionViewW – Создать ассоциативный вид разреза/сечения (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_SECTION_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetSectionView.

Синтаксис:

reference LIB_FUNC ksCreateSheetSectionViewW (AssociationViewParamW *par,
int *number,
reference obj);

Входные параметры:

par - структура параметров ассоциативного вида AssociationViewParam,
number - номер создаваемого вида,
obj - указатель на объект "линия разреза".

Возвращаемое значение:

указатель на вид - в случае успешного завершения,
0 - в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "линия разреза" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании ANSI следует использовать функцию ksCreateSheetSectionView.

ksCreateSheetStandartViews – Создать стандартные ассоциативные виды

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_STANDART_VIEW.htm
Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetStandartViews.

Синтаксис:

int ksCreateSheetStandartViews (AssociationViewParam *par,
unsigned int bitVector,
double dx,
double dy);

Входные параметры:

par - структура параметров ассоциативного вида AssociationViewParam,
bitVector - набор типов стандартных видов, которые нужно создать,
dx - расстояние между видами по горизонтали,
dy - расстояние между видами по вертикали.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Последний из созданных видов становится текущим.
2. При использовании Unicode следует использовать функцию ksCreateSheetStandartViewsW.

ksCreateSheetStandartViewsW – Создать стандартные ассоциативные виды (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_CREATE_STANDART_VIEW.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetStandartViews.

Синтаксис:

int LIB_FUNC ksCreateSheetStandartViewsW (AssociationViewParamW *par,
unsigned int bitVector,
double dx,
double dy);

Входные параметры:

par	- структура параметров ассоциативного вида AssociationViewParam,
bitVector	- набор типов стандартных видов, которые нужно создать,
dx	- расстояние между видами по горизонтали,
dy	- расстояние между видами по вертикали.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Последний из созданных видов становится текущим.
2. При использовании ANSI следует использовать функцию ksCreateSheetStandartViews.

ksCreateSheetRemoteView – Создать ассоциативный выносной вид

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetRemoteView.

Синтаксис:

reference ksCreateSheetRemoteView (AssociationViewParam *par,
int *number,
reference obj);

Входные параметры:

par	- структура параметров ассоциативного вида AssociationViewParam,
number	- номер создаваемого вида,
obj	- указатель на объект "Обозначение выносного элемента".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "Обозначение выносного элемента" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании Unicode следует использовать функцию ksCreateSheetRemoteViewW.

ksCreateSheetRemoteViewW – Создать ассоциативный выносной вид (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateSheetRemoteView.

Синтаксис:

reference LIB_FUNC ksCreateSheetRemoteViewW (AssociationViewParamW *par,
int *number,
reference obj);

Входные параметры:

par	- структура параметров ассоциативного вида AssociationViewParam,
number	- номер создаваемого вида,
obj	- указатель на объект "Обозначение выносного элемента".

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Примечания:

1. Созданный вид становится текущим.
2. Объект "Обозначение выносного элемента" должен быть построен в ассоциативном виде. Этот вид будет базовым для создаваемого вида.
3. Если значение параметра number = 0, то создается вид с номером по возрастанию.
4. Если значение параметра *number = n, где n = 1 … 255, то создается вид с номером n. Если вид n существует, то ничего не создается.
5. При использовании ANSI следует использовать функцию ksCreateSheetRemoteView.

ksGetQualityNames – Получить массив полей допусков, которые поддерживают размер dimValue и не превышают указанных отклонений

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetQualityNames.

Синтаксис:

int ksGetQualityNames (reference names,

double dimValue,

double high,

double low,

unsigned char system,

unsigned char withLimitation);

Входные параметры:

names	- указатель на динамический массив полей допусков,
dimValue	- номинальное значение размера (в мм),
high	- верхнее отклонение,
low	- нижнее отклонение,
system	- система:
	1 - отверстия,
	0 - вала.
withLimitation	- признак учета ограничений:
	0 - без учёта ограничений,
	1 - с учётом ограничений, наложенных в системе.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksPoint3DToAssociationView – Преобразовать координаты 3D точки в координаты ассоциативного вида

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPoint3DToAssociationView.

Синтаксис:

int ksPoint3DToAssociationView (reference view,
double x3D, double y3D, double z3D,
double * x2D, double * y2D);

Входные параметры:

view - указатель на ассоциативный вид,
x3D, y3D, z3D - координаты 3D точки.

Выходные параметры:

x2D, y2D - координаты 3D точки в ассоциативном виде.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если указатель на вид не задан (view = 0), то функция будет выполняться для текущего вида.

NewViewNumber – Определить номер следующего вида

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/387_45_1_Perekljuchenie_mezhdu_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNewViewNumber.

Синтаксис:

int NewViewNumber();

Возвращаемое значение:

номер следующего вида	- в случае успешного завершения,
0	- в случае неудачи.

OpenView – Сделать текущим существующий вид с указанным номером

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/379_44_2_Sostojanija_vidov.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenView.

Синтаксис:

reference OpenView (int view);

Входной параметр:

view	- номер вида, который требуется открыть.

Возвращаемое значение:

указатель на вид	- в случае успешного завершения,
0	- в случае неудачи.

Описание:

Открытый вид становится текущим (после открытия или создания чертежа по умолчанию текущим является системный вид с номером 0).

Размеры и технологические обозначения

Функции построения размеров /обозначений

Функции данного раздела обеспечивают простановку в чертеже обозначений различных типов: размеры, обозначения шероховатости, линии-выноски, обозначения базы, допуски формы, линии разреза или сечения, стрелки вида.

AngBreakDimension – Проставить угловой размер с обрывом

Справка системы КОМПАС...

KOMPAS.chm::/CM_CUT_DIMA.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAngBreakDimension.

Синтаксис:

reference AngBreakDimension (ABreakDimParam *angPar);

Выходной параметр:

angPar	- указатель на структуру параметров углового размера с обрывом ABreakDimParam.

Возвращаемое значение:

указатель на угловой размер с обрывом	- в случае удачного завершения,
0	- в случае неудачи.

AngDimension – Проставить угловой размер

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_DIMA.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAngDimension.

Синтаксис:

reference AngDimension (ADimParam *Par);

Выходной параметр:

Par	- указатель на структуру параметров углового размера ADimParam.

Возвращаемое значение:

указатель на угловой размер	- в случае удачного завершения,
0	- в случае неудачи.

Base – Проставить обозначения базы

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BASE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksBase.

Синтаксис:

reference Base (BaseParam *basePar);

Выходной параметр:

basePar - указатель на структуру параметров обозначения базы BaseParam.

Возвращаемое значение:

указатель на обозначение базы - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию BaseW.

BaseW – Проставить обозначения базы (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_BASE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksBase.

Синтаксис:

reference BaseW (BaseParamW *basePar);

Выходной параметр:

basePar - указатель на структуру параметров обозначения базы BaseParamW.

Возвращаемое значение:

указатель на обозначение базы - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию Base.

BrandLeader – Создать линию-выноску для обозначения клеймения

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BRANDLEADER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksBrandLeader.

Синтаксис:

reference BrandLeader(BrandLeaderParam *leaderPar);

Выходной параметр:

brandLeaderParam	- указатель на структуру параметров линии-выноски для обозначения клеймения BrandLeaderParam.

Возвращаемое значение:

указатель на линию-выноску для обозначения клеймения	- в случае удачного завершения,
0	- в случае неудачи.

CloseTechnicalDemand – Закрыть технические требования

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/IPR_TEXT_SLAVE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCloseTechnicalDemand.

Синтаксис:

reference CloseTechnicalDemand ();

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

CutLine – Создать линию разреза/сечения

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CUTLINE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCutLine.

Синтаксис:

reference CutLine (CutLineParam *cutPar);

Выходной параметр:

cutPar	- указатель на структуру параметров линии разреза/сечения CutLineParam.

Возвращаемое значение:

указатель на линию разреза/сечения	- в случае удачного завершения,
0	- в случае неудачи.

DiamDimension – Проставить диаметральный размер

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_DIMD.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDiamDimension.

Синтаксис:

reference DiamDimension (RDimParam *Par);

Выходной параметр:

Par	- указатель на структуру параметров диаметрального размера RDimParam.

Возвращаемое значение:

указатель на диаметральный размер	- в случае удачного завершения,
0	- в случае неудачи.

ksAxisLine – Создать объект "Осевая линия"

Справка системы КОМПАС...

KOMPAS.chm::/CM_AXEDLINESEG.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAxisLine.

Синтаксис:

reference ksAxisLine (AxisLineParam * param);

Входные параметры:

param	- указатель на структуру параметров осевой линии AxisLineParam.

Возвращаемое значение:

указатель на объект "Осевая линия"	- в случае успешного завершения,
0	- в случае неудачи.

ksCentreMarker – Создать обозначение центра

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_CENTRE_MARKER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCentreMarker.

Синтаксис:

reference ksCentreMarker (CentreParam * par);

Выходной параметр:

par	- указатель на структуру параметров обозначения центра CentreParam.

Возвращаемое значение:

указатель на обозначение центра	- в случае удачного завершения,
0	- в случае неудачи.

ksCreateQualityIterator – Создать итератор по квалитетам

Аналог данной функции при использовании Automation - метод ksIterator::ksCreateQualityIterator

Синтаксис:

reference ksCreateQualityIterator (unsigned char system,

unsigned char withLimitation);

Входные параметры:

system	- система допуска 1 - система отверстия, 0 - система вала,
withLimitation	- признак учета ограничений, наложенных в системе: 0 - без учёта ограничений, 1 - с учётом ограничений.

Возвращаемое значение:

указатель на итератор	- в случае успешного завершения,
0	- в случае неудачи.

ksExecQualityDialog – Вызов диалога Выбор квалитета

Аналог данной функции при использовании Automation - метод KompasObject::ksExecQualityDialog .

Синтаксис:

int LIB_FUNC ksExecQualityDialog(void *HWindow, char * curQual, double * dimValue,
short inMM, QualityContensParam * param);

Входные параметры:

HWindow - дескриптор окна,
curQual - текущее значение квалитета,
dimValue - значение для которого требуется выбрать квалитет или NULL,
inMM - размерность параметров dimValue, high, low :
1 - миллиметры,
0 - единицы измерения текущего документа.

Выходной параметр:

param - указатель на структуру параметров квалитета QualityContensParam.

Возвращаемое значение:

1 - в случае выбора квалитета,
0 - в случае отмены выбора или ошибки.

ksExecQualityDialogW – Вызов диалога Выбор квалитета (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksExecQualityDialog .

Синтаксис:

int LIB_FUNC ksExecQualityDialogW(void *HWindow, LPWSTR curQual, double * dimValue,
short inMM, QualityContensParamW * param);

Входные параметры:

HWindow - дескриптор окна,
curQual - текущее значение квалитета,
dimValue - значение, для которого требуется выбрать квалитет или NULL,
inMM - размерность параметров dimValue, high, low :
1 - миллиметры,
0 - единицы измерения текущего документа.

Выходной параметр:

param - указатель на структуру параметров квалитета QualityContensParamW.

Возвращаемое значение:

1 - в случае выбора квалитета,
0 - в случае отмены выбора или ошибки.

ksGetLeaderShelfLength – Получить длину полки линии-выноски и координаты ее конечной точки в системе координат вида

Справка системы КОМПАС...

`KOMPAS.chm::/CM_LEADER.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetLeaderShelfLength.

Синтаксис:

double ksGetLeaderShelfLength (long leader, double *x, double *y);

Входные параметры:

leader - указатель на линию-выноску.

Выходные параметры:

x - координата конечной точки полки линии выноски по оси X,
y - координата конечной точки полки линии выноски по оси Y.

ksGetShelfPoint – Получить координаты начала и конца выносной полки и ножки

Справка системы КОМПАС...

`KOMPAS.chm::/CM_LEADER.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetShelfPoint.

Синтаксис:

int LIB_FUNC ksGetShelfPoint(reference p, int index, double *x, double *y, int paramType);

double ksGetLeaderShelfLength (long leader, double *x, double *y);

Входные параметры:

p - указатель на объект (размер или линию-выноску),
index - индекс точки.
Значения индекса:
Для линии с ножкой:

```
        1 __ 2
        /
    0 /
```

Для линии без ножки (с выносной полкой на продолжении выносной линии):

```
    0 __ 1
```

Точка 0 примыкает к выносной линии;
paramType - тип параметров, задает систему координат:
ALLPARAM - в системе координат владельца;
SHEET_ALLPARAM - в системе координат листа;
VIEW_ALLPARAM - в системе координат вида.

Выходные параметры:

x, y - координаты точки.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае неудачи.

ksGetQualityContensParam – Получить параметры квалитета

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksGetQualityContensParam.

Синтаксис:

int ksGetQualityContensParam (const char* name,
QualityContensParam* param,
unsigned char inMM);

Входные параметры:

inMM - размерность параметров minLimit, maxLimit, high, low :
1 - миллиметры,
0 - единицы измерения текущего документа.

Выходной параметр:

param - указатель на структуру параметров квалитета QualityContensParam.
name - поле допуска.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksGetQualityContensParamW.

ksGetQualityContensParamW – Получить параметры квалитета (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetQualityContensParam.

Синтаксис:

int LIB_FUNC ksGetQualityContensParamW (LPWSTR* name,
QualityContensParamW* param,
unsigned char inMM);

Входные параметры:

inMM - размерность параметров minLimit, maxLimit, high, low :
1 - миллиметры,
0 - единицы измерения текущего документа.

Выходной параметр:

param - указатель на структуру параметров квалитета QualityContensParam.
name - поле допуска.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksGetQualityContensParam.

ksGetQualityDefects – Получить отклонения

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetQualityDefects.

Синтаксис:

int ksGetQualityDefects (const char *name,
double dimValue,
double *high,
double *low,
unsigned char inMM);

Входные параметры:

dimValue - значение размера,
inMM - размерность параметров dimValue, high, low:
1 - миллиметры,
0 - единицы измерения текущего документа.

Выходные параметры:

name - поле допуска, (например, "H7"),
high - верхнее отклонение,
low - нижнее отклонение.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksGetQualityDefectsW.

ksGetQualityDefectsW – Получить отклонения (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/185_23_4_Vybor_kvaliteta.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetQualityDefects.

Синтаксис:

int LIB_FUNC ksGetQualityDefectsW (const LPWSTRname,
double dimValue,
double *high,
double *low,
unsigned char inMM);

Входные параметры:

dimValue	- значение размера,
inMM	- размерность параметров dimValue, high, low:
	1 - миллиметры,
	0 - единицы измерения текущего документа.

Выходные параметры:

name	- поле допуска, (например, "H7"),
high	- верхнее отклонение,
low	- нижнее отклонение.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksGetQualityDefects.

ksMoveQualityIterator– Двигаться по квалитетам

Аналог данной функции при использовании Automation - метод ksIterator::ksMoveQualityIterator.

Синтаксис:

int ksMoveQualityIterator (reference iterator,

QualityContensParam* param,

unsigned char inMM,

unsigned char ch);

Входные параметры:

inMM	- размерность параметров minLimit, maxLimit, high, low: 1 - миллиметры, 0 - единицы измерения текущего документа.
ch	- направление перемещения итератора: F - первый квалитет, N - следующий квалитет.

Выходные параметры:

param	- указатель на структуру параметров квалитета StructQualityContensParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksMoveQualityIteratorW – Двигаться по квалитетам (Unicode)

Аналог данной функции при использовании Automation - метод ksIterator::ksMoveQualityIterator.

Синтаксис:

int LIB_FUNC ksMoveQualityIteratorW (reference iterator,

QualityContensParamW* param,

unsigned char inMM,

unsigned char ch);

Входные параметры:

inMM	- размерность параметров minLimit, maxLimit, high, low : 1 - миллиметры, 0 - единицы измерения текущего документа.
ch	- направление перемещения итератора: F - первый квалитет, N - следующий квалитет.

Выходные параметры:

param	- указатель на структуру параметров квалитета StructQualityContensParam.

Возвращаемое значение:

1	- в случае успешного завершения,

0 - в случае неудачи.

ksOrdinatedDimension – Проставить размер высоты

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ORDINATE_DIM.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOrdinatedDimension.

Синтаксис:

reference ksOrdinatedDimension (OrdinatedDimParam *ordPar);

Выходной параметр:

ordPar - указатель на структуру параметров размера высоты OrdinatedDimParam.

Возвращаемое значение:

указатель на размер высоты - в случае удачного завершения,
0 - в случае неудачи.

ksRemoteElement – Создать объект "Выносной элемент"

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRemoteElement.

Синтаксис:

reference ksRemoteElement (RemoteElementParam * par);

Входные параметры:

par - структура параметров выносного элемента RemoteElementParam,

Возвращаемое значение:

указатель на объект "Выносной элемент" - в случае успешного завершения,

Примечание:

Создается новый объект "Выносной элемент" в текущем виде и в текущем документе.

ksSetMaterialParam – Установить параметры материала в чертеже

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMaterialParam.

Синтаксис:

int ksSetMaterialParam (reference material, double density);

Входные параметры:

material	- наименование материала, динамический массив строк текста (TEXT_LINE_ARR),
density	- плотность (г/куб. мм).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSpecRough – Проставить знак неуказанной шероховатости в чертеже

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_ROUGH.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSpecRough.

Синтаксис:

reference LIB_FUNK ksSpecRough (SpecRoughParam * par);

Выходной параметр:

par	- указатель на структуру параметров знака неуказанной шероховатости SpecRoughParam.

Возвращаемое значение:

указатель на знак неуказанной шероховатости	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksSpecRoughW.

ksSpecRoughW – Проставить знак неуказанной шероховатости в чертеже (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSpecRough.

Синтаксис:

reference LIB_FUNK ksSpecRoughW (SpecRoughParamW * par);

Выходной параметр:

par	- указатель на структуру параметров знака неуказанной шероховатости SpecRoughParam.

Возвращаемое значение:

указатель на знак неуказанной шероховатости	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksSpecRough.

ksTolerance – Проставить обозначение допуска формы

Справка системы КОМПАС...

KOMPAS.chm::/CM_FORMTOLERANCE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTolerance.

Синтаксис:

int ksTolerance (ksToleranceParam *par);

Выходной параметр:

par	- указатель на структуру параметров обозначения допуска формы ksTolerancePar.

Возвращаемое значение:

указатель на допуск формы	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Допуск формы является составным объектом, состоящим из таблицы (в общем случае 10x10) и двух опор. Каждая ячейка таблицы имеет свой номер. Нумерация идет слева направо и сверху вниз.
2. Номер ячейки задается функцией ColumnNumber, а текст ячейки определяется функцией TextLine (первые два параметра - SPECIAL_SYMBOL и SPECIAL);

Leader – Создать линию-выноску

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_LEADER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLeader.

Синтаксис:

reference Leader (LeaderParam *leaderPar);

Выходной параметр:

leaderPar	- указатель на структуру параметров линии-выноски LeaderParam.

Возвращаемое значение:

указатель на линию-выноску	- в случае удачного завершения,
0	- в случае неудачи.

LinDimension – Проставить линейный размер

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_DIML.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLinDimension.

Синтаксис:

reference LinDimension (LDimParam *Par);

Выходной параметр:

Par	- указатель на структуру параметров линейного размера LdimParam.

Возвращаемое значение:

указатель на линейный размер	- в случае удачного завершения,
0	- в случае неудачи.

LinBreakDimension – Проставить линейный размер с обрывом

Справка системы КОМПАС...

KOMPAS.chm::/CM_CUT_DIML.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLinBreakDimension.

Синтаксис:

reference LinBreakDimension (LBreakDimParam *linPar);

Выходной параметр:

linPar	- указатель на структуру параметров линейного размера с обрывом LBreakDimParam.

Возвращаемое значение:

указатель на линейный размер с обрывом	- в случае удачного завершения,
0	- в случае неудачи.

MarkerLeader – Создать линию-выноску для обозначения маркировки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/obozn_priemy_raboty.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMarkerLeader.

Синтаксис:

reference MarkerLeader (MarkerLeaderParam *leaderPar);

Выходной параметр:

leaderParam	- указатель на структуру параметров линии-выноски для обозначения маркировки MarkerLeaderParam.

Возвращаемое значение:

указатель на линию-выноску для обозначения маркирования	- в случае удачного завершения,

0 - в случае неудачи.

OpenTechnicalDemand – Открыть технические требования

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/413_Glava47_Tekhnicheskie_trebo.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenTechnicalDemand

Синтаксис:

int OpenTechnicalDemand (reference pGab);

Входные параметры:

pGab - указатель на динамический массив параметров листов технических требований типа RECT_ARR.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Следующие далее функции до вызова CloseTechnicalDemand определяют текст технических требований. Параметр pGab определяет массив габаритных прямоугольников страниц технических требований неопределенной длины. Если pGab = 0, то технические требования размещаются на одной странице автоматически.

PositionLeader – Создать позиционную линию-выноску

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_POSITIONLEADER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPositionLeader.

Синтаксис:

reference PositionLeader(PosLeaderParam *leaderPar);

Выходной параметр:

posLeaderParam - указатель на структуру параметров позиционной линии-выноски PosLeaderParam.

Возвращаемое значение:

указатель на позиционную линию-выноску	- в случае удачного завершения,
0	- в случае неудачи.

RadDimension – Проставить радиальный размер

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_DIMR.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRadDimension.

Синтаксис:

reference RadDimension (RDimParam *Par);

Выходной параметр:

Par	- указатель на структуру параметров радиального размера RDimParam.

Возвращаемое значение:

указатель на радиальный размер	- в случае удачного завершения,
0	- в случае неудачи.

RadBreakDimension – Проставить радиальный размер с изломом

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_DIMR_WITH_BREAK.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRadBreakDimension.

Синтаксис:

reference RadBreakDimension (RBreakDimParam *Par);

Выходной параметр:

Par	- указатель на структуру параметров радиального размера с изломом RBreakDimParam.

Возвращаемое значение:

указатель на радиальный размер с изломом - в случае удачного завершения,
0 - в случае неудачи.

Rough – Проставить обозначение шероховатости

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRough.

Синтаксис:

reference Rough (RoughParam *roughPar);

Выходной параметр:

roughPar - указатель на структуру параметров обозначения шероховатости...

Возвращаемое значение:

указатель на обозначение шероховатости - в случае удачного завершения,
0 - в случае неудачи.

SpecRough – Задать шероховатость неуказанных поверхностей

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_ROUGH.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSpecRough.

Синтаксис:

int SpecRough (unsigned char type, unsigned char t, char * s);

Входные параметры:

type - тип знака
0 - вид обработки не устанавливается
1 - обработка удалением слоя материала
2 - обработка без удаления слоя материала
t - наличие знака в скобках (1 - есть, 0 - нет).

Выходные параметры:

s - строка текста.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Tolerance – Проставить обозначение допуска формы

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_FORMTOLERANCE.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTolerance.

Синтаксис:

int Tolerance (ToleranceParam *par);

Выходной параметр:

par	- указатель на структуру параметров обозначения допуска формы ToleranceParam.

Возвращаемое значение:

указатель на допуск формы	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Допуск формы является составным объектом, состоящим из таблицы (в общем случае 10x10) и двух опор. Каждая ячейка таблицы имеет свой номер. Нумерация идет слева направо и сверху вниз.
2. Номер ячейки задается функцией ColumnNumber, а текст ячейки определяется функцией TextLine (первые два параметра - SPECIAL_SYMBOL и SPECIAL);

ViewPointer – Создать стрелку направления взгляда

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_VIEWPOINTER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksViewPointer.

Синтаксис:

reference ViewPointer (ViewPointerParam *par);

Выходной параметр:

par - указатель на структуру параметров стрелки направления взгляда ViewPointerParam.

Возвращаемое значение:

указатель на стрелку взгляда - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ViewPointerW.

ViewPointerW – Создать стрелку направления взгляда

Справка системы КОМПАС...

KOMPAS.chm::/CM_VIEWPOINTER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksViewPointer.

Синтаксис:

reference ViewPointerW (ViewPointerParamW *par);

Выходной параметр:

par - указатель на структуру параметров стрелки направления взгляда ViewPointerParam.

Возвращаемое значение:

указатель на стрелку взгляда - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ViewPointer.

Работа с группами объектов

Функции работы с группами объектов

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Операции редактирования можно выполнять с выделенными объектами. После выделения их можно объединить в отдельную группу для дальнейшего использования. Функции данного раздела позволяют выделить объекты, объединить их в группу на текущий сеанс работы или с последующей записью в документ, а также обрабатывать имеющиеся в документе группы объектов.

AddObjGroup – Добавить объект в группу

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAddObjGroup.

Синтаксис:

int AddObjGroup (reference group, reference obj);

Входные параметры:

group - указатель на группу,
obj - указатель на добавляемый объект.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то добавление производится в группу выделения (т.е. происходит выделение объекта).

ClearGroup – Очистить группу объектов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksClearGroup.

Синтаксис:

int ClearGroup (reference group);

Входные параметры:

group - указатель на очищаемую группу
(0 - очищается группа выделения),

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Описание:

1. Все объекты исключаются из группы. Если группа создавалась как временная, то при этом все ее объекты автоматически удаляются.
2. Если указатель group равен нулю, то очистится группа выделения (снимется текущее выделение объектов).

EndGroup – Завершить создание группы объектов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEndGroup.

Синтаксис:

void EndGroup (void);

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Завершить создание группы объектов, начатое функцией NewGroup.

ExcludeObjGroup – Исключить объект из группы

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksExcludeObjGroup.

Синтаксис:

int ExcludeObjGroup (reference group, reference obj);

Входные параметры:

group	- указатель на группу,
obj	- указатель на исключаемый объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то исключение производится из группы выделения (т.е. снимается выделение объекта чертежа). Если объект создавался в режиме формирования временной группы, то при исключении он автоматически удаляется.

ExistGroupObj – Проверить группу на наличие объектов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksExistGroupObj.

Синтаксис:

int ExistGroupObj (reference group);

Входной параметр:

group	- указатель на группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель группы равен нулю, то проверяется группа выделения (есть ли выделенные объекты документа).

GetGroup – Получить указатель на группу по ее имени

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetGroup.

Синтаксис:

reference GetGroup (char *name);

Входной параметр:

name	- имя группы.

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию GetGroupW.

GetGroupW – Получить указатель на группу по ее имени (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetGroup.

Синтаксис:

reference LIB_FUNC GetGroupW (LPWSTR name);

Входной параметр:

name	- имя группы.

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию GetGroup.

ksClearGroup – Очистить группу

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksClearGroup.

Синтаксис:

int ksClearGroup (reference g,

unsigned char deleteTmp);

Входные параметры:

g	- указатель на очищаемую группу (0 - очищается группа выделения),
deleteTmp	- признак удаления временных элементов: 1 - временные элементы удаляются, 0 - временные элементы сохраняются.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksGetGroupName – Получить имя группы по указателю на группу

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetGroupName.

Синтаксис:

int ksGetGroupName (reference gr,
char *name,
int size);

Входные параметры:

gr	- указатель на группу,
name	- указатель строки для имени группы,
size	- размер строки.

Выходной параметр:

-1	- у группы есть имя, но размер size меньше требуемой длины имени группы (имя не передается),
0	- ошибка указания группы,
1	- именная группа (имя есть),
2	- рабочая группа (имени нет).

Возвращаемое значение:

строка с именем группы.

Примечание.

При использовании Unicode следует использовать функцию ksGetGroupNameW.

ksGetGroupNameW – Получить имя группы по указателю на группу (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetGroupName.

Синтаксис:

int LIB_FUNC ksGetGroupNameW (reference gr,
LPWSTR name,
int size);

Входные параметры:

gr	- указатель на группу,
name	- указатель строки для имени группы,
size	- размер строки.

Выходной параметр:

-1	- у группы есть имя, но размер size меньше требуемой длины имени группы (имя не передается),
0	- ошибка указания группы,
1	- именная группа (имя есть),
2	- рабочая группа (имени нет).

Возвращаемое значение:

строка с именем группы.

Примечание.

При использовании ANSI следует использовать функцию ksGetGroupName.

ksMakeEncloseContoursEx – Получить группу объектов, охватывающих заданную точку

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMakeEncloseContoursEx.

Синтаксис:

reference ksMakeEncloseContoursEx(reference gr,
double x,
double y,
int forHatch);

Входные параметры:

gr	- указатель на временную группу или 0, если требуются контуры в текущем виде,
x, y	- координаты точки внутри охватываемых контуров,
forHatch	- признак создания контура для штриховки.

Возвращаемое значение:

указатель на временную группу контуров	- в случае удачного завершения,
0	- если контуров не нашлось или в случае неудачи.

Примечание:

forHatch == 1 - При создании контуров для штриховки используются объекты со стилем основная и утолщенная, объекты других стилей не учитываются.

forHatch == 0 - При создании контура с использованием любых стилей линий.

ksViewGetObjectArea – Получить группу графических объектов, определяющих область выделения, используя визуальный процесс

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksViewGetObjectArea.

Синтаксис:

reference ksViewGetObjectArea();

Возвращаемое значение:

указатель на временную группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Так как система может работать только с одним процессом, нужно завершить другие процессные функции: Cursor, Placement, CommandWindow, ksEditViewObject.

NewGroup – Создать новую группу объектов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksNewGroup.

Синтаксис:

reference NewGroup (int type);

Входной параметр:

type	- тип группы: 0 - модельная, 1 - временная.

Возвращаемое значение:

указатель на группу	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. В модельной группе лежат объекты, которые уже созданы в документе. Во временной группе могут лежать временные и существующие объекты.
2. Если для временной группы не будет вызван функция StoreTmpGroup, то временные объекты группы будут уничтожены по окончании работы библиотеки.
3. Создаваемые в дальнейшем до вызова функции EndGroup объекты чертежа записываются в модель или во временный список объектов (например, для фантомной прорисовки при вводе).
4. Для дальнейшей обработки группы используются те же функции редактирования, что и для отдельных объектов, так как указатель на группу ничем не отличается от указателя на отдельный объект (оба они имеют тип reference).
5. Группа может объединять объекты вида, виды и слои.
6. При создании новой группы необязательно закрывать предыдущую (т.е. поддерживается вложенность групп).

SaveGroup – Сохранить группу объектов в документе

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSaveGroup.

Синтаксис:

int SaveGroup (reference group, char *name);

Входные параметры:

group - указатель на группу,
name - имя сохраняемой группы.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Данный метод действителен только для модельной группы. После его выполнения группа автоматически сохраняется в чертеже при его записи. В противном случае группа действительна только в текущем сеансе работы. Если указатель группы равен нулю, то сохраняется группа селектирования (выделенные объекты документа).
2. При использовании Unicode следует использовать функцию SaveGroupW.

SaveGroupW – Сохранить группу объектов в документе (Unicode)

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_STORE_GROUP.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSaveGroup.

Синтаксис:

int LIB_FUNC SaveGroup (reference group, LPWSTR name);

Входные параметры:

group - указатель на группу,
name - имя сохраняемой группы.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Данный метод действителен только для модельной группы. После его выполнения группа автоматически сохраняется в чертеже при его записи. В противном случае группа действительна только в текущем сеансе работы. Если указатель группы равен нулю, то сохраняется группа селектирования (выделенные объекты документа).
2. При использовании ANSI следует использовать функцию SaveGroup.

SelectGroup – Автоматически сформировать группу объектов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Команды выделения объектов рамкой...

KOMPAS.chm::/99_8_5_2_Vydelenie_obwektov_s_p.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSelectGroup.

Синтаксис:

int SelectGroup (reference group, int selectMode, double x1, double y1, double x2, double y2);

Входные параметры:

group	- указатель на группу,
selectMode	- тип выделения: 1 - объекты внутри прямоугольника-"ловушки", 2 - объекты снаружи прямоугольника-"ловушки", 3 - объекты, целиком или частично попавшие в прямоугольник-"ловушку".
x1, y1	- координаты левой нижней вершины прямоугольника-"ловушки",
x2, y2	- координаты правой верхней вершины прямоугольника-"ловушки".

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель на группу равен нулю, то добавление производится в группу выделения (т.е. происходит выделение объектов документа).

StoreTmpGroup – Вставить временную группу в документ (группа "рассыпается")

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_STORE_GROUP.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksStoreTmpGroup.

Синтаксис:

int StoreTmpGroup (reference g);

Входной параметр:

g	- указатель на группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Временные объекты не записываются в документ и на экране не отображаются. Они сохраняются только до конца работы создающей их библиотечной команды. Временная группа чаще всего служит для отрисовки фантомного изображения и используется в качестве параметров в функциях Placement и Cursor. Функция StoreTmpGroup записывает все объекты временной группы в модель (т.е. фиксирует ее). Сама же группа получает статус постоянной.

Навигация по графическому документу

Функции навигации

Функции данного раздела позволяют перемещаться по объектам графического документа.

CreateIterator – Создать блок параметров навигации по объектам

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksCreateIterator

Синтаксис:

reference CreateIterator(int searchType, reference parent);

Входные параметры:

searchType	- тип поиска объекта
parent	- указатель на объект (для движения по группе, внутри макроэлемента, по слою).

Типы объектов и интерфейсы...

Возвращаемое значение:

указатель на найденный объект	- в случае успешного завершения,
0	- в случае неудачи.

Описание:

1. Передвижение по модели документа (навигация) производится в соответствии с условиями, заданными в специальном блоке параметров. Он содержит тип движения, определяющий режим перемещения (например, по видам, слоям, всем объектам, объектам заданного типа и т.д.) и указатель комплексного объекта (макроэлемента, слоя, группы) при перемещении по составляющим его объектам. Итератор привязан к конкретному режиму графического редактора (например, документу, виду), поэтому Вы не сможете использовать один и тот же итератор для навигации в разных видах, штампах и т.п. Итератор сохраняет свое действия до окончания сеанса работы с библиотекой. При переходе внутри библиотечной функции под управление КОМПАС-ГРАФИК, после возврата значение всех итераторов будет сброшено.
2. Итератор работает с атрибутами активного документа.

   В случае движения по всем объектам или при движении по объектам определенного типа, можно определить порядок выдачи не только в порядке создания объектов как было прежде, но и в порядке отрисовки.

   Для этого searchType нужно задать с минусом.

   Например:

   MACRO_OBJ - движение по макроэлементам в порядке создания;

   -MACRO_OBJ - движение по макроэлементам в порядке отрисовки.

   Для движения по всем объектам в порядке отрисовки в таблице Типы графических документов служит константа ALL_OBJ_SHOW_ORDER -1000 - все объекты, которые могут входить в вид, в порядке отрисовки.

   Например:

   ALL_OBJ - движение по всем объектам в порядке создания.

   ALL_OBJ_SHOW_ORDER - движение по всем объектам в порядке отрисовки.

DeleteIterator – Удалить блок параметров навигации по объектам

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksDeleteIterator.

Синтаксис:

int DeleteIterator (reference iterator);

Входной параметр:

iterator - указатель на итератор.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

FindObj – Найти ближайший к заданной точке объект вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksFindObj.

Синтаксис:

reference FindObj (double x,
double y,
double dist);

Входные параметры:

x, y	- координаты точки,
dist	- размер стороны квадрата-"ловушки" с центром в указанной точке.

Возвращаемое значение:

указатель на найденный объект вида	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Зона поиска - квадрат с указанной стороной и центром.
2. Если значение dist равно MAXDOUBLE, выполняется поиск ближайшего к указанной точке объекта без ограничения зоны поиска.
3. Функцию FindObj можно использовать для поиска объектов внутри макроэлемента.

   Для этого макроэлемент должен быть открыт на редактирование функцией ksOpenMacro.

   Пример:

```
ksOpenMacro(obj);
  long obj_i=FindObj(x, y, AXIS_FIND_SENS);
  if( ExistObj(obj_i) )
     LightObj(obj_i, 1);
EndObj();
```

   При этом, если для макроэлемента задана система координат (см. SetMacroPlacement), то координаты в FindObj должны передаваться в системе координат макроэлемента.

   Пересчитать координаты в систему координат макроэлемента можно через функцию ksPointIntoMtr. Для этого нужно функцией ksMtr создать матрицу преобразования координат.

   Пример:

```
ksMtr(xp, yp, ang, 1,1); // xp, yp, ang, scaleX, scaleY, - параметры Placement макроэлемента.
  ksPointIntoMtr (x, y, &xn, &yn);
```

EndMtr();

GetObjGabaritRect – Получить габаритный прямоугольник объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjGabaritRect.

Синтаксис:

int GetObjGabaritRect (reference p, RectParam *par);

Входные параметры:

p	- указатель на объект,
par	- указатель на структуру размеров габаритного прямоугольника RectParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Габаритный прямоугольник возвращается в координатах листа.

GetViewObjCount – Получить количество объектов вида

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetViewObjCount.

Синтаксис:

long GetViewObjCount (reference p);

Входной параметр:

p	- указатель на вид.

Возвращаемое значение:

количество объектов в виде	- в случае удачного завершения,
-1	- в случае неудачи.

Примечание:

В случае, если p=0, возвращается число объектов текущего вида или фрагмента.

MoveIterator – Переместить итератор (позиционироваться на объекте)

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksMoveIterator

Синтаксис:

reference MoveIterator (reference iterator,

unsigned char type);

Входной параметр:

iterator	- указатель на итератор,
type	- направление перемещения итератора:
	F - первый объект,
	N - следующий объект.

Возвращаемое значение:

указатель на объект	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Функция перемещает текущий указатель итератора (блока параметров навигации) iterator на очередной объект модели. Режим позиционирования (тип объектов, участвующих в поиске) определяются при создании итератора.

При позиционировании на именованную группу она автоматически становится текущей (рабочей). К ней будут относиться операции добавления и исключения объектов (при исключении всех объектов из именованной группы она автоматически удаляется).

KeepReference – Запретить удалять временный объект после завершения команды библиотеки

Синтаксис:

int LIB_FUNC KeepReference(reference r);

Входной параметр:

r	- указатель на графический объект или группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ReleaseReference – Освободить указатель на объект

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksReleaseReference.

Синтаксис:

int ReleaseReference (reference p);

Входной параметр:

p - указатель на объект.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Не временный объект не уничтожается. В результате таблица объектов сокращается, а операции, связанные с перебором объектов, ускоряются.

ksShowHideTmpObj – Скрыть /Показать временный объект в документе

Синтаксис:

```
int    LIB_FUNC ksShowHideTmpObj( reference ref,
                          int      show );
```

Входные параметры:

ref - указатель на объект,
show -значение из перечисления ksShowHideTmpObjTypeEnum
ksTmpObjHide - скрыть объект,
ksTmpObjShow - показать временный объект как обычный объект,
ksTmpObjShowPhantom - показать временный объект как фантом.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksSetLightObjType – Установить тип подсветки объекта (light=1 – красный) или (light=0 – зеленый)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetLightObjType.

Синтаксис:

int LIB_FUNC ksSetLightObjType(reference ref,
unsigned char light);

Входные параметры:

ref - указатель на объект,
light - light - (light=1 - красный) или (light=0 - зеленый).

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksGetObjectsNameByType – Вернуть имя объекта по его типу. Множественное число

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjectsNameByType.

Синтаксис:

int LIB_FUNC ksGetObjectsNameByType(int type, char * name, unsigned int bufLen);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum,
bufLen - размер буфера name в символах.

Выходные параметры:

name - Строковое имя типа объекта во множественном числе - отрезки, дуги и т.д,

name - Строковое имя типа объекта во множественном числе - отрезки, дуги и т.д.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksGetObjectsNameByTypeW – Вернуть имя объекта по его типу. Множественное число. Unicode

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjectsNameByType.

Синтаксис:

int LIB_FUNC ksGetObjectsNameByTypeW(int type, LPWSTR * name, unsigned int bufLen);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum,
bufLen - размер буфера name в символах.

Выходные параметры:

name - Строковое имя типа объекта во множественном числе - отрезки, дуги и т.д,

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksGetObjectNameByType – Вернуть имя объекта по его типу

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjectsNameByType.

Синтаксис:

int LIB_FUNC ksGetObjectNameByType(int type, char * name, unsigned int bufLen);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum,
bufLen - размер буфера name в символах.

Выходные параметры:

name - Строковое имя типа объекта во множественном числе - отрезки, дуги и т.д,

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

ksGetObjectNameByTypeW – Вернуть имя объекта по его типу. Unicode

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjectsNameByType.

Синтаксис:

int LIB_FUNC ksGetObjectNameByType(int type, LPWSTR * name, unsigned int bufLen);

Входные параметры:

type - тип объекта из перечисления DrawingObjectTypeEnum,
bufLen - размер буфера name в символах.

Выходные параметры:

name - Строковое имя типа объекта во множественном числе - отрезки, дуги и т.д,

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Функции работы с параметрическими переменными и связями

Функции параметризации и ограничений

Функции данного раздела позволяют управлять параметрическими связями и ограничениями.

ksDestroyObjConstraint – Удалить параметрическую связь или ограничение, наложенные на указанный объект

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDestroyObjConstraint.

Синтаксис:

int ksDestroyObjConstraint (reference obj,

ConstraintParam *par);

Входные параметры:

obj - указатель на объект,
par - указатель на структуру параметрической связи или ограничения ConstraintParam...

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetDimensionVariableName – Получить имя параметрической переменной, связанной с размером

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetDimensionVariableName.

Синтаксис:

int ksGetDimensionVariableName (reference dimObj,

char * varName,

int size);

Входной параметр:

dimObj - указатель на размер.

Выходной параметр:

varName - имя переменной, если размеру поставлена в соответствие переменная,
size - размер буфера, выделенного под имя переменной,
пустая строка - если размеру не поставлена в соответствие переменная
или в случае неудачи.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

ksGetDocVariableArray – Получить массив параметрических переменных графического документа или вставки фрагмента

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetDocVariableArray.

Синтаксис:

reference ksGetDocVariableArray (reference p);

Входной параметр:

p - указатель на документ или вставку фрагмента.

Возвращаемое значение:

указатель на динамический массив структур VariableParam - в случае успеха,
0 - в случае неудачи.

ksGetObjConstraints – Получить параметрические связи и ограничения, наложенные на указанный объект

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/476_Glava55_Nalozhenie_svjazej_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjConstraints.

Синтаксис:

reference ksGetObjConstraints (reference obj);

Входной параметр:

obj - указатель на объект,

Возвращаемое значение:

указатель на структуру параметров связей и ограничений ConstraintParam типа CONSTRAINT_ARR. - в случае успеха,
0 - в случае неудачи.

ksParametrizeObjects – Параметризовать группу объектов

Аналог данной функции при использовании Automation - метод ksDocument2D::ksParametrizeObjects.

Синтаксис:

int ksParametrizeObjects (reference group, IParametrizeParam* param);

Входные параметры:

group - указатель на группу объектов,
param - параметры параметризации объектов.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если group = 0, параметризуется группа выделенных объектов.
2. Допуск на совпадение точек должен лежать в диапазоне [0...10] мм.
3. Угловой допуск должен лежать в диапазоне [0...10] градусов.
4. Получить интерфейс параметров параметризации можно при помощи функции ksGetParametrizationParam.

ksSetDocVariableArray – Заменить значения параметрических переменных

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/495_59_3_2_Izmenenie_znachenij_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetDocVariableArray.

Синтаксис:

int ksSetDocVariableArray (reference obj,

reference arr,

unsigned char setNote);

Входные параметры:

obj	- указатель на документ или вставку фрагмента,
arr	- указатель на динамический массив VARIABLE_ARR,
setNote	признак редактирования комментариев:
	0 - комментарии не менять,
	1 - комментарии менять.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksSetObjConstraint – Установить параметрическую связь или ограничение

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/476_Glava55_Nalozhenie_svjazej_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetObjConstraint.

Синтаксис:

int ksSetObjConstraint (reference obj,

ConstraintParam *par);

Входные параметры:

obj	- объект, на который накладывается связь или ограничение,
par	- указатель на структуру параметрической связи или ограничения ConstraintParam....

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Редактирование графических объектов

Функции редактирования объектов чертежа

Функции данного радела обеспечивают операции редактирования изображения чертежа (сдвиг, поворот, масштабирование, зеркальное отображение, удаление, копирование в точку, по линии или по узлам сетки, деформацию, запись в отдельный фрагмент), а также обработку структуры объектов и работу с графическим буфером.

CopyGroupToDocument – Скопировать группу в документ

Синтаксис:

reference LIB_FUNC CopyGroupToDocument (reference gr,

reference from,

reference to)

Входные параметры:

From	- указатель на документ, откуда копируется группа,
to	- указатель на документ, куда копируется группа,
gr	- указатель на копию группы.

Возвращает указатель на группу в документе to.

CopyObj – Копировать объект

Пример...

Синтаксис:

int CopyObj (reference obj,

double xb,

double yb,

double xn,

double yn,

double scale,

double ang);

Входные параметры:

obj	- указатель на объект,
xb, yb	- координаты базовой точки объекта,
xn, yn	- координаты нового положения базовой точки,
scale	- масштаб,
ang	- угол поворота в градусах.

Возвращаемое значение:

указатель на получившийся объект или группу объектов - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Если указатель на объект равен нулю, то копируются выделенные объекты документа.
2. При копировании одиночного объекта новый объект создается на текущем слое текущего вида.
3. При копировании группы объектов слой сохраняется.
4. При копировании одиночного многослойного макрообъекта он перестает быть многослойным. Макро и входящие в него объекты переносятся на текущий слой. Для копирования многослойного макро с сохранением многослойности требуется добавить его в группу.
5. Функция устарела, рекомендуется вместо нее использовать функцию ksCopyObj.

CursorEx – Запрос к системе на получение точки.

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCursorEx.

Синтаксис:

```
int CursorEx (RequestInfo *info,
double *x,
double *y,
void *phantom,
LPUNKNOWN processParam);
```

Входные параметры:

info - указатель на структуру параметров запроса к системе RequestInfo,
x, y - координаты введенной точки,
phantom - указатель на структуру управления фантомом, определяющую тип движения курсора (аналог типа резиновой нити в версии 4),
processParam - указатель на интерфейс параметров процесса IProcessParam.

Выходные параметры:

x, y - возвращаемые координаты точки.

Возвращаемое значение:

-1 - если указана точка,
0 - отказ (Esc), идентификатор выбранной команды из командной строки или меню, определенный в файле ресурсов.

Примечание:

1. В качестве функции обратной связи передается указатель на функцию типа CursorCallBack. Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.
2. При использовании Unicode следует использовать функцию CursorExW.

CursorExW – Запрос к системе на получение точки (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCursorEx.

Синтаксис:

```
int LIB_FUNC CursorExW (RequestInfoW *info,
double *x,
double *y,
void *phantom,
LPUNKNOWN processParam);
```

Входные параметры:

info	- указатель на структуру параметров запроса к системе RequestInfo,
x, y	- координаты введенной точки,
phantom	- указатель на структуру управления фантомом, определяющую тип движения курсора (аналог типа резиновой нити в версии 4),
processParam	- указатель на интерфейс параметров процесса IProcessParam.

Выходные параметры:

x, y	- возвращаемые координаты точки.

Возвращаемое значение:

-1	- если указана точка,
0	- отказ (Esc), идентификатор выбранной команды из командной строки или меню, определенный в файле ресурсов.

Примечание:

1. В качестве функции обратной связи передается указатель на функцию типа CursorCallBack. Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствую-

щее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.

2. При использовании ANSI следует использовать функцию CursorEx.

DecomposeObj – Разбить объект на составляющие части – отрезки, дуги, тексты

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_DESTROYMACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDecomposeObj.

Синтаксис:

reference DecomposeObj (reference p,

unsigned char level,

double arrow,

unsigned char type);

Входные параметры:

p - указатель на разбиваемый объект,
level - степень детализации разбиения:
0 - отрезки, дуги, тексты, точки,
1 - отрезки, тексты, точки,
2 - отрезки, дуги, тексты,
4 - отрезки, дуги, точки,
5 - отрезки, дуги, тексты, заливки стрелок и треугольников баз,
6 - разбиение объектов из ассоциативного чертежа на составляющие с учетом видимых и невидимых участков,
arrow - размер "стрелки прогиба",
type - признак выбранной системы координат:
0 - разбиение объекта в СК вида,
1 - разбиение объекта в СК листа.

Возвращаемое значение:

указатель на временную группу компонентов сложного объекта - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

1. Метод используется при разработке различных конверторов, преобразующих информацию из системы КОМПАС во внешние форматы.
2. Текущий документ должен быть графическим.

3. Графический документ разбивается по частям. Такими частями могут быть объекты вида, основная надпись, технические требования, спецификация на листе, знак неуказанной шероховатости.
4. Документ-спецификация разбивается по листам (количество листов спецификации можно получить при помощи функции ksGetSpcDocumentPagesCount). Если функция используется для разбиения спецификации, то параметр type представляет собой номер листа спецификации, начиная с 1.
5. Сложные кривые заменяются набором отрезков и дуг (при level=1 - только набором отрезков).
6. Точность приближения к исходному объекту задается значением параметра arrow - максимальным расстоянием между исходным объектом и аппроксимирующим отрезком.
7. Если level=2, точки превращаются в графические объекты, служащие для отрисовки этих точек в КОМПАС-ГРАФИК (например, в два отрезка для точки типа "крест"). В остальных случаях точки (в том числе отрисованные в виде "крестов", "треугольников" и т.д.) превращаются в объект типа "точка".
8. Во всех случаях, кроме level=4, сложные тексты (например, тексты, написанные буквами разного начертания - прямого и курсивного) разбиваются на тексты с одинаковыми признаками. При level=4 тексты не изменяются.
9. Исходный объект после разбиения не изменяется.

DeleteObj – Удалить объект

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDeleteObj.

Синтаксис:

int DeleteObj (reference obj);

Входной параметр:

obj - указатель на объект.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если указатель obj равен нулю, то удаляются выделенные объекты документа.

ExistObj – Проверить существование объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksExistObj.

Синтаксис:

int ExistObj (reference obj);

Входной параметр:

obj - указатель на объект.

Возвращаемое значение:

1 - в случае наличия объекта,
0 - в случае отсутствия объекта.

GetObjParam – Получить параметры объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjParam.

Синтаксис:

int GetObjParam (reference ref,

void *param,

int parSize,

int parType);

Входной параметр:

ref - указатель на объект,
parType - тип параметров объекта или индекс строки в массиве для объектов TECHNICALDEMAND_OBJ и TEXT_OBJ или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ),
parSize - размер структуры для записи параметров.

Выходные параметры:

param - указатель на структуру параметров,

Возвращаемое значение:

тип объекта - в случае удачного завершения,
0 - в случае неудачи.

Типы объектов и интерфейсы...

Смотрите также

SetObjParam

Примечание:

Вызов метода с нулевыми значениями параметров param, parSize и parType позволяет получить тип объекта по его reference.

ksApproximationCurve – Аппроксимировать кривую дугами

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksApproximationCurve.

Синтаксис:

reference ksApproximationCurve (reference p,

double eps,

unsigned char curentLayer,

double maxRad,

unsigned char smooth);

Входные параметры:

p	- указатель на аппроксимируемую кривую,
eps	- точность аппроксимации (от 1e-7 до 1)
curentLayer	- тип размещения получившихся объектов на слоях:
	0 - на слой кривой,
	1 - в текущий слой,
maxRad	- максимально допустимый радиус дуг,
	(0 - ограничение не накладывается),
smooth	- признак сопряжения:
	1 - гладкое,
	0 - не гладкое.

Возвращаемое значение:

указатель на контур, дугу, отрезок	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Аппроксимируются кривые Безье, NURBS, эквидистанты, эллипсы, дуги эллипсов и контуры. Остальные кривые игнорируются.

ksCalcRasterScale – Рассчитать масштаб для вставки растра в прямоугольник заданных габаритов

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCalcRasterScale.

Синтаксис:

double ksCalcRasterScale (const char* fileName, double w, double h);

Входные параметры:

fileName	- полный путь к файлу,
w	- ширина габаритного прямоугольника,
h	- высота габаритного прямоугольника.

Возвращаемое значение:

значение масштаба	- в случае успешного завершения,
0	- в случае неудачи.

ksCalcRasterScaleW – Рассчитать масштаб для вставки растра в прямоугольник заданных габаритов (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCalcRasterScale.

Синтаксис:

double LIB_FUNC ksCalcRasterScaleW (LPWSTR fileName, double w, double h);

Входные параметры:

fileName	- полный путь к файлу,
w	- ширина габаритного прямоугольника,
h	- высота габаритного прямоугольника.

Возвращаемое значение:

значение масштаба	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksCalcRasterScale.

ksChangeObjectInLibRequest – Изменить фантом или компоненты команд

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksChangeObjectInLibRequest.

Синтаксис:

int ksChangeObjectInLibRequest (RequestInfo *info, void *phantom);

Входные параметры:

info	- указатель на область памяти для замены состава команд,
phantom	- указатель на структуру управления фантомом.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Во время работы функций Cursor и Placement может возникнуть необходимость изменить фантом или компоненты команд. Функция ksChangeObjectInLibRequest позволяет передать изменения в цикл Cursor и Placement. Это может понадобиться для отработки команд пользовательских инструментальных панелей. Предварительно нужно убедиться, что вызов Cursor и требование к Cursor передаются в одном документе. Если какой-либо указатель равен 0, соответствующий параметр не используется. Если оба параметра равны 0, процессы Cursor и Placement прерываются. Это аналогично нажатию Esc.

   После завершения функции, вызвавшей останов процесса, управление будет передано в Cursor или Placement, а затем в функцию пользователя, которая вызывала Cursor или Placement.

2. При использовании Unicode следует использовать функцию ksChangeObjectInLibRequestW.

ksChangeObjectInLibRequestW – Изменить фантом или компоненты команд (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksChangeObjectInLibRequest.

Синтаксис:

int LIB_FUNC ksChangeObjectInLibRequestW (RequestInfoW *info, void *phantom);

Входные параметры:

info	- указатель на область памяти для замены состава команд,
phantom	- указатель на структуру управления фантомом.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Во время работы функций Cursor и Placement может возникнуть необходимость изменить фантом или компоненты команд. Функция ksChangeObjectInLibRequest позволяет передать изменения в цикл Cursor и Placement. Это может понадобиться для отработки команд пользовательских инструментальных панелей. Предварительно нужно убедиться, что вызов Cursor и требование к Cursor передаются в одном документе. Если какой-

либо указатель равен 0, соответствующий параметр не используется. Если оба параметра равны 0, процессы Cursor и Placement прерываются. Это аналогично нажатию Esc.

После завершения функции, вызвавшей останов процесса, управление будет передано в Cursor или Placement, а затем в функцию пользователя, которая вызывала Cursor или Placement.

2. При использовании ANSI следует использовать функцию ksChangeObjectInLibRequest.

ksChangeObjectsOrder – Изменить порядок отрисовки объектов чертежа

Справка системы КОМПАС...

KOMPAS.chm::/65_6_3_Upravlenie_porjadkom_otr.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksChangeObjectsOrder.

Синтаксис:

int ksChangeObjectsOrder (reference group, reference obj, int orderType);

Входной параметр:

obj	- указатель на объект относительно которого изменяется отрисовка группы,
group	- указатель на группу объектов, для которой требуется изменить последовательность отрисовки,
orderType	- тип изменения порядка объектов.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

ksClearRegion – Очистить указанную область

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_BLACKBOX.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksClearRegion.

Синтаксис:

int ksClearRegion (reference grClear,

reference grRegion,

unsigned char inside);

Входные параметры:

grClear	- группа обрабатываемых объектов (0 - просматривать все объекты текущего вида или выделенные, если они есть),
grRegion	- группа объектов, задающая область очистки,
inside	- место очистки: 0 - снаружи области, ограниченной grRegion, 1 - внутри области.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

При очистке объекты, которые пересекаются с границей области очистки, могут быть усечены или полностью удалены.

В зависимости от типа при усечении объекты изменяются (например, отрезок) или заменяются на новые объекты (например, окружность становится дугой).

Новые объекты, созданные в результате операции очистки, помещаются в группу обрабатываемых объектов.

Если группа временная, то для сохранения новых объектов в документе, для нее нужно выполнить функцию StoreTmpGroup.

ksCopyObj – Копировать объект

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCopyObj.

Синтаксис:

reference int ksCopyObj (reference obj,

double xb, double yb,

double xn, double yn,

double scale,

double ang);

Входные параметры:

obj	- указатель на объект,
xb, yb	- координаты базовой точки объекта,
xn, yn	- координаты нового положения базовой точки,
scale	- масштаб,
ang	- угол поворота в градусах.

Возвращаемое значение:

указатель на полученный объект или группу объектов - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Если указатель на объект равен нулю, то копируются выделенные объекты документа.
2. Значения xb,yb, xn,yn задаются в системе координат текущего вида.
3. При копировании одиночного объекта новый объект создается на текущем слое текущего вида.
4. При копировании группы объектов слой сохраняется.
5. При копировании одиночного многослойного макрообъекта он перестает быть многослойным. Макро и входящие в него объекты переносятся на текущий слой. Для копирования многослойного макро с сохранением многослойности требуется добавить его в группу.

ksCopyObjEx – Копировать объект

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCopyObjEx.

Синтаксис:

reference ksCopyObjEx(CopyObjectParam * param);

Входные параметры:

param - указатель на структуру параметров CopyObjectParam.

Возвращаемое значение:

указатель на объект или группу объектов - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция позволяет копировать объект (объект вида, вид, группу, слой) в новую точку с возможностью задания масштабирования копии и поворота ее вокруг базовой точки.
2. При копировании одиночного объекта новый объект создается на текущем слое текущего вида.
3. При копировании группы объектов слой сохраняется.
4. При копировании одиночного многослойного макрообъекта он перестает быть многослойным. Макро и входящие в него объекты переносятся на текущий слой. Для копирования многослойного макро с сохранением многослойности требуется добавить его в группу.

ksDestroyObjects – Разрушить присланные составные объекты

Справка системы КОМПАС...

KOMPAS.chm::/CM_DESTROYMACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDestroyObjects.

Синтаксис:

long ksDestroyObjects(reference p).

Входной параметр:

p	- указатель на объект, вид, слой или группу.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Разрушает не только составные объекты (макроэлементы, вставки фрагментов, эквидистанты, прямоугольники, контуры), но и виды (если у них есть связь с моделью).
2. Функция работает только для документов, открытых в «видимом» режиме.

ksEditViewObject – Запустить визуальный процесс редактирования объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEditViewObject.

Синтаксис:

int ksEditViewObject(reference obj);

Входной параметр:

obj	- указатель на объект.

Возвращаемое значение:

1	- объект отредактирован,
0	- объект не отредактирован.

Примечание:

Функция распространяется на все объекты вида кроме:

LAYER_OBJ	-слой,
CONTOUR_OBJ	-контур,
MACRO_OBJ	-нетипизированный макроэлемент,
FRAGMENT_OBJ	-вставной фрагмент,
ELLIPSE_ARC_OBJ	-дуга эллипса.

ksGetCurvePerpendicularByT – Получить угол нормали к кривой в заданной точке по параметру кривой

Синтаксис:

double LIB_FUNC ksGetCurvePerpendicularByT(reference curve,
double t)

Входные параметры:

curve	- указатель на кривую,
t	- параметр кривой.

Возвращаемое значение:

Функция возвращает угол нормали к кривой в заданной точке по параметру кривой (0 - 360 градусов). В случае неудачи возвращается -1.

ksGetEditMacroVisibleRegime – Находится ли документ или вид в режиме редактирования макроэлемента

Аналог данной функции при использовании Automation - метод IKompasDocument2D1::EditMacroVisibleRegime.

Синтаксис:

int ksGetEditMacroVisibleRegime(reference p)

Входные параметры:

p	- указатель на 2D-документ или вид, 0 - текущий документ.

Возвращаемое значение:

1	- включен визуальный режим редактирования макроэлемента,
0	- обычный режим создания/редактирования объектов.

ksGetObject2DNotifyResult – Получить интерфейс результатов редактирования объекта 2D документа

Аналог данной функции при использовании Automation - метод ksDocument2D::GetObject2DNotifyResult.

Синтаксис:

IObject2DNotifyResult * ksGetObject2DNotifyResult();

Возвращаемое значение:

- указатель на интерфейс источника событий IObject2DNotifyResult

ksGetOrthoMode – Получить режим ортогонального черчения

Справка системы КОМПАС...

KOMPAS.chm::/CM_ORTHO_MODE_ONOFF.htm

Синтаксис:

long ksGetOrthoMode (reference doc);

Входной параметр:

doc - указатель на документ.

Возвращаемое значение:

1 - режим ортогонального черчения включен,
0 - режим ортогонального черчения выключен.

ksGetParametrizationParam – Получить интерфейс параметров параметризации объектов

Справка системы КОМПАС...

KOMPAS.chm::/476_Glava55_Nalozhenie_svjazej_.htm

Аналог данной функции при использовании Automation - метод KompasObject::GetParamStruct. В метод необходимо передать константу ko_ParametrizationParam.

Синтаксис:

IParametrizationParam * ksGetParametrizationParam();

Возвращаемое значение:

Указатель на интерфейс параметров параметризации объектов IParametrizationParam - в случае успешного завершения,
NULL - в случае неудачи.

Примечание:

Данная функция используется совместно с ksParametrizeObjects. Объявление интерфейса IParametrizationParam дано в библиотеке типов kAPI2D5COM.tlb.

ksGetSnapInfo – Получить текущую информацию о привязках

Синтаксис:

ISnapInfo * LIB_FUNC ksGetSnapInfo();

Возвращаемое значение:

- интерфейс информации о текущей привязке ISnapInfo.

Примечание:

Функция позволяет получить информацию о привязке к объектам документа, сетке документа и о других типах привязки в процессах CursorEx, PlacementEx.

Типы возможных привязок см. в перечислении ksSnapTypeEnum.

ksMovePointOnCurve – Получить координаты точки, находящейся на указанном расстоянии вдоль кривой от указанной точки

Пример...

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksMovePointOnCurve.

Синтаксис:

int ksMovePointOnCurve (reference curve,

double * x, double * y,

double len,

int dir);

Входные параметры:

curve	- указатель на кривую,
len	- расстояние, на которое нужно сместить точку,
dir	- направление продвижения точки:
	1 - в направлении построения кривой,
	-1 - в обратном направлении.

Выходные параметры:

x, y	- координаты точки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если точка не находится на кривой, она проецируется на кривую.

ksMovePointOnCurveEx – Получить координаты точки, находящейся на указанном расстоянии вдоль кривой от указанной точки

Синтаксис:

```
int LIB_FUNC  ksMovePointOnCurveEx( reference curve,
                                    double *x, double *y,
                                    double *t,
                                    double len,
                                    int    dir,
                                    int    ext );
```

Входные параметры:

curve	- указатель на кривую,
len	- расстояние, на которое нужно сместить точку,
dir	- направление продвижения точки: 1 - в направлении построения кривой, -1 - в обратном направлении.
ext	-1- Если кривая не замкнута и длина ее части от точки до конца в заданном направлении меньше, чем нужное смещение, то вычислить значение на продолжении кривой, если можно построить продолжение.

Выходные параметры::

x, y	- координаты точки,
t	- параметр кривой.

Примечание:

Функция продвигает точку на расстояние len по кривой.

Если точка не находится на кривой, то точка проецируется на кривую.

Функция возвращает 1 в случае успеха или 0 – при неудаче.

ksSetCursorText – Установить текст курсора для процесса

Синтаксис:

```
int LIB_FUNC ksSetCursorText( char * text );
```

Входной параметр:

Text	- текст.

Возвращаемое значение:

1 - в случае удачи.

Примечание

Функция работает при запущенном процессе.

Текст сразу выдается над курсором.

ksSetCursorTextW – Установить текст курсора для процесса (Unocode)

Синтаксис:

int LIB_FUNC ksSetCursorTextW(LPWSTR text);

Входной параметр:

Text - текст.

Возвращаемое значение:

1 - в случае удачи.

Примечание:

Функция работает при запущенном процессе.

Текст сразу выдается над курсором.

ksReadGroupFromClip – Прочитать графические объекты из буфера обмена и разместить их во временной группе

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/110_8_8_Ispolqzovanie_bufera_ob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksReadGroupFromClip.

Синтаксис:

reference ksReadGroupFromClip();

Возвращаемое значение:

указатель на группу - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Графические объекты должны принадлежать одному виду.
2. Параметрические связи и ограничения объектов при чтении из буфера теряются, атрибуты сохраняются.

ksSetOrthoMode – Задать режим ортогонального черчения

Справка системы КОМПАС...

KOMPAS.chm::/CM_ORTHO_MODE_ONOFF.htm

Синтаксис:

long ksSetOrthoMode (reference doc, long orthoMode);

Входной параметр:

doc - указатель на документ,
orthoMode - режим ортогонального черчения (1 - включить, 0 - выключить).

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

Примечание :

1. Используется только для визуальных документов.
2. Если указатель doc = 0, устанавливается режим ортогонального черчения для активного документа.

ksSymmetryObj – Отразить объект относительно оси

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/CM_OBJSYMMETRY.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSymmetryObj.

Синтаксис:

int SymmetryObj (reference obj,

double x1, double y1,

double x2, double y2 ,

unsigned char copy);

Входные параметры:

obj - указатель на объект,
x1, y1 - координаты первой точки на оси,
x2, y2 - координаты второй точки на оси,
copy - режим копирования:
0 - исходные объекты удаляются,
1 - исходные объекты оставляются.

Возвращаемое значение:

указатель на получившийся объект или группу объектов	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Если указатель obj равен нулю, то зеркально отображаются выделенные объекты документа.

ksTextEx – Создать многострочный текст по структуре параметровTextParam

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTextEx.

Синтаксис:

reference ksTextEx (TextParam * txtParam, int align);

Входные параметры:

txtParam	- указатель на структуру TextParam параметров текста,
align	- выравнивание текста: 0 - слева, 1 - по центру, 2 - справа, 3 - по ширине, -1 установить выравнивание как у стиля текста.

Возвращаемое значение:

указатель на созданный тест	- в случае успешного завершения,
0	- в случае неудачи.

ksTrimCurve – Усечь кривую, оставив часть между указанными точками

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/truncate_objects.htm#truncate_curve

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTrimmCurve.

Синтаксис:

reference ksTrimCurve (reference curve,

double x1, double y1,

double x2, double y2,

double x3, double y3,

unsigned char deleteOldCurve);

Входные параметры:

curve	- указатель на усекаемую кривую,
x1, y1	- координаты начала оставляемого участка,
x2, y2	- координаты конца оставляемого участка,
x3, y3	- координаты точки, определяющей направление усечения для замкнутых кривых (эта точка принадлежит оставляемому участку),
deleteOldCurve	- признак удаления усекаемой кривой: 1 - кривая будет удалена после усечения, 0 - кривая не будет удалена после усечения.

Возвращаемое значение:

указатель на усеченную кривую	- в случае успеха,
0	- в случае неудачи.

Примечание:

Если указанные точки не лежат на кривой, то усечение производится по их проекциям на кривую.

ksTrimNurbs – Усечь NURBS

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTrimNurbs.

Синтаксис:

int ksTrimNurbs (reference pObj,

double tMin, double tMax);

Входные параметры:

pObj	- указатель на кривую NURBS или 0 для создаваемой кривой NURBS.
tMin, tMax	- границы интервала для усечения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksUndoContainer – Включить/отключить объединение операций для Undo

Справка системы КОМПАС...

KOMPAS.chm::/DLG_OTHER.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksUndoContainer.

Синтаксис:

void ksUndoContainer(unsigned char add);

Входные параметры:

add	- false - закрывает текущий контейнер Undo для объединения операций, true - создает текущий контейнер Undo для объединения операций.

Примечание:

Функция позволяет включить/отключить объединение операций для Undo.

ksWriteGroupToClip – Разместить группу в буфере обмена с удалением или оставлением геометрии в документе-источнике (скопировать или вырезать геометрию в буфер обмена)

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/110_8_8_Ispolqzovanie_bufera_ob.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksWriteGroupToClip.

Синтаксис:

int ksWriteGroupToClip (reference g, unsigned char copy);

Входные параметры:

g	- указатель на группу,
copy	- признак копирования или вырезания: 1 - копирование, 0 - вырезание.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

LightObj – Выделить объект цветом

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksLightObj.

Синтаксис:

int LightObj(reference obj, int reg);

Входные параметры:

obj - указатель на объект,
reg - режим подсветки:
1 - установить выделение,
0 - снять выделение.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

При reg=1 объект выделяется, при reg=0 выделение снимается.

MoveObj – Сдвинуть объект

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/make_shift.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksMoveObj.

Синтаксис:

int MoveObj (reference obj, double dx, double dy);

Входные параметры:

obj - указатель на объект,
dx, dy - вектор смещения объекта.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если указатель obj равен нулю, то сдвигаются выделенные объекты документа.

PlacementEx – Запрос к системе на получение точки и угла

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPlacementEx.

Синтаксис:

int PlacementEx (RequestInfo *info,
double *x,

double *y,
double *angle,
void *phantom,
LPUNKNOWN processParam);

Входные параметры:

info	- указатель на структуру параметров запроса к системе,
x, y	- координаты введенной точки,
angle	- введенный угол,
phantom	- указатель на структуру управления фантомом, определяющую тип движения курсора (аналог типа резиновой нити в версии 4),
processParam	- указатель на интерфейс параметры процесса ProcessParam.

Выходные параметры:

x, y	- возвращаемые координаты точки,
angle	- возвращаемое значение угла.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.
2. При использовании Unicode следует использовать функцию PlacementExW.

PlacementExW – Запрос к системе на получение точки и угла

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPlacementEx.

Синтаксис:

int LIB_FUNC PlacementExW (RequestInfoW *info,
double *x,
double *y,
double *angle,
void *phantom,
LPUNKNOWN processParam);

Входные параметры:

info	- указатель на структуру параметров запроса к системе,

x, y - координаты введенной точки,
angle - введенный угол,
phantom - указатель на структуру управления фантомом, определяющую тип движения курсора (аналог типа резиновой нити в версии 4),
processParam - указатель на интерфейс параметры процесса IProcessParam.

Выходные параметры:

x, y - возвращаемые координаты точки,
angle - возвращаемое значение угла.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие метода прекращается после первого шага.
2. При использовании ANSI следует использовать функцию PlacementEx.

RotateObj – Повернуть объект

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/CM_OBJROTATE.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksRotateObj.

Синтаксис:

int RotateObj (reference obj, double xc, double yc, double ang);

Входные параметры:

obj - указатель на объект,
xc, yc - координаты центра поворота,
ang - угол поворота (в градусах).

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если указатель obj равен нулю, то поворачиваются выделенные объекты документа.

SetObjParam – Задать параметры указанного объекта

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetObjParam и ksDocument3D::ksSetObjParam

Синтаксис:

int SetObjParam (reference ref,

void *param,

int parSize,

int paramType);

Входные параметры:

ref	- указатель на объект,
parSize	- размер структуры для записи параметров,
paramType	- тип параметров объекта или индекс строки в массиве для объектов TECHNICALDEMAND_OBJ и TEXT_OBJ или номер страницы (габаритного прямоугольника) технических требований (TECHNICALDEMAND_OBJ).

Выходные параметры:

param	- указатель на структуру параметров.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Типы объектов и интерфейсы...

Примечание:

Данная функция позволяет обрабатывать параметры графического объекта (отрезка, размеров и т.п.), вида, слоя.

Смотрите также GetObjParam

SymmetryObj – Отразить объект относительно оси

Пример...

Синтаксис:

int SymmetryObj (reference obj,

double x1, double y1,

double x2, double y2 ,

unsigned char withCopy);

Входные параметры:

obj	- указатель на объект,
x1, y1	- координаты первой точки на оси,
x2, y2	- координаты второй точки на оси,
withCopy	- режим копирования:
	0 - исходные объекты удаляются,
	1 - исходные объекты оставляются.

Возвращаемое значение:

указатель на получившийся объект или группу объектов	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если указатель obj равен нулю, то зеркально отображаются выделенные объекты документа.
2. Функция устарела, рекомендуется вместо нее использовать функцию ksSymmetryObj.

TransformObj – Преобразовать объект по установленной матрице

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksTransformObj.

Синтаксис:

int TransformObj (reference ref);

Входной параметр:

ref	- указатель на объект.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Для работы с матрицами служат функции Mtr и DeleteMtr.

Функции работы с текстовым документом

CreateTextDocument – Создать текстовый документ

Аналог данной функции при использовании Automation - метод ksDocumentTxt::ksCreateDocument.

Синтаксис:

reference CreateTextDocument (TextDocumentParam * par);

Входные параметры:

par - указатель на структуру параметров TextDocumentParam.

Возвращаемое значение:

указатель на документ - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Документ становится текущим (видимым или невидимым).
2. Задается полное имя файла.
3. При использовании Unicode следует использовать функцию CreateTextDocumentW.

CreateTextDocumentW – Создать текстовый документ (Unicode)

Аналог данной функции при использовании Automation - метод ksDocumentTxt::ksCreateDocument.

Синтаксис:

reference LIB_FUNC CreateTextDocumentW (TextDocumentParamW * par);

Входные параметры:

par - указатель на структуру параметров TextDocumentParam.

Возвращаемое значение:

указатель на документ - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Документ становится текущим (видимым или невидимым).
2. Задается полное имя файла.
3. При использовании ANSI следует использовать функцию CreateTextDocument.

ksGetTxtDocumentPagesCount – Получить количество листов текстового документа

Аналог данной функции при использовании Automation - метод ksDocumentTxt::GetTxtDocumentPagesCount.

Синтаксис:

int ksGetTxtDocumentPagesCount (reference txtDoc);

Входные параметры:

txtDoc - указатель на текстовый документ.

Возвращаемое значение:

количество листов - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если txtDoc=0 - возвращается количество листов текущего текстового документа.

Функции работы с файлами документов

Функции данного раздела обеспечивают вызов диалогов выбора файлов и получение ссылки на текущий документ.

ChoiceFile – Показать диалог и выбрать в нем имя файла для чтения

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFile.

Синтаксис:

unsigned int ChoiceFile (char *ext, char *filtr, char *name, unsigned int sizeBuf);

Входные параметры:

ext - расширение имени файла,
filtr - фильтр поиска
(0 - формируется автоматически),
sizeBuf - размер буфера для имени файла.

Выходные параметры:

name - буфер для имени файла.

Возвращаемое значение:

длина имени файла - в случае удачного завершения,
NULL - в случае отказа от выбора.

FullFileName – Получить полное имя файла

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksFullFileName.

Синтаксис:

unsigned int LIB_FUNC FullFileName (LPSTR oldName, LPSTR newName, unsigned int sizeBuf);

Входные параметры:

oldName - буфер для имени файла,
sizeBuf - размер буфера для имени файла.

Выходные параметры:

newName - буфер для полного имени файла.

Возвращаемое значение:

длина имени файла - в случае удачного завершения,
NULL - в случае отказа от выбора.

Примечание:

1. Если sizeBuf меньше требуемой длины - newName не формируется.
2. При использовании Unicode следует использовать функцию FullFileNameW.

FullFileNameW – Получить полное имя файла (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksFullFileName.

Синтаксис:

unsigned int FullFileNameW (LPWSTR oldName, LPWSTR newName, unsigned int sizeBuf);

Входные параметры:

oldName - буфер для имени файла,
sizeBuf - размер буфера для имени файла.

Выходные параметры:

newName - буфер для полного имени файла.

Возвращаемое значение:

длина имени файла - в случае удачного завершения,
NULL - в случае отказа от выбора.

Примечание:

1. Если sizeBuf меньше требуемой длины - newName не формируется.
2. При использовании ANSI следует использовать функцию FullFileName.

GetRightFileName – Получить действительное имя файла

Пример...

Синтаксис:

unsigned int GetRightFileName (char *fileName, unsigned int sizeBuf);

Входные параметры:

fileName - буфер для имени файла,
sizeBuf - размер буфера для имени файла.

Возвращаемое значение:

длина имени файла - в случае удачного завершения,
NULL - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию GetRightFileNameW.

GetRightFileNameW – Получить действительное имя файла (Unicode)

Синтаксис:

unsigned int LIB_FUNC GetRightFileNameW (LPWSTR fileName, unsigned int sizeBuf);

Входные параметры:

fileName - буфер для имени файла,
sizeBuf - размер буфера для имени файла.

Возвращаемое значение:

длина имени файла - в случае удачного завершения,
NULL - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию GetRightFileName.

ksSelectD3Model – Выбрать из списка открытых или из файла модели

Аналог данной функции при использовании Automation - метод KompasObject::ksSelectD3Model.

Синтаксис:

int LIB_FUNC ksSelectD3Model(char *name,
unsigned int bufLen,

bool onlyDetail,
bool showAddNum);

Входные параметры:

bufLen	- размер буфера name в символах,
onlyDetail	- только детали,
showAddNum	- отображать комбобокс выбора дополнительного номера.

Выходные параметры:

name	- имя файла.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

В диалоге выбора файла могут быть выбраны также обозначение исполнения и дополнительный номер исполнения, если они были созданы в файле модели.

Получить выбранное обозначение исполнения можно с помощью функции ksGetSelectedEmbodimentMarking.

Получить выбранный дополнительный номер исполнения можно с помощью функции ksGetSelectedEmbodimentAdditionalNumber.

ksSelectD3ModelW – Выбрать из списка открытых или из файла модели. Unicode

Аналог данной функции при использовании Automation - метод KompasObject::ksSelectD3Model.

Синтаксис:

int LIB_FUNC ksSelectD3Model(LPWSTR name,
unsigned int bufLen,
bool onlyDetail,
bool showAddNum);

Входные параметры:

bufLen	- размер буфера name в символах,
onlyDetail	- только детали,
showAddNum	- отображать комбобокс выбора дополнительного номера.

Выходные параметры:

name	- имя файла.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

В диалоге выбора файла могут быть выбраны также обозначение исполнения и дополнительный номер исполнения, если они были созданы в файле модели.

Получить выбранное обозначение исполнения можно с помощью функции ksGetSelectedEmbodimentMarkingW.

Получить выбранный дополнительный номер исполнения можно с помощью функции ksGetSelectedEmbodimentAdditionalNumberW.

ksGetSelectedEmbodimentMarking – Возвращает обозначение исполнения, выбранное в диалоге выбора файла (ksSelectD3Model)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSelectedEmbodimentMarking.

Синтаксис:

```
int LIB_FUNC  ksGetSelectedEmbodimentMarking( char      *marking,
                                              unsigned int bufLen );
```

Входные параметры:

bufLen - размер буфера marking в символах.

Выходные параметры:

marking - обозначение исполнения.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

В диалоге выбора файла ksSelectD3Model может быть выбран также дополнительный номер, если он был создан в файле модели.

Получить выбранный дополнительный номер исполнения можно с помощью функции ksGetSelectedEmbodimentAdditionalNumber.

ksGetSelectedEmbodimentMarkingW – Возвращает обозначение исполнения, выбранное в диалоге выбора файла (ksSelectD3ModelW)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSelectedEmbodimentMarking.

Синтаксис:

int LIB_FUNC ksGetSelectedEmbodimentMarkingW(LPWSTR marking,
unsigned int bufLen);

Входные параметры:

bufLen - размер буфера marking в символах.

Выходные параметры:

marking - обозначение исполнения.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

В диалоге выбора файла ksSelectD3ModelW может быть выбран также дополнительный номера, если он был создан в файле модели.

Получить выбранный дополнительный номер исполнения можно с помощью функции ksGetSelectedEmbodimentAdditionalNumberW.

ksGetSelectedEmbodimentAdditionalNumber – Возвращает дополнительный номер исполнения, выбранного в диалоге выбора файла (ksSelectD3Model)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSelectedEmbodimentMarking.

Синтаксис:

int LIB_FUNC ksGetSelectedEmbodimentAdditionalNumber(char *additionalNumber,
unsigned int bufLen);

Входные параметры:

bufLen - размер буфера marking в символах.

Выходные параметры:

additionalNumber - дополнительный номер исполнения.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

В диалоге выбора файла ksSelectD3Model может быть выбрано также обозначение исполнения, если оно было создано в файле модели.

Получить выбранное обозначение исполнения можно с помощью функции ksGetSelectedEmbodimentMarking.

ksGetSelectedEmbodimentAdditionalNumberW – Возвращает дополнительный номер исполнения, выбранного в диалоге выбора файла (ksSelectD3ModelW). Unicode

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSelectedEmbodimentMarking.

Синтаксис:

int LIB_FUNC ksGetSelectedEmbodimentAdditionalNumberW(LPWSTR additionalNumber,
unsigned int bufLen);

Входные параметры:

bufLen - размер буфера marking в символах.

Выходные параметры:

additionalNumber - дополнительный номер исполнения.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

В диалоге выбора файла ksSelectD3ModelW может быть выбрано также обозначение исполнения, если оно было создано в файле модели.

Получить выбранное обозначение исполнения можно с помощью функции ksGetSelectedEmbodimentMarkingW.

ksChoiceFile – Выдать диалог выбора файла для чтения

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFile

Синтаксис:

unsigned int ksChoiceFile (char *ext,

char * filter,

char * name,

unsigned int bufLen,

unsigned char preview);

Входные параметры:

Ext	- расширение имени файла,
Filter	- фильтр поиска (0 - формируется автоматически),
Preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходные параметры:

Name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

требуемая длина буфера в символах	- в случае удачного завершения,
NULL	- в случае отказа от выбора.

Примечание:

1. Если размер отведенного буфера bufLen меньше требуемой длины, то имя файла не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.
2. При использовании Unicode следует использовать функцию ksChoiceFileW.

ksChoiceFileW – Выдать диалог выбора файла для чтения (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFile

Синтаксис:

unsigned int LIB_FUNK ksChoiceFileW (LPWSTR ext,

LPWSTR filter,

LPWSTR name,

unsigned int bufLen,

unsigned char preview);

Входные параметры:

Ext	- расширение имени файла,
Filter	- фильтр поиска (0 - формируется автоматически),
Preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходные параметры:

name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

требуемая длина буфера в символах	- в случае удачного завершения,
NULL	- в случае отказа от выбора.

Примечание:

1. Если размер отведенного буфера bufLen меньше требуемой длины - имя файла не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.
2. При использовании ANSI следует использовать функцию ksChoiceFile.

ksChoiceFiles – Выдать диалог выбора файлов для чтения

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFiles

Синтаксис:

int ksChoiceFile (char * ext,

char * filter,

reference p,

unsigned char preview);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - формируется автоматически),

preview - признак подключения окна предварительного просмотра:
1 - с подключением окна,
0 - без подключения окна.

Выходной параметр:

p - указатель на динамический массив строк CHAR_STR_ARR или CHAR_STR_ARR_W.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksChoiceFilesW.

ksChoiceFilesW – Выдать диалог выбора файлов для чтения (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFiles

Синтаксис:

int LIB_FUNK ksChoiceFileW (LPWSTR ext,

LPWSTR filter,

reference p,

unsigned char preview);

Входные параметры:

ext - расширение имени файла,
filter - фильтр поиска
(0 - формируется автоматически),
preview - признак подключения окна предварительного просмотра:
1 - с подключением окна,
0 - без подключения окна.

Выходной параметр:

p - указатель на динамический массив строк CHAR_STR_ARR или CHAR_STR_ARR_W.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksChoiceFiles.

ksChoiceFileAppointedDir – Выдать диалог выбора файла для чтения

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFileAppointedDir

Синтаксис:

unsigned int ksChoiceFileAppointedDir (char *ext,

char * filter,

char * name,

unsigned int bufLen,

unsigned char preview

int typeDir);

Входные параметры:

ext	- расширение имени файла,
filter	- фильтр поиска (0 - фильтр формируется автоматически),
preview	- признак подключения окна предварительного просмотра:
	1 - с подключением окна,
	0 - без подключения окна.
typeDir	- стартовая папка.

Выходные параметры:

Name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

требуемая длина буфера в символах	- в случае удачного завершения,
NULL	- в случае отказа от выбора.

Примечание:

1. Параметр typeDir может иметь значения:

- sptSYSTEM_FILES - 0 - открывается папка системных файлов,
- sptLIBS_FILES - 1 - открывается папка файлов библиотек. Во всех остальных случаях открывается текущая папка.

2. Если размер отведенного буфера bufLen меньше требуемой длины - name не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.
3. При использовании Unicode следует использовать функцию ksChoiceFileAppointedDirW.

ksChoiceFileAppointedDirW – Выдать диалог выбора файла для чтения (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksChoiceFileAppointedDir

Синтаксис:

unsigned int LIB_FUNK ksChoiceFileAppointedDirW (LPWSTR ext,

LPWSTR filter,

LPWSTR name,

unsigned int bufLen,

unsigned char preview

int typeDir);

Входные параметры:

Ext	- расширение имени файла,
Filter	- фильтр поиска (0 - фильтр формируется автоматически),
preview	- признак подключения окна предварительного просмотра:
	1 - с подключением окна,
	0 - без подключения окна.
typeDir	- стартовая папка.

Выходные параметры:

name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

требуемая длина буфера в символах	- в случае удачного завершения,
NULL	- в случае отказа от выбора.

Примечание:

1. Параметр typeDir может иметь значения:

- sptSYSTEM_FILES - 0 - открывается папка системных файлов,
- sptLIBS_FILES - 1 - открывается папка файлов библиотек. Во всех остальных случаях открывается текущая папка

2. Если размер отведенного буфера bufLen меньше требуемой длины - name не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.
3. При использовании ANSI следует использовать функцию ksChoiceFileAppointedDir.

ksClearFileCache – Очистить кеш поиска файлов

Синтаксис:

int LIB_FUNC ksClearFileCache();

Примечание:

Кеширование включается при запуске библиотечной команды.

ksClearRecoverError – Очистить признак ошибки после открытия файла с восстановлением

Синтаксис:

int LIB_FUNC ksClearRecoverError();

ksFileNameCompare – Сравнить имена файлов

Синтаксис:

int LIB_FUNC ksFileNameCompare(char * fileName1, char * fileName2);

Входные параметры:

fileName1, fileName2 - сравниваемые имена файлов.

Возвращаемое значение:

1 - имена не совпадают,
0 - имена совпадают.

Примечание.

При использовании Unicode следует использовать функцию ksFileNameCompareW.

ksFileNameCompareW – Сравнить имена файлов (Unicode)

Синтаксис:

int LIB_FUNC ksFileNameCompareW(LPWSTR fileName1, LPWSTR fileName2);

Входные параметры:

fileName1, fileName2 - сравниваемые имена файлов.

Возвращаемое значение:

1 - имена не совпадают,
0 - имена совпадают.

Примечание.

При использовании ANSI следует использовать функцию ksFileNameCompare.

ksGetCurrentDocument – Получить указатель на текущий документ

Пример...

Синтаксис:

reference ksGetCurrentDocument (unsigned char type);

Входные параметры:

type - тип документа:
0 - любой документ,
1 - только графический документ,
2 - только спецификация
3 - объемная модель (деталь или сборка)
4 - текстовый документ.

Возвращаемое значение:

указатель на документ - в случае удачного завершения,
0 - в случае неудачи.

ksGetFullPathFromSystemPath – Сформировать полный путь к файлу из заданного относительного пути к файлу и системного пути установленного типа

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetFullPathFromSystemPath.

Синтаксис:

void ksGetFullPathFromSystemPath (char *relativePath,
char *destBuff,
int bufLen,
int pathType);

Входные параметры:

relativePath - относительный путь к файлу,
pathType - тип системной папки,
bufLen - длина отведенного буфера для возвращаемого значения в символах.

Типы системных папок...

Выходные параметры:

destBuff - буфер для сформированного полного пути к файлу.

Примечание.

При использовании Unicode следует использовать функцию ksGetFullPathFromSystemPathW.

ksGetFullPathFromSystemPathW – Сформировать полный путь к файлу из заданного относительного пути к файлу и системного пути установленного типа

Аналог данной функции при использовании Automation - метод KompasObject::ksGetFullPathFromSystemPath.

Синтаксис:

void LIB_FUNC ksGetFullPathFromSystemPath (LPWSTR relativePath,
LPWSTRdestBuff,
int bufLen,
int pathType);

Входные параметры:

relativePath - относительный путь к файлу,
pathType - тип системной папки,
bufLen - длина отведенного буфера для возвращаемого значения в символах.

Типы системных папок...

Выходные параметры:

destBuff - буфер для сформированного полного пути к файлу.

Примечание.

При использовании ANSI следует использовать функцию ksGetFullPathFromSystemPath.

ksGetRecoverError – Получить признак ошибки после открытия файла с восстановлением

Синтаксис:

int LIB_FUNC ksGetRecoverError();

ksGetRecoverMode – Получить признак открытия файлов в режиме восстановления

Синтаксис:

int LIB_FUNC ksGetRecoverMode();

ksGetFullPathFromRelativePath – Сформировать полный путь к файлу из заданного пути к задающему файлу и относительного пути к файлу

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetFullPathFromRelativePath.

Синтаксис:

void ksGetFullPathFromRelativePath (char *mainFilePath,
char *relativePath,
char *destBuff,
int bufLen);

Входные параметры:

mainFilePath	- полный путь к задающему файлу,
relativePath	- относительный путь к требуемому файлу (без общей с задающим файлом части пути),
bufLen	- длина отведенного буфера для возвращаемого значения в символах.

Выходные параметры:

destBuff	- буфер для сформированного относительного пути к файлу.

Примечание.

При использовании Unicode следует использовать функцию ksGetFullPathFromRelativePathW.

ksGetFullPathFromRelativePathW – Сформировать полный путь к файлу из заданного пути к задающему файлу и относительного пути к файлу (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetFullPathFromRelativePath.

Синтаксис:

void LIB_FUNC ksGetFullPathFromRelativePathW (LPWSTR mainFilePath,
LPWSTR relativePath,
LPWSTR destBuff,
int bufLen);

Входные параметры:

mainFilePath	- полный путь к задающему файлу,

relativePath - относительный путь к требуемому файлу (без общей с задающим файлом части пути),
bufLen - размер буфера для возвращаемого значения.

Выходные параметры:

destBuff - буфер для сформированного относительного пути к файлу.

Примечание.

При использовании ANSI следует использовать функцию ksGetFullPathFromRelativePath.

ksGetRelativePathFromFullPath – Сформировать относительный путь к файлу из полного пути к задающему файлу и полного пути к файлу

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetRelativePathFromFullPath.

Синтаксис:

void ksGetRelativePathFromFullPath (char *mainFilePath,
char *sourcePath,
char *destBuff,
int bufLen);

Входные параметры:

mainFilePath - полный путь к задающему файлу,
sourcePath - полный путь к требуемому файлу,
bufLen - длина отведенного буфера для возвращаемого значения в символах.

Выходные параметры:

destBuff - буфер для сформированного относительного пути к файлу.

Примечание.

При использовании Unicode следует использовать функцию ksGetRelativePathFromFullPathW.

ksGetRelativePathFromFullPathW – Сформировать относительный путь к файлу из полного пути к задающему файлу и полного пути к файлу (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetRelativePathFromFullPath.

Синтаксис:

void LIB_FUNC ksGetRelativePathFromFullPath (LPWSTR mainFilePath,
LPWSTR sourcePath,
LPWSTR destBuff,
int bufLen);

Входные параметры:

mainFilePath - полный путь к задающему файлу,
sourcePath - полный путь к требуемому файлу,
bufLen - длина отведенного буфера для возвращаемого значения в символах.

Возвращаемое значение:

destBuff - буфер для сформированного относительного пути к файлу.

Примечание.

При использовании ANSI следует использовать функцию ksGetRelativePathFromFullPath.

ksGetRelativePathFromSystemPath – Сформировать относительный путь к файлу из заданного полного пути к файлу

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetRelativePathFromSystemPath.

Синтаксис:

void ksGetRelativePathFromSystemPath (char *sourcePath,
char *destBuff,
int bufLen,
int pathType);

Входные параметры:

sourcePath - полный путь к файлу,
bufLen - длина отведенного буфера для возвращаемого значения в символах,
pathType - тип системной папки.

Типы системных папок...

Выходные параметры:

destBuff - буфер для сформированного относительного пути к файлу.

Примечание.

При использовании Unicode следует использовать функцию ksGetRelativePathFromSystemPathW.

ksGetRelativePathFromSystemPathW – Сформировать относительный путь к файлу из заданного полного пути к файлу (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetRelativePathFromSystemPath.

Синтаксис:

void LIB_FUNC ksGetRelativePathFromSystemPath (LPWSTR sourcePath,
LPWSTR destBuff,
int bufLen,
int pathType);

Входные параметры:

sourcePath - полный путь к файлу,
bufLen - длина отведенного буфера для возвращаемого значения в символах,
pathType - тип системной папки.

Типы системных папок...

Выходные параметры:

destBuff - буфер для сформированного относительного пути к файлу.

Примечание.

При использовании ANSI следует использовать функцию ksGetRelativePathFromSystemPath.

ksLockFileCache – Выключить/включить кеширование поиска файлов

Синтаксис:

int LIB_FUNC ksLockFileCache(int lock);

Входные параметры:

lock - TRUE - запретить кеширование,
- FALSE - разрешить кеширование.

Возвращаемое значение:

Предыдущее значение флага.

Примечание:

Кеширование включается при запуске библиотечной команды.

ksSaveFile – Выдать диалог выбора файла для сохранения

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksSaveFile.

Синтаксис:

unsigned int ksSaveFile (char * ext,

char * oldName,

char * filter,

char * name,

unsigned int bufLen,

unsigned char preview);

Входные параметры:

Ext	- расширение имени файла,
oldName	- имя файла по умолчанию,
Filter	- фильтр поиска (0 - формируется автоматически),
Preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходные параметры:

Name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

- требуемая длина буфера в символах.

Описание:

Если размер отведенного буфера меньше требуемой длины bufLen, имя файла не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.

Примечание.

При использовании Unicode следует использовать функцию ksSaveFileW.

ksSaveFileW – Выдать диалог выбора файла для сохранения (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksSaveFile.

Синтаксис:

unsigned int LIB_FUNC ksSaveFile (LPWSTR ext,

LPWSTR oldName,

LPWSTR filter,

LPWSTR name,

unsigned int bufLen,

unsigned char preview);

Входные параметры:

Ext	- расширение имени файла,
oldName	- имя файла по умолчанию,
Filter	- фильтр поиска (0 - формируется автоматически),
preview	- признак подключения окна предварительного просмотра: 1 - с подключением окна, 0 - без подключения окна.

Выходные параметры:

Name	- буфер для имени файла,
bufLen	- длина отведенного буфера для name в символах.

Возвращаемое значение:

- требуемая длина буфера в символах.

Описание:

Если размер отведенного буфера меньше требуемой длины bufLen - имя файла не заполняется. В этом случае имя файла можно получить функцией GetRightFileName.

Примечание.

При использовании ANSI следует использовать функцию ksSaveFile.

ksSetRecoverMode – Установить признак открытия файлов в режиме восстановления

Синтаксис:

int LIB_FUNC ksSetRecoverMode(int mode);

RemoveUniqueFile – Удалить уникальный служебный файл

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksRemoveUniqueFile.

Синтаксис:

void RemoveUniqueFile (char *nameFile);

Описание:

Удалить служебный файл с уникальным именем, предварительно полученным с помощью функции UniqueFileName.

Возвращаемое значение:

1	- в случае успешного выполнения,
0	- в случае ошибки.

Примечание.

При использовании Unicode следует использовать функцию RemoveUniqueFileW.

RemoveUniqueFileW – Удалить уникальный служебный файл (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksRemoveUniqueFile.

Синтаксис:

void LIB_FUNC RemoveUniqueFileW (LPWSTR nameFile);

Описание:

Удалить служебный файл с уникальным именем, предварительно полученным с помощью функции UniqueFileName.

Возвращаемое значение:

1	- в случае успешного выполнения,
0	- в случае ошибки.

Примечание.

При использовании ANSI следует использовать функцию RemoveUniqueFile.

UniqueFileName – Получить уникальное имя файла

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksUniqueFileName.

Синтаксис:

unsigned int UniqueFileName (char *buff, unsigned int sizeBuf);

Входные параметры:

buff	- буфер для имени файла,
sizeBuf	- размер буфера для имени файла.

Возвращаемое значение:

длина имени файла	- в случае удачного завершения,
NULL	- в случае неудачи.

Примечание:

1. Получить уникальное имя для создания служебного файла в процессе работы библиотеки и поместить его в буфер buff. Параметр sizeBuf определяет размер буфера. Имя регистрируется и запоминается системой. По окончании работы в случае, если файл с таким именем существует, он будет автоматически удален.
2. При использовании Unicode следует использовать функцию UniqueFileNameW.

UniqueFileNameW – Получить уникальное имя файла (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksUniqueFileName.

Синтаксис:

unsigned int LIB_FUNC UniqueFileNameW (LPWSTR buff, unsigned int sizeBuf);

Входные параметры:

buff	- буфер для имени файла,
sizeBuf	- размер буфера для имени файла.

Возвращаемое значение:

длина имени файла	- в случае удачного завершения,
NULL	- в случае неудачи.

Примечание:

1. Получить уникальное имя для создания служебного файла в процессе работы библиотеки и поместить его в буфер buff. Параметр sizeBuf определяет размер буфера. Имя регистрируется и запоминается системой. По окончании работы в случае, если файл с таким именем существует, он будет автоматически удален.
2. При использовании ANSI следует использовать функцию UniqueFileName.

Функции работы с документом-спецификацией

Функции данного раздела позволяют работать с документом-спецификацией.

CreateSpcIterator – Создать итератор для перебора объектов спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksCreateSpcIterator.

Синтаксис:

reference CreateSpcIterator (char * nameLib,
unsigned int styleNumb,
int spcObjType);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
styleNumb	- номер стиля спецификации в библиотеке,
spcObjType	- тип объектов:
	0 - базовые,
	1 - вспомогательные,
	2 - базовые и вспомогательные из сортированного массива,
	3 - все объекты.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если функция вызывается в графическом документе, то итератор создается для объектов, образованных по описанию, определяемому именем библиотеки стилей спецификации и номером стиля.
2. В документе-спецификации параметры nameLib и styleNumb не используются (т.к. все объекты в ней создаются по единственному описанию, однозначно определяемому текущим стилем).
3. При использовании Unicode следует использовать функцию CreateSpcIteratorW.

CreateSpcIteratorW – Создать итератор для перебора объектов спецификации (Unicode)

Аналог данной функции при использовании Automation - метод ksIterator::ksCreateSpcIterator.

Синтаксис:

reference CreateSpcIteratorW (LPWSTR nameLib,
unsigned int styleNumb,
int spcObjType);

Входные параметры:

nameLib - имя библиотеки стилей спецификации,
styleNumb - номер стиля спецификации в библиотеке,
spcObjType - тип объектов:
0 - базовые,
1 - вспомогательные,
2 - базовые и вспомогательные из сортированного массива,
3 - все объекты.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если функция вызывается в графическом документе, то итератор создается для объектов, образованных по описанию, определяемому именем библиотеки стилей спецификации и номером стиля.
2. В документе-спецификации параметры nameLib и styleNumb не используются (т.к. все объекты в ней создаются по единственному описанию, однозначно определяемому текущим стилем).
3. При использовании ANSI следует использовать функцию CreateSpcIterator.

D3GetSpcObjForGeomWithLimit – Получить указатель на объект спецификации, подключенный к трехмерному компоненту (детали или подсборке) с ограничениями по номеру раздела и типу атрибута

Справка системы КОМПАС...

KOMPAS.chm::/1154_132_2_Obwekty_specifikacii.htm

Аналог данного метода при при использовании Automation - метод D3GetSpcObjForGeomWithLimit.

Синтаксис:

long D3GetSpcObjForGeomWithLimit (BSTR nameLib,

long numb,

LPDISPATCH part,

short first,

short section,

double attrTypeNumb);

Входные параметры:

nameLib - имя библиотеки стилей спецификации,
numb - номер стиля спецификации в библиотеке,

part - указатель на интерфейс добавляемого компонента ksPart,
first - 1 -первый объект, 0 - следующий объект,
section - номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb - номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

D3SpcIncludePart – Добавить или изменить трехмерный компонент (деталь или подсборку) в объекте спецификации

Аналог данного метода при использовании Automation - метод D3SpcIncludePart.

Синтаксис:

BOOL D3SpcIncludePart (LPDISPATCH part, BOOL fillTexts);

Входные параметры:

part - указатель на интерфейс компонента ksPart,
fillTexts - признак автоматического заполнения: наименование, обозначение, масса из свойств ksPart:
TRUE - автозаполнение включено,
FALSE - выключено.

Возвращаемое значение:

TRUE - в случае удачного завершения.

ksAddSpcDescription – Добавить описание спецификации в графический документ

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksAddSpcDescription.

Синтаксис:

int ksAddSpcDescription (reference pDoc,
SpcDescrParam* param);

Входной параметр:

pDoc - указатель на документ,
param - параметры описания спецификации.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Структура параметров описания спецификации....

Примечание.

При использовании Unicode следует использовать функцию ksAddSpcDescriptionW.

ksAddSpcDescriptionW – Добавить описание спецификации в графический документ (Unicode

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksAddSpcDescription.

Синтаксис:

int ksAddSpcDescriptionW (reference pDoc,
SpcDescrParamW* param);

Входной параметр:

pDoc - указатель на документ,
param - параметры описания спецификации.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Структура параметров описания спецификации....

Примечание.

При использовании ANSI следует использовать функцию ksAddSpcDescription.

ksD3GetSpcObjGeometry – Получить деталь или подсборку, подключенную к объекту спецификации

Аналог данного метода при при использовании Automation - метод ksD3GetSpcObjGeometry.

Синтаксис:

LPDISPATCH D3GetSpcObjGeometry (long spcObj);

Входные параметры:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

Указатель на интерфейс ksPart - в случае успешного завершения,
0 - если деталь не подключена или в случае ошибки.

ksDeleteSpcDescription – Удалить описание спецификации из графического документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksDeleteSpcDescription.

Синтаксис:

int ksDeleteSpcDescription (reference pDoc,

int index);

Входной параметр:

pDoc - указатель на документ,
index - индекс (номер) описания спецификации в документе.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksEditWindowSpcObject – Запустить окно редактирования для объекта спецификации, существующего в графическом документе

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksEditWindowSpcObject.

Синтаксис:

int ksEditWindowSpcObject (reference spcObj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Функция открывает диалог редактирования для объекта спецификации, расположенного в графическом документе.
2. Функция запускает процесс. Так как система может работать только с одним процессом, нужно завершить другие процессные функции: Cursor, Placement, CommandWindow, ksEditViewObject, ksCreateViewObject;

ksGetApplyLibrarySpcStyle – Получить признак использования стиля спецификации

Синтаксис:

int LIB_FUNC ksGetApplyLibrarySpcStyle();

Возвращаемое значение:

1 - для работы со спецификацией использовать стиль СП из библиотеки, а не втянутый в документ,
0 - для работы со спецификацией использовать стиль втянутый в документ.

ksGetCurrentSpcObject – Получить указатель текущего (выделенного или редактируемого) объекта спецификации

Аналог данного метода при использовании использовании Automation - метод ksGetCurrentSpcObject.

Синтаксис Automation:

long ksGetCurrentSpcObject();

Возвращаемое значение:

указатель на текущий объект - в случае успеха,
0 - в случае неудачи.

Примечание:

Метод работает для видимых таблиц спецификации.

ksGetSpcColumnNumb – Для данного объекта спецификации по типу колонки и номеру блока получить номер колонки

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcColumnNumb.

Синтаксис:

int ksGetSpcColumnNumb (reference spcObj,
unsigned int columnType,
unsigned int typeNumb,
unsigned int block);

Входные параметры:

spcObj	- указатель на объект спецификации,
columnType	- тип колонки,
ispoln	- номер колонки данного типа, начиная с 1,
block	- номер блока исполнений.

Возвращаемое значение:

номер колонки спецификации.	- в случае успеха,
0	- в случае неудачи.

Типы колонок спецификации...

ksGetSpcColumnType – По номеру колонки для данного объекта спецификации получить параметры колонки

Пример...

По номеру колонки для данного объекта спецификации получить параметры колонки:

- тип колонки (SPC_CLM_FORMAT...SPC_CLM_USER),
- исполнение данного типа,
- номер блока,
- тип значения (зависит от раздела),
- имя колонки.

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcColumnType.

Синтаксис:

int ksGetSpcColumnType (reference spcObj,
unsigned int colNumb,
SpcColumnParam * par);

Входные параметры:

spcObj - указатель на объект спецификации,
colNumb - номер колонки, начиная с 1.

Выходной параметр:

par - указатель на структуру параметров колонки спецификации SpcColumnParam.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Типы колонок спецификации...

Примечание:

1. Если число исполнений больше количества колонок для исполнений в бланке спецификации, недостающие колонки исполнений пристыковываются справа от отображаемых колонок и получают номера, следующие за номерами колонок таблицы спецификации, затем пристыковываются и нумеруются дополнительные колонки.
2. При использовании Unicode следует использовать функцию ksGetSpcColumnTypeW.

ksGetSpcColumnTypeW – По номеру колонки для данного объекта спецификации получить параметры колонки (Unicode)

По номеру колонки для данного объекта получить параметры:

- тип колонки (SPC_CLM_FORMAT...SPC_CLM_USER),
- исполнение данного типа,
- номер блока,
- тип значения(зависит от раздела),
- имя колонки.

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcColumnType.

Синтаксис:

int ksGetSpcColumnTypeW (reference spcObj,
unsigned int colNumb,
SpcColumnParamW *par);

Входные параметры:

spcObj - указатель на объект спецификации,
colNumb - номер колонки, начиная с 1.

Выходной параметр:

par - указатель на структуру параметров колонки спецификации SpcColumnParamW.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Типы колонок спецификации...

Примечание:

1. Если число исполнений больше количества колонок для исполнений в бланке спецификации, недостающие колонки исполнений пристыковываются справа от отображаемых колонок и получают номера, следующие за номерами колонок таблицы спецификации, затем пристыковываются и нумеруются дополнительные колонки.
2. При использовании ANSI следует использовать функцию ksGetSpcColumnType.

ksGetSpcDescription – Получить параметры описания спецификации для заданного документа

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SPC_DESCRIBE.htm`

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcDescription.

Синтаксис:

int ksGetSpcDescription (reference pDoc,

int index,

SpcDescrParam * param,

unsigned char* state);

Входной параметр:

pDoc - указатель на документ,
index - индекс (номер) описания спецификации в документе.

Выходные параметры:

param - структура параметров описания спецификации,
state - состояние описания спецификации:
TRUE - текущее,
FALSE - не текущее.

Структура параметров описания спецификации....

Описание:

Функция возвращает параметры описания спецификации для указанного документа.

Примечание:

1. Если index = -1, берется текущее описание. Для документа-спецификации параметр index не используется.
2. При использовании Unicode следует использовать функцию ksGetSpcDescriptionW.

ksGetSpcDescriptionW – Получить параметры описания спецификации для заданного документа (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcDescription.

Синтаксис:

int ksGetSpcDescriptionW (reference pDoc,

int index,

SpcDescrParamW * param,

unsigned char* state);

Входной параметр:

pDoc	- указатель на документ,
index	- индекс (номер) описания спецификации в документе.

Выходные параметры:

param	- структура параметров описания спецификации,
state	- состояние описания спецификации: TRUE - текущее, FALSE - не текущее.

Структура параметров описания спецификации....

Описание:

Функция возвращает параметры описания спецификации для указанного документа.

Примечание:

1. Если index = -1, берется текущее описание. Для документа-спецификации параметр index не используется.
2. При использовании ANSI следует использовать функцию ksGetSpcDescription.

ksGetSpcDocumentPagesCount – Получить количество листов документа-спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpcDocument::ksGetSpcDocumentPagesCount.

Синтаксис:

int ksGetSpcDocumentPagesCount (reference spcDoc);

Входные параметры:

spcDoc - указатель на документ-спецификацию (если spcDoc = 0, функция работает для текущей спецификации).

Возвращаемое значение:

количество листов документа-спецификации.

ksGetSpcObject – Получить объект спецификации по номеру

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObject.

Синтаксис:

reference ksGetSpcObject (double objNumb);

Входной параметр:

objNumb - уникальный номер объекта спецификации.

Возвращаемое значение:

- указатель на объект спецификации.

ksGetSpcObjectAttributeNumber – Получить номер атрибута объекта спецификации

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectAttributeNumber.

Синтаксис:

double ksGetSpcObjectAttributeNumber (reference spcObj);

Входные параметры:

spcObj - объект спецификации.

Возвращаемое значение:

- Номер атрибута.

Примечания:

Если spcObj = 0, то возвращается номер редактируемого в текущий момент объекта спецификации, т.е. внутри блока:

ksSpcObjectEdit(...); .

ksGetSpcObjectAttributeNumber(...);

ksSpcObjectEnd();

ksGetSpcObjectColumnText – Получить текст из колонки определенного типа заданного исполнения объекта спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectColumnText

Синтаксис:

int ksGetSpcObjectColumnText (reference spcObj,

unsigned int columnType

unsigned int ispoln,

unsigned int block,

char *s,

unsigned int size_s); // длина строки s

Входные параметры:

spcObj	- указатель на объект спецификации,
columnType	- тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер исполнения данного типа,
block	- номер блока исполнений, начиная с 0 (если количество исполнений меньше количества исполнений в бланке спецификации),
s	- указатель на строку, в которую требуется положить полученный текст,
size_s	длина строки s.

Типы колонок спецификации...

Выходные параметры:

строка с текстом из колонки.

Возвращаемое значение:

0	- в случае успешного завершения,
количество символов	- если длина строки оказалась меньше требуемой size_s,

-1 в случае ошибки.

Примечание:

1. Для групповой спецификации 2.113-75 число исполнений меньше 10.
2. При использовании Unicode следует использовать функцию ksGetSpcObjectColumnTextW.

ksGetSpcObjectColumnTextEx – Получить текст объекта спецификации для определенного типа колонки и исполнения в виде динамического массива TEXT_LINE_ARR

Аналог данной функции при использовании Automation - метод ksSpecification::GetSpcObjectColumnTextEx.

Синтаксис:

reference ksGetSpcObjectColumnTextEx (reference spcObj,
unsigned int columnType,
unsigned int ispoln,
unsigned int block)

Входные параметры:

spcObj	объект спецификации,
columnType	тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	номер колонки данного типа, начиная с 1,
block	номер блока.

Возвращаемое значение:

- динамический массив строк текста TEXT_LINE_ARR,	- в случае успеха,
0	- в случае ошибки.

Примечание:

Если номер блока block = 0 - число исполнений меньше количества исполнений в бланке спецификации (для групповой спецификации 2.113-75 число исполнений меньше 10).

ksGetSpcObjectColumnTextW – Получить текст из колонки определенного типа заданного исполнения объекта спецификации (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectColumnText.

Синтаксис:

int ksGetSpcObjectColumnTextW (reference spcObj,

unsigned int columnType

unsigned int ispoln,

unsigned int block,

LPWSTR s,

unsigned int size_s); // длина строки s

Входные параметры:

spcObj	- указатель на объект спецификации,
columnType	- тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер исполнения данного типа,
block	- номер блока исполнений, начиная с 0 (если количество исполнений меньше количества исполнений в бланке спецификации),
s	- указатель на строку, в которую требуется положить полученный текст,
size_s	длина строки s.

Типы колонок спецификации...

Выходные параметры:

строка с текстом из колонки.

Возвращаемое значение:

0	- в случае успешного завершения,
количество символов	- если длина строки оказалась меньше требуемой size_s,
-1	в случае ошибки.

Примечание:

1. Для групповой спецификации 2.113-75 число исполнений меньше 10.
2. При использовании ANSI следует использовать функцию ksGetSpcObjectColumnText.

ksGetSpcObjectColumnTextAlign – Получить выравнивание для текста колонки объекта спецификации

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectColumnTextAlign.

Синтаксис:

int ksGetSpcObjectColumnTextAlign (reference spcObj,
int columnNumber,
int lineIndex);

Входной параметр:

spcObj - указатель на объект спецификации,
columnNumber - номер колонки, начиная с 1,
lineIndex - индекс строки текста.

Возвращаемое значение:

- Способ выравнивания в ячейке.

Примечания:

Значение переменной align может принимать значения:

- 0 - Выравнивание влево,
- 1 - Выравнивание по центру,
- 2 - Выравнивание вправо,
- 3 - Выравнивание по ширине.

ksGetSpcObjectColumnValues – Получить массив значений для определенного типа колонки и исполнения значений для определенного типа колонки и исполнения

Синтаксис:

reference ksGetSpcObjectColumnVa

lues (reference spcObj,
unsigned int columnType,
unsigned int ispoln,
unsigned int block);

Входные параметры:

spcObj - указатель на объект спецификации,
columnType - тип колонки,
ispoln - номер колонки данного типа, начиная с 1,
block - номер блока исполнений.

Возвращаемое значение:

динамический массив типа LTVARIANT_ARR - в случае успеха,
0 - в случае неудачи.

Примечание.

Для колонки типа "запись" функция вернет массив значений записи, а для колонки, состоящей из одного элемента - массив из одного элемента соответствующего типа.

ksGetSpcObjectNumber – Получить уникальный номер объекта спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectNumber.

Синтаксис:

double ksGetSpcObjectNumber (reference spcObj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

- уникальный номер объекта спецификации.

ksGetSpcObjForGeom – Получить указатель объекта СП по геометрии для текущего графического документа

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjForGeom.

Синтаксис:

reference ksGetSpcObjForGeom (char * nameLib,
unsigned int numb,
reference obj,
unsigned char equal,
unsigned char first);

Входные параметры:

nameLib - имя библиотеки стилей спецификации,
numb - номер стиля спецификации в библиотеке,
obj - указатель на графический объект или группу объектов (0 - макроэлемент редактирования),
equal - признак вхождения геометрии в состав объекта:
1 - геометрия идентична геометрии объекта спецификации,
0 - геометрия входит в объект спецификации,
first - 1 -первый объект, 0 - следующий объект.

Возвращаемое значение:

- указатель на объект спецификации.

Примечания:

1. Данный метод - только для графических документов.
2. В настоящий момент метод реализован только для equal = 1.

ksGetSpcObjForGeomWithLimit – Получить указатель на объект спецификации, подключенный к данному графическому объекту с ограничениями по номеру раздела и типу атрибута

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjForGeomWithLimit.

Синтаксис:

reference ksGetSpcObjForGeomWithLimit (char * nameLib,

unsigned int numb,

reference obj,

unsigned char equal,

unsigned char first,

unsigned short section,

double attrTypeNumb);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
obj	- указатель на графический объект или группу объектов (если obj = 0, функция работает с макроэлементом редактирования),
equal	- признак вхождения геометрии в состав объекта: 1 - геометрия идентична геометрии объекта спецификации, 0 - геометрия входит в объект спецификации,
first	- 1 -первый объект, 0 - следующий объект
section	- номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb	- номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

Примечания:

1. Данный метод - только для графических документов.
2. В настоящий момент метод реализован только для equal = 1.

3. При использовании Unicode следует использовать функцию ksGetSpcObjForGeomWithLimitW.

ksGetSpcObjForGeomWithLimitW – Получить указатель на объект спецификации, подключенный к данному графическому объекту с ограничениями по номеру раздела и типу атрибута (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjForGeomWithLimit.

Синтаксис:

reference ksGetSpcObjForGeomWithLimitW (LPWSTR nameLib,

unsigned int numb,

reference obj,

unsigned char equal,

unsigned char first,

unsigned short section,

double attrTypeNumb);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
numb	- номер стиля спецификации в библиотеке,
obj	- указатель на графический объект или группу объектов (если obj = 0, функция работает с макроэлементом редактирования),
equal	- признак вхождения геометрии в состав объекта: 1 - геометрия идентична геометрии объекта спецификации, 0 - геометрия входит в объект спецификации,
first	- 1 -первый объект, 0 - следующий объект
section	- номер раздела, в котором нужно найти объект (0 - номер не ограничивается),
attrTypeNumb	- номер типа атрибута, для которого нужно найти объект (0 - номер не ограничивается).

Возвращаемое значение:

- указатель на объект спецификации.

Примечания:

1. Данный метод - только для графических документов.
2. В настоящий момент метод реализован только для equal = 1.
3. При использовании ANSI следует использовать функцию ksGetSpcObjForGeomWithLimit.

ksGetSpcObjGeometry – Получить деталь или подсборку, подключенную к объекту спецификации

Справка системы КОМПАС...

KOMPAS.chm::/1183_135_5_3_Prosmotr_geometrii.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjGeometry.

Синтаксис:

reference ksGetSpcObjGeometry (reference spcObj);

Входные параметры:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

указатель на группу - в случае успешного завершения,
0 - в случае если геометрия не подключена или в случае ошибки.

Примечания:

Функция устарела, рекомендуется вместо нее использовать функцию ksGetSpcObjGeometryEx.

ksGetSpcObjGeometryEx – Получить графический объект, подключенный к объекту спецификации

Справка системы КОМПАС...

KOMPAS.chm::/1183_135_5_3_Prosmotr_geometrii.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjGeometryEx.

Синтаксис:

reference ksGetSpcObjGeometryEx (reference spcObj, int geomMode);

Входные параметры:

spcObj - указатель на объект спецификации,
geomMode 0 - только геометрия;
1 - только линии выноски;
2 - геометрия и линии выноски.

Возвращаемое значение:

указатель на группу - в случае успешного завершения,
0 - в случае, если геометрия не подключена или в случае ошибки.

ksGetSpcObjectSummaryCount – Получить суммарное количество для одинаковых объектов

Справка системы КОМПАС...

KOMPAS.chm::/1218_139_2_Podschet_summy_znach.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectSummaryCount.

Синтаксис:

double ksGetSpcObjectSummaryCount (reference spcObj,
int ispoln,
int blockNumber);

Входные параметры:

spcObj	- объект спецификации,
ispoln	- номер колонки типа "количество", начиная с 1 (актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А).

Возвращаемое значение:

- суммарное количество.

ksGetSpcPerformanceName – Получить отображаемое имя исполнения

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcPerformanceName.

Синтаксис:

int LIB_FUNC ksGetSpcPerformanceName(int index,
int ispoln,
int blockNumber,
LPSTR name,
int nameLen);

Входные параметры:

index	- индекс описания СП в документе,
ispoln	- номер колонки типа "количество", начиная с 1(актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А),
nameLen	- длина буфера строки.

Выходные параметры:

name - имя исполнения.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Для групповой спецификации А или Б у объекта спецификации может быть несколько исполнений.
2. Количество исполнений настраивается в настройке стиля спецификации. По умолчанию имя исполнения формируется из номера исполнения: "-01", "-02" и т.д. Имя исполнения можно отредактировать вручную.
3. Функция предназначена для получения имени исполнения.
4. При использовании Unicode следует использовать функцию ksGetSpcPerformanceNameW.

ksGetSpcPerformanceNameW – Получить отображаемое имя исполнения (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcPerformanceName.

Синтаксис:

```
int LIB_FUNC ksGetSpcPerformanceNameW( int index,
int ispoln,
int blockNumber,
LPWSTR name,
int   nameLen );
```

Входные параметры:

index - индекс описания СП в документе,
ispoln - номер колонки типа "количество", начиная с 1(актуально для СП типа Б),
blockNumber - номер блока, начиная с 0 (актуально для СП типа А),
nameLen - длина буфера строки.

Выходные параметры:

name - имя исполнения.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Для групповой спецификации А или Б у объекта спецификации может быть несколько исполнений.
2. Количество исполнений настраивается в настройке стиля спецификации. По умолчанию имя исполнения формируется из номера исполнения: "-01", "-02" и т.д. Имя исполнения можно отредактировать вручную.
3. Функция предназначена для получения имени исполнения.
4. При использовании ANSI следует использовать функцию ksGetSpcPerformanceName.

ksGetSpcPropertyFill – Получить признак синхронизации текстов объекта спецификации со свойствами

Синтаксис:

long LIB_FUNC ksGetSpcPropertyFill(reference spcObj);

Возвращаемое значение:

1	- флаг Синхронизировать со свойствами компонента взведен,
0	- флаг Синхронизировать со свойствами компонента не взведен.

ksGetSpcSectionName – Получить название раздела спецификации, которому принадлежит указанный объект спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcSectionName.

Синтаксис:

int ksGetSpcSectionName (reference spcObj, char* name, int size);

Входной параметр:

spcObj	- указатель на объект спецификации.

Выходные параметры:

name	- буфер для имени раздела,
size	- размер буфера.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksGetSpcSectionNameW.

ksGetSpcSectionNameW – Получить название раздела спецификации, которому принадлежит указанный объект спецификации (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcSectionName.

Синтаксис:

int ksGetSpcSectionNameW (reference spcObj, LPWSTR name, int size);

Входной параметр:

spcObj - указатель на объект спецификации.

Выходные параметры:

name - буфер для имени раздела,
size - размер буфера.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksGetSpcSectionName.

ksGetSpcSheetSB – Получить динамический массив листов сборочного чертежа, подключенных к спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpcDocument::ksGetSpcSheetSB.

Синтаксис:

reference ksGetSpcSheetSB (reference spcDoc);

Входной параметр:

spcDoc - указатель на спецификацию.

Возвращаемое значение:

указатель на динамический массив листов сборочного чертежа типа CHAR_STR_ARR - в случае успеха,
0 - в случае неудачи.

ksGetSpcSheetSB_W – Получить динамический массив листов сборочного чертежа, подключенных к спецификации (Unicode)

Аналог данной функции при использовании Automation - метод ksSpcDocument::ksGetSpcSheetSB.

Синтаксис:

reference ksGetSpcSheetSB_W (reference spcDoc);

Входной параметр:

spcDoc - указатель на спецификацию.

Возвращаемое значение:

указатель на динамический массив листов сборочного чертежа типа CHAR_STR_ARR_W. - в случае успеха,
0 - в случае неудачи.

ksGetSpcStyleParam – Получить параметры указанного стиля спецификации из заданной библиотеки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_LIB_STYLE_DIALOG.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcStyleParam.

Синтаксис:

int LIB_FUNK ksGetSpcStyleParam (char * nameLib,

unsigned int numb,

void * par,

unsigned int size,

int tPar);

Входные параметры:

nameLib - имя библиотеки стилей спецификаций
(NULL - параметры берутся у текущего документа),
numb - номер стиля спецификации в библиотеке,
tPar - тип возвращаемых параметров.

Выходной параметр:

par - структура параметров стиля спецификации,
size - размер структуры параметров.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Если tPar =ALLPARAM, то заполняется структура параметров стиля спецификации SpcStyleParam....
2. Если tPar = SPC_TUNING_PARAM, то заполняется заполняется структура параметров настроек спецификации SpcTuningStyleParam....
3. При использовании Unicode следует использовать функцию ksGetSpcStyleParamW.

ksGetSpcStyleParamW – Получить параметры указанного стиля спецификации из заданной библиотеки (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_LIB_STYLE_DIALOG.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcStyleParam.

Синтаксис:

```
int LIB_FUNK ksGetSpcStyleParamW (LPWSTR nameLib,
unsigned int numb,
void * par,
unsigned int size,
int tPar);
```

Входные параметры:

nameLib - имя библиотеки стилей спецификаций
(NULL - параметры берутся у текущего документа),
numb - номер стиля спецификации в библиотеке,
tPar - тип возвращаемых параметров.

Выходной параметр:

par	- структура параметров стиля спецификации,
size	- размер структуры параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если tPar =ALLPARAM, то заполняется структура параметров стиля спецификации SpcStyleParam....
2. Если tPar = SPC_TUNING_PARAM, то заполняется заполняется структура параметров настроек спецификации SpcTuningStyleParam....
3. При использовании ANSI следует использовать функцию ksGetSpcStyleParam.

ksGetSpcTableColumn – Получить количество колонок для стиля спецификации в текущем документе

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcTableColumn.

Синтаксис:

int ksGetSpcTableColumn (char * nameLib,
unsigned int numb,
unsigned char additioanalCol);

Входные параметры:

nameLib	- имя библиотеки стилей спецификаций,
numb	- номер стиля в библиотеке,
additioanalCol	- признак колонок: 0 - колонки таблицы спецификации, 1 - дополнительные колонки.

Возвращаемое значение:

количество колонок для стиля спецификации в текущем документе.	-в случае успеха,
0	- в случае неудачи.

Примечания:

1. Если документ - спецификация, то параметры nameLib и numb не используются.
2. Если количество исполнений больше числа колонок "количество" в бланке спецификации, то количество колонок выдается с учетом всех исполнений.

ksGetTuningSpcStyleParam – Получить параметры настройки спецификации документа

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetTuningSpcStyleParam.

Синтаксис:

int ksGetTuningSpcStyleParam (reference pDoc,
int index,
SpcTuningStyleParam * par);

Входные параметры:

pDoc	- указатель на документ,
index	- индекс описания спецификации в документе,
par	- структура параметров SpcTuningStyleParam.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если pDoc == 0, то изменяются параметры настроек для текущего документа;
2. Если index == -1, то изменяются параметры настроек текущего описания спецификации;
3. Для документа-спецификации настройки можно заменить всегда; для графического документа: в случае, если документ подключен к спецификации или включен режим "Спецификация на листе"; для 3d документа: в случае, если документ подключен к спецификации.
4. При использовании Unicode следует использовать функцию ksGetTuningSpcStyleParamW.

ksGetTuningSpcStyleParamW – Получить параметры настройки спецификации документа (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetTuningSpcStyleParam.

Синтаксис:

int ksGetTuningSpcStyleParamW (reference pDoc,
int index,
SpcTuningStyleParamW * par);

Входные параметры:

pDoc	- указатель на документ,
index	- индекс описания спецификации в документе,
par	- структура параметров SpcTuningStyleParamW.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если pDoc == 0, то изменяются параметры настроек для текущего документа;
2. Если index == -1, то изменяются параметры настроек текущего описания спецификации;
3. Для документа-спецификации настройки можно заменить всегда; для графического документа: в случае, если документ подключен к спецификации или включен режим "Спецификация на листе"; для 3d документа: в случае, если документ подключен к спецификации.
4. При использовании ANSI следует использовать функцию ksGetTuningSpcStyleParam.

ksGetWidthColumnSpc – Получить ширину колонки или текста в колонке

Аналог данного метода при использовании Automation - метод ksSpecification::ksGetWidthColumnSpc.

Синтаксис Automation:

double ksGetWidthColumnSpc (int numColumn,

unsigned char cellOrText)

Входные параметры:

numColumn	- номер колонки,
cellOrText	- признак объекта, ширину которого требуется узнать: 1 - колонка, 0 - текст в колонке.

Возвращаемое значение:

ширина колонки или текста в колонке.

Примечание:

Номер колонки должен начинаться с единицы.

ksIsModuleSpecificationActive – Проверить, разрешена ли работа со спецификацией

Аналог данной функции при использовании Automation - метод KompasObject::ksIsModuleSpecificationActive

Синтаксис:

int ksIsModuleSpecificationActive();

Возвращаемое значение:

1 - работа со спецификацией разрешена,
0 - работа со спецификацией запрещена.

Примечание:

Если ksIsModuleSpecificationActive == FALSE, работа со спецификацией запрещена.

Для включения работы со спецификацией используйте функцию ksModuleSpecification. Эта же функция позволяет отключить работу со спецификацией.

ksIsSlaveSpcOpened – Проверить, открыто ли окно спецификации в подчиненном режиме

Справка системы КОМПАС...

KOMPAS.chm::/1155_132_3_Podchinennyj_rezhim.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsSlaveSpcOpened.

Синтаксис:

int ksIsSlaveSpcOpened (reference doc);

Входные параметры:

doc - указатель на документ.

Возвращаемое значение:

0 - окно спецификации в Slave режиме не открыто,
1 - окно спецификации в Slave режиме открыто,
2 - окно спецификации в Slave режиме открыто и активно.

Примечание.

Если doc = 0, метод вызывается для активного документа.

ksModuleSpecification – Управление возможностью работы со спецификацией

Аналог данной функции при использовании Automation - метод KompasObject::ksModuleSpecification.

Синтаксис:

int ksModuleSpecification (unsigned char attach);

Входной параметр:

attach	признак выполняемого действия: 1 - включить работу со спецификацией, 0 - выключить работу со спецификацией.

Возвращаемое значение:

1	- если был инициирован процесс включения или отключения возможности работы со спецификацией,
0	- в случае неудачного завершения.

Примечание:

Проверить, включена или отключена работа со спецификацией, можно с помощью функции ksIsModuleSpecificationActive.

5. При использовании ANSI следует использовать функцию ksSpcObjectCreate.

ksSetApplyLibrarySpcStyle –Установить признак использования стиля спецификации

Синтаксис:

int LIB_FUNC ksSetApplyLibrarySpcStyle(int librarySpcStyle);

Входные параметры:

librarySpcStyle	1 - использовать стиль СП из библиотеки, а не втянутый в документ.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSetCurrentSpcObject – Установить текущий объект спецификации по указателю на него либо по его индексу

Аналог данного метода при использовании Automation - метод ksSetCurrentSpcObject.

Синтаксис:

long ksSetCurrentSpcObject (long spcObj, long index);

Входные параметры:

spcObj	- указатель на объект,
index	- индекс объекта.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Метод работает для видимых таблиц спецификации.

ksSetSpcDescription – Установить параметры описания спецификации по индексу для указанного документа

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SPC_DESCRIBE.htm`

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcDescription.

Синтаксис:

int ksSetSpcDescription (reference pDoc,

int index,

SpcDescrParam* param,

unsigned char state);

Входные параметры:

pDoc	- указатель на документ,
index	- индекс (номер) описания спецификации в документе (-1 - текущее описание),
param	- указатель на параметры описания спецификации,
state	- состояние описания спецификации:
	TRUE - текущее,
	FALSE - не текущее.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Структура параметров описания спецификации....

Примечания:

1. Метод изменяет параметры описания спецификации и/или делает его активным, если state = TRUE.
2. Если param = NULL - меняется только состояние.
3. Для документа-спецификации параметр index не используется.
4. При использовании Unicode следует использовать функцию ksSetSpcDescriptionW.

ksSetSpcDescriptionW – Установить параметры описания спецификации по индексу для указанного документа (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SPC_DESCRIBE.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcDescription.

Синтаксис:

int ksSetSpcDescriptionW (reference pDoc,

int index,

SpcDescrParamW* param,

unsigned char state);

Входные параметры:

pDoc	- указатель на документ,
index	- индекс (номер) описания спецификации в документе (-1 - текущее описание),
param	- указатель на параметры описания спецификации,
state	- состояние описания спецификации:
	TRUE - текущее,
	FALSE - не текущее.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Структура параметров описания спецификации....

Примечания:

1. Метод изменяет параметры описания спецификации и/или делает его активным, если state = TRUE.
2. Если param = NULL - меняется только состояние.
3. Для документа-спецификации параметр index не используется.
4. При использовании ANSI следует использовать функцию ksSetSpcDescription.

ksSetSpcObjectAttributeNumber – Установить номер атрибута объекта спецификации

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcObjectAttributeNumber.

Синтаксис:

int ksSetSpcObjectAttributeNumber (reference spcObj, double attrNumber);

Входные параметры:

spcObj	- объект спецификации,
attrNumber	- номер атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.- в случае неудачи.

Примечания:

Если spcObj = 0, то возвращается номер редактируемого в текущий момент объекта спецификации, т.е. внутри блока:

ksSpcObjectEdit(...); .

...

...

ksSpcObjectEnd();

ksSetSpcObjectColumnText – Установить текст колонки определенного типа заданного исполнения объекта спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcObjectColumnText.

Синтаксис:

int ksSetSpcObjectColumnText (unsigned int columnType,

unsigned int ispoln,

unsigned int block,

char *s);

Входные параметры:

columnType	- тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер исполнения данного типа,

block	- номер блока исполнений, начиная с 0 (если количество исполнений меньше количества исполнений в бланке спецификации),
s	- строка, из которой нужно взять текст.

Типы колонок спецификации...

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Для групповой спецификации 2.113-75 число исполнений меньше 10.
2. Функция работает в режиме создания нового или редактирования существующего объекта спецификации ksSpcObjectCreate или ksSpcObjectEdit;

```
//...
//   ksSetSpcObjectColumnText(...);
//...
// reference spsObj = ksSpcObjectEnd();
```

3. При использовании Unicode следует использовать функцию ksSetSpcObjectColumnTextW.

ksSetSpcObjectColumnTextEx – Задать текст объекта спецификации для определенного типа колонки и исполнения

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcObjectColumnTextEx.

Синтаксис:

```
int ksSetSpcObjectColumnTextEx (unsigned int columnType,
unsigned int ispoln,
unsigned int block,
reference p);
```

Входные параметры:

columnType	тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	номер колонки данного типа, начиная с 1,
block	номер блока,
p	динамический массив строк текста TEXT_LINE_ARR.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае ошибки.

Примечание:

Функция работает в режиме создания нового или редактирования существующего объекта спецификации ksSpcObjectCreate или ksSpcObjectEdit
... ksSetSpcObjectColumnTextEx(...); reference spsObj = ksSpcObjectEnd();.

ksSetSpcObjectColumnTextW – Установить текст колонки определенного типа заданного исполнения объекта спецификации (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcObjectColumnText.

Синтаксис:

int ksSetSpcObjectColumnTextW (unsigned int columnType,

unsigned int ispoln,

unsigned int block,

LPWSTR s);

Входные параметры:

columnType	- тип колонки SPC_CLM_FORMAT...SPC_CLM_USER,
ispoln	- номер исполнения данного типа,
block	- номер блока исполнений, начиная с 0 (если количество исполнений меньше количества исполнений в бланке спецификации),
s	- строка, из которой нужно взять текст.

Типы колонок спецификации...

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Для групповой спецификации 2.113-75 число исполнений меньше 10.
2. Функция работает в режиме создания нового или редактирования существующего объекта спецификации

   ksSpcObjectCreate или ksSpcObjectEdit (Unicode)

   //...

   //ksSetSpcObjectColumnTextW(...);

   //...

   // reference spsObj = ksSpcObjectEnd();
3. При использовании ANSI следует использовать функцию ksSetSpcObjectColumnText.

ksSetSpcObjectColumnTextAlign – Установить выравнивание для текста колонки объекта спецификации

Аналог данной функции при использовании Automation - метод ksSpecification::ksGetSpcObjectSummaryCount.

Синтаксис:

int ksSetSpcObjectColumnTextAlign (reference spcObj,
int columnNumber,
int lineIndex,
int align);

Входной параметр:

spcObj	- указатель на объект спецификации,
columnNumber	- номер колонки, начиная с 1,
lineIndex	- индекс строки текста,
align	- выравнивание.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

Значение переменной align может принимать значения:

- 0 - Выравнивание влево
- 1 - Выравнивание по центру
- 2 - Выравнивание вправо
- 3 - Выравнивание по ширине

ksSetSpcPerformanceName – Установить отображаемое имя исполнения

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcPerformanceName.

Синтаксис:

int LIB_FUNC ksSetSpcPerformanceName(int index,
int ispoln,
int blockNumber,
LPSTR name,);

Входные параметры:

Index	- индекс описания СП в документе,

Ispoln	- номер колонки типа "количество", начиная с 1(актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А),
Name	- имя исполнения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Для групповой спецификации А или Б у объекта спецификации может быть несколько исполнений.
2. Количество исполнений настраивается в настройке стиля спецификации. По умолчанию имя исполнения формируется из номера исполнения: "-01", "-02" и т.д. Имя исполнения можно отредактировать вручную. В текст обозначения исполнения, которое редактируется, нужно включать разделитель.
3. Функция предназначена для получения имени исполнения.
4. При использовании Unicode следует использовать функцию ksSetSpcPerformanceNameW.

ksSetSpcPerformanceNameW – Установить отображаемое имя исполнения (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetSpcPerformanceName.

Синтаксис:

```
int LIB_FUNC ksSetSpcPerformanceName(int index,
int ispoln,
int blockNumber,
LPWSTR name,);
```

Входные параметры:

Index	- индекс описания СП в документе,
Ispoln	- номер колонки типа "количество", начиная с 1(актуально для СП типа Б),
blockNumber	- номер блока, начиная с 0 (актуально для СП типа А),
name	- имя исполнения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Для групповой спецификации А или Б у объекта спецификации может быть несколько исполнений.
2. Количество исполнений настраивается в настройке стиля спецификации. По умолчанию имя исполнения формируется из номера исполнения: "-01", "-02" и т.д. Имя исполнения можно отредактировать вручную.
3. Функция предназначена для получения имени исполнения.
4. При использовании ANSI следует использовать функцию ksSetSpcPerformanceName.

ksSetSpcPropertyFill – Установить признак синхронизации текстов объекта спецификации со свойствами

Синтаксис:

int LIB_FUNC ksSetSpcPropertyFill(reference spcObj, long val);

Входные параметры:

spcObj	- указатель на объект спецификации,
val	- 1 - флаг Синхронизировать со свойствами компонента взведен - 0 - флаг Синхронизировать со свойствами компонента не взведен.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksSetSpcSheetSB – Установить динамический массив листов сборочного чертежа, подключенных к спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpcDocument::ksSetSpcSheetSB.

Синтаксис:

int ksSetSpcSheetSB (reference spcDoc, reference arr);

Входной параметр:

spcDoc	- указатель на спецификацию,
arr	- указатель на динамический массив типа CHAR_STR_ARR или CHAR_STR_ARR_W.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksSetTuningSpcStyleParam – Установить параметры настроек спецификации документа

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetTuningSpcStyleParam.

Синтаксис:

int ksSetTuningSpcStyleParam (reference pDoc,
int index,
SpcTuningStyleParam * par);

Входные параметры:

pDoc - указатель на документ,
index - индекс описания спецификации в документе,
par - структура параметров SpcTuningStyleParam.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если pDoc == 0, то изменяются параметры настроек для текущего документа;
2. Если index == -1, то изменяются параметры настроек текущего описания спецификации;
3. Для документа-спецификации настройки можно заменить всегда;
4. Для графического документа настройки можно заменить в случае, если документ подключен к спецификации или включен режим "Спецификация на листе";
5. Для документа-модели настройки можно заменить в случае, если документ подключен к спецификации.
6. При использовании Unicode следует использовать функцию ksSetTuningSpcStyleParamW.

ksSetTuningSpcStyleParamW – Установить параметры настроек спецификации документа (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/1300_Glava145_Stilq_specifikaci.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSetTuningSpcStyleParam.

Синтаксис:

int ksSetTuningSpcStyleParamW (reference pDoc,
int index,
SpcTuningStyleParamW * par);

Входные параметры:

pDoc	- указатель на документ,
index	- индекс описания спецификации в документе,
par	- структура параметров SpcTuningStyleParamW.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если pDoc == 0, то изменяются параметры настроек для текущего документа;
2. Если index == -1, то изменяются параметры настроек текущего описания спецификации;
3. Для документа-спецификации настройки можно заменить всегда;
4. Для графического документа настройки можно заменить в случае, если документ подключен к спецификации или включен режим "Спецификация на листе";
5. Для документа-модели настройки можно заменить в случае, если документ подключен к спецификации.
6. При использовании ANSI следует использовать функцию ksSetTuningSpcStyleParam.

ksSpcChangeValue – Изменить значение компонента с указанным номером в указанной колонке

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcChangeValue.

Синтаксис:

int ksSpcChangeValue (unsigned int colNumb,
unsigned int itemNumb,

void *val,
unsigned char typeVal);

Входные параметры:

colNumb	- номер колонки, начиная с единицы,
itemNumb	- номер компоненты, начиная с единицы,
val	- указатель на значение,
typeVal	- тип данных.

Типы данных...

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.- в случае неудачи.

Описание:

Функция позволяет изменить значение компонента с номером itemNumb в колонке с номером colNumb.

Если в колонке содержатся данные типа *Запись*, она может содержать несколько компонентов. При этом itemNumb может быть не равен 1.

Значение параметра typeVal - может лежать в интервале CHAR_ATTR_TYPE ...STRING_ATTR_TYPE (см. ltdefine.h). Если typeVal== STRING_ATTR_TYPE, то val - строка char[MAX_TEXT_LENGTH].

Функция работает в режиме создания нового или редактирования существующего объекта спецификации ksSpcObjectCreate; или ksSpcObjectEdit;

.....

ksSpcChangeValue();

.....

reference spsObj = ksSpcObjectEnd();

Примечание.

При использовании Unicode следует использовать функцию ksSpcChangeValueW.

ksSpcChangeValueW – Изменить значение компоненты с указанным номером в указанной колонке (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcChangeValue.

Синтаксис:

int ksSpcChangeValueW (unsigned int colNumb,
unsigned int itemNumb,
void* val,
unsigned char typeVal);

Входные параметры:

colNumb - номер колонки, начиная с единицы,
itemNumb - номер компоненты, начиная с 1,
val - указатель на значение,
typeVal - тип данных.

Типы данных...

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Описание:

Функция позволяет изменить значение компоненты с номером itemNumb в колонке с номером colNumb.

Если в колонке содержатся данные типа *Запись*, она может содержать несколько компонент. При этом itemNumb может быть не равен 1.

Значение параметра typeVal - может лежать в интервале CHAR_ATTR_TYPE ...STRING_ATTR_TYPE (см. ltdefine.h). Если typeVal== STRING_ATTR_TYPE, то val - строка wchar_t[MAX_TEXT_LENGTH].

Функция работает в режиме создания нового или редактирования существующего объекта спецификации ksSpcObjectCreate; или ksSpcObjectEdit;

.....

ksSpcChangeValueW();

.....

reference spsObj = ksSpcObjectEnd();

Примечание.

При использовании ANSI следует использовать функцию ksSpcChangeValue.

ksSpcCount – Установить количество деталей для определенного исполнения

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcCount.

Синтаксис:

int ksSpcCount (unsigned char ispoln, char * sCount);

Входные параметры:

ispoln - номер исполнения, начиная с 1,
sCount - количество деталей.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка с типом *Количество*),
0 - в случае неудачи.

Примечание:

1. Функция работает в режиме создания нового или редактирования существующего объекта спецификации.

   ksSpcObjectCreate или ksSpcObjectEdit;

   .....

   ksSpcCount(...);

   .....

   reference spsObj = ksSpcObjectEnd();

2. При использовании Unicode следует использовать функцию ksSpcCountW.

ksSpcCountW – Установить количество деталей для определенного исполнения (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcCount.

Синтаксис:

int ksSpcCountW (unsigned char ispoln, LPWSTR sCount);

Входные параметры:

ispoln - номер исполнения, начиная с 1,
sCount - количество деталей.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка с типом *Количество*).
0 - в случае неудачи.

Примечание:

1. Функция работает в режиме создания нового или редактирования существующего объекта спецификации.

   ksSpcObjectCreatek или ksSpcObjectEdit;

   .....

   ksSpcCountW(...);

   .....

   reference spsObj = ksSpcObjectEnd();

2. При использовании ANSI следует использовать функцию ksSpcCount.

ksSpcDocLinksClear – Удалить связки с документом-владельцем объекта спецификации

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcDocLinksClear.

Синтаксис:

int ksSpcDocLinksClear(reference doc);

Входные параметры:

doc	- указатель на документ-источник объекта спецификации (графический, 3D, спецификация).

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

Функция удаляет связки документа-источника объекта спецификации (графический, 3D, спецификация) с документом-владельцем объекта спецификации;

Если doc = 0, то документом-источником объекта спецификации считается текущий документ.

ksSpcDocLinksClearEx – Удалить связки документа-источника с документом-владельцем объекта спецификации

Синтаксис:

int LIB_FUNC ksSpcDocLinksClearEx(reference pDoc, int mode);

Входные параметры:

doc	- указатель на документ-источник объекта спецификации (графический, 3D, спецификация),
mode	- режим удаления ссылок.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Функция удаляет связки документа-источника объекта спецификации (графический, 3D, спецификация) с документом-владельцем объекта спецификации.
2. Если doc = 0, то документом-источником объекта спецификации считается текущий документ.

3. Если mode = 1, то удаляются все ссылки.
4. Если mode = 0, то удаляются ссылки на не найденные файлы.

ksSpcIncludeReference – Добавить или изменить геометрию или линии-выноски в объекте спецификации

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1182_135_5_1_Vkljuchenie_geomet.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcIncludeReference.

Синтаксис:

int ksSpcIncludeReference (reference obj, unsigned char clear);

Входные параметры:

obj - группа объектов или объект вида,
clear - признак обработки существующей геометрии объекта спецификации:
0 - добавить новые объекты к существующей геометрии,
1 - удалить геометрию,
2 - очистить список линий выносок.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

Если obj == 0 и clear == 0, очищается список объектов и список линий выносок.

ksSpcMassa – Установить массу детали

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcMassa.

Синтаксис:

int ksSpcMassa (char * sMassa);

Выходной параметр:

sMassa - строка с массой детали.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка с типом *Масса*),
0 - в случае неудачи.

Примечание:

1. Функция работает в режиме создания нового или редактирования существующего объекта спецификации.

   ksSpcObjectCreate или ksSpcObjectEdit;

   .....

   ksSpcMassa(...);

   .....

   reference spsObj = ksSpcObjectEnd();
2. При использовании Unicode следует использовать функцию ksSpcMassaW.

ksSpcMassaW – Установить массу детали (Unicode)

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcMassa.

Синтаксис:

int ksSpcMassaW (LPWSTR sMassa);

Выходной параметр:

sMassa - строка с массой детали.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка типа *Масса*),
0 - в случае неудачи.

Примечание:

1. Функция работает в режиме создания нового или редактирования существующего объекта спецификации.

   ksSpcObjectCreate или ksSpcObjectEdit;

   .....

   ksSpcMassaW(...);

   .....

   reference spsObj = ksSpcObjectEnd();
2. При использовании ANSI следует использовать функцию ksSpcMassa.

ksSpcObjectCreate – Создать объект спецификации в графическом документе

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1195_136_1_Sozdanie_obwektov_sp.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcObjectCreate.

Синтаксис:

int ksSpcObjectCreate (char * nameLib,

unsigned int styleNumb,

unsigned short secNumb,

unsigned short subSecNumb,

double numb,

unsigned char typeObj);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
styleNumb	- номер стиля спецификации в библиотеке,
secNumb	- номер раздела,
subSecNumb	- номер подраздела,
numb	- для базового объекта - тип атрибута, который задан по ключам, или 0, для вспомогательного объекта - номер базового объекта, к которому прикрепляется вспомогательный, либо 0,
typeObj	- тип строки спецификации (0- базовый объект, 1 – вспомогательный объект).

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Объект спецификации - составной объект. Его создание завершается функцией ksSpcObjectEnd.
2. При использовании Unicode следует использовать функцию ksSpcObjectCreateW.

ksSpcObjectCreateW – Создать объект спецификации в графическом документе (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/1195_136_1_Sozdanie_obwektov_sp.htm

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcObjectCreate.

Синтаксис:

int ksSpcObjectCreateW (LPWSTR * nameLib,

unsigned int styleNumb,

unsigned short secNumb,

unsigned short subSecNumb,

double numb,

unsigned char typeObj);

Входные параметры:

nameLib	- имя библиотеки стилей спецификации,
styleNumb	- номер стиля спецификации в библиотеке,
secNumb	- номер раздела,
subSecNumb	- номер подраздела,
numb	- для базового объекта – тип атрибута, который задан по ключам, или 0, для вспомогательного объекта - номер базового объекта, к которому прикрепляется вспомогательный, либо 0,
typeObj	- тип объекта спецификации (0- базовый объект, 1 – вспомогательный объект).

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Объект спецификации - составной объект. Его создание завершается функцией ksSpcObjectEnd.

ksSpcObjectEdit – Редактировать объект спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcObjectEdit.

Синтаксис:

int ksSpcObjectEdit (reference spcObj);

Входной параметр:

spcObj - указатель на объект спецификации.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

1. После вызова данного метода у объекта можно изменить атрибуты, геометрию, линии выноски, позицию, массу.
2. Редактирование объекта завершится вызовом функции ksSpcObjectEnd.

ksSpcObjectEnd – Завершить создание или редактирование объекта спецификации

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcObjectEnd.

Синтаксис:

reference ksSpcObjectEnd();

Возвращаемое значение:

- указатель на объект спецификации.

Примечание:

1. Все изменения будут занесены в документ.
2. Если объект новый, он подключается к соответствующему описанию спецификации текущего документа.

ksSpcPosition – Установить номер позиции

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcPosition.

Синтаксис:

int ksSpcPosition (unsigned short pos);

Входной параметр:

pos - номер позиции.

Возвращаемое значение:

1 - в случае удачного завершения (если в объекте существует колонка с типом *Позиция*),
0 - в случае неудачи.

Примечания:

Функция работает в режиме создания нового или редактирования существующего объекта спецификации, например:

ksSpcObjectCreate(...); или ksSpcObjectEdit(...);

.....

ksSpcPosition(...);

.....

reference spsObj = ksSpcObjectEnd();

ksSpcVisible – Установить признак видимости компонента

Пример...

Аналог данной функции при использовании Automation - метод ksSpecification::ksSpcVisible.

Синтаксис:

int ksSpcVisible (unsigned int colNumb,
unsigned int itemNumb,
unsigned char flagOn);

Входные параметры:

colNumb - номер колонки, начиная с единицы,
itemNumb - номер компоненты, начиная с единицы,
flagOn - признак видимости компонента:
1 -включить,
0- выключить.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

1. Функция работает в режиме создания нового или редактирования существующего объекта спецификации.
2. В случае записи в колонке может быть несколько компонент и itemNumb может быть не равен 1.

ksSpecificationOnSheet – Включить или выключить показ спецификации на листе чертежа

Справка системы КОМПАС...

KOMPAS.chm::/CM_SHEETSPC.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSpecificationOnSheet.

Синтаксис:

int ksSpecificationOnSheet (unsigned char onSheet);

Входной параметр:

onSheet	- признак размещения спецификации на листе 1 - включено, 0 - выключено.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Функции работы с документами

Общие функции позволяют выполнять операции с документами - создавать, открывать, сохранять и т.п., а также управлять его печатью.

Функции навигации по объектам обеспечивают обработку объектов чертежа - поиск объекта по характерной точке, вычисление его габаритов, а также навигацию по графической модели чертежа. Использование функций работы с моделью позволяет проводить специализированную обработку изображения (например, перебрать все объекты вида и выдать их во внешний файл специального формата).

Атрибут - дополнительная информация произвольной структуры, сохраняемая с любым объектом чертежа, начиная от отрезка и заканчивая видом, штампом или техническими требованиями. Каждый объект может иметь произвольное количество атрибутов. Описания типов атрибутов могут храниться как отдельной специальной библиотеке, так и в конкретном чертеже. Атрибуты используются для хранения в чертеже негеометрической информации с целью ее дальнейшей обработки в конкретном приложении. Функции работы с атрибутами позволяют обрабатывать атрибуты объектов.

Функции обработки параметрических переменных и параметрических связей обеспечивают работу с параметрическими переменными в графическом документе.

Функции работы с настройками и стилями документа позволяют получать и изменять настройки системы, документа и примененные стили.

Общие функции

CreateDocument – Создать документ (чертеж, фрагмент, текстовый документ, деталь, сборку)

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateDocument.

Синтаксис:

reference LIB_FUNC CreateDocument (DocumentParam * par);

Выходной параметр:

par - указатель на структуру параметров документа DocumentParam.

Возвращаемое значение:

указатель на документ - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Документ может быть открыт как в видимом режиме, так и в невидимом (в этом случае документ не виден на экране, и все операции над ним скрыты от пользователя).
2. Созданный документ автоматически становится текущим.
3. Документ не будет создан в случае, если уже открыт документ с таким же именем.
4. При использовании Unicode следует использовать функцию CreateDocumentW.

CreateDocumentW – Создать документ (чертеж, фрагмент, текстовый документ, деталь, сборку) (Unicode)

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCreateDocument.

Синтаксис:

reference LIB_FUNC CreateDocumentW (DocumentParamW * par);

Выходной параметр:

par - указатель на структуру параметров документа DocumentParamW.

Возвращаемое значение:

указатель на документ - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Документ может быть открыт как в видимом режиме, так и в невидимом (в этом случае документ не виден на экране и все операции над ним скрыты от пользователя).
2. Созданный документ автоматически становится текущим.
3. Документ не будет создан в случае, если уже открыт документ с таким же именем.
4. При использовании ANSI следует использовать функцию CreateDocument.

OpenDocument – Открыть документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/Glava_13_Otkritie_zakritie_dok.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenDocument.

Синтаксис:

reference OpenDocument (char *name, unsigned char regim);

Входные параметры:

regim	режим работы с документом: 0 - видимый, 1 - невидимый ("слепой") 3 - для спецификации - видимый без синхронизации со сборкой, 4 - для спецификации - "слепой" без синхронизации со сборкой.

Выходные параметры:

name	- полное имя файла открываемого документа,

Возвращаемое значение:

указатель на документ	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Невидимый режим применяется при пакетном (скрытом) формировании документа.
2. Открытый документ автоматически становится текущим.
3. При использовании Unicode следует использовать функцию OpenDocumentW.

OpenDocumentW – Открыть документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку) (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/Glava_13_Otkritie_zakritie_dok.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksOpenDocument.

Синтаксис:

reference LIB_FUNC OpenDocumentW (LPWSTR name, unsigned char regim);

Входные параметры:

regim	- режим работы с документом: 0 - видимый, 1 - невидимый ("слепой") 3 - для спецификации - видимый без синхронизации со сборкой, 4 - для спецификации - "слепой" без синхронизации со сборкой.

Выходные параметры:

name	- полное имя файла открываемого документа.

Возвращаемое значение:

указатель на документ	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Невидимый режим применяется при пакетном (скрытом) формировании документа.
2. Открытый документ автоматически становится текущим.
3. При использовании ANSI следует использовать функцию OpenDocument.

CloseDocument – Закрыть документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCloseDocument.

Синтаксис:

int CloseDocument (reference sheet);

Входные параметры:

sheet - указатель на закрываемый документ.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если sheet = 0, закрывается текущий документ.
2. Если документ не сохранен, взводится флаг соответствующей ошибки.

SaveDocument – Сохранить документ (чертеж, фрагмент, спецификацию, текстовый документ, деталь, сборку)

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSaveDocument.

Синтаксис:

int SaveDocument (reference sheet,

char * fileName);

Входные параметры:

sheet - указатель на сохраняемый документ.

Выходные параметры:

fileName - полное имя файла, в котором требуется сохранить документ.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Если sheet = 0, сохраняется текущий документ.
2. Если fileName - NULL, используется имя файла из документа (т.е. параметр fileName определяет имя файла при сохранении в режиме Save As...). Если и в документе имя файла также NULL, появляется ошибка.

SaveDocumentEx – Сохранить документ в выбранной версии

Справка системы КОМПАС...

KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSaveDocumentEx.

Синтаксис:

int SaveDocumentEx (reference doc, char * fileName, int version);

Входные параметры:

doc	- указатель на документ,
fileName	- полное имя файла,
version	- версия для сохранения: -1 - в предыдущую версию, 0 - в текущую версию, 1 - в версию 5.11.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если doc = 0, метод сохраняет текущий документ.
2. Если fileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.
3. При использовании Unicode следует использовать функцию SaveDocumentExW.

SaveDocumentExW – Сохранить документ в выбранной версии (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/Glava_12_Sozdanie_sohranenie_dok.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSaveDocumentEx.

Синтаксис:

int LIB_FUNC SaveDocumentExW (reference doc, LPWSTR fileName, int version);

Входные параметры:

doc	- указатель на документ,
fileName	- полное имя файла,

version	- версия для сохранения: -1 - в предыдущую версию, 0 - в текущую версию, 1 - в версию 5.11.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если doc = 0, метод сохраняет текущий документ.
2. Если fileName = NULL, то используется имя файла из документа. Если же и в документе отсутствует имя файла, то взводится ошибка.
3. При использовании ANSI следует использовать функцию SaveDocumentEx.

GetDocOptions – Получить настройки документа

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SID_SETUP_HELP_ID.htm`

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetDocOptions.

Синтаксис:

int GetDocOptions (int optionsType void * param, int sizePar);

Входной параметр:

optionsType	- тип настройки,
sizePar	- размер структуры параметров.

Выходной параметр:

param	- указатель на структуру параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

Примечание:

Функция позволяет получить настройки текущего документа (в настоящее время функция реализована только для настроек размеров) и заполнить ими соответствующую структуру.

SetDocOptions – Задать настройки документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SID_SETUP_HELP_ID.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetDocOptions.

Синтаксис:

int SetDocOptions (int optionsType void * param, int sizePar);

Входной параметр:

optionsType	- тип настройки,
sizePar	- размер структуры параметров.

Выходной параметр:

param	- указатель на структуру параметров.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Соответствие входных и выходных параметров...

Примечание:

Функция позволяет получить настройки текущего документа (в настоящее время функция реализована только для настроек размеров) и заполнить ими соответствующую структуру.

ksGetDocumentType – Получить тип документа

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetDocumentType.

Синтаксис:

int ksGetDocumentType (reference doc);

Входные параметры:

doc	- указатель на документ.

Возвращаемое значение:

тип документа	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель doc = 0, возвращается тип активного документа.

ksDrawKompasDocumentByReference – Отрисовать КОМПАС–документ как слайд в присланном окне

Аналог данной функции при использовании Automation - метод KompasObject::DrawKompasDocumentByReference.

Синтаксис:

int ksDrawKompasDocumentByReference (void *HWindow, reference pDoc);

Входные параметры:

Hwindow	- несущее окно,
pDoc	- указатель на документ.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Если указатель pdoc = 0, отрисовывается текущий документ.

ksGetDocumentTypeByName – Получить тип документа

Аналог данной функции при использовании Automation - метод KompasObject::GetDocumentTypeByName.

Синтаксис:

int ksGetDocumentTypeByName (char * fileName);

Входные параметры:

fileName	- полное имя файла документа.

Возвращаемое значение:

int	- тип документа,
0	- в случае неудачи.

Примечание:

1. Документ при вызове этой функции не создается.
2. Независимо от используемого оформления возвращается:

- для чертежа - lt_DocSheetStandart,
- для спецификации - lt_DocSpc,
- для текстового документа - lt_DocTxtStandart.

ksGetDocumentTypeByNameW – Получить тип документа (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::GetDocumentTypeByName.

Синтаксис:

int LIB_FUNC ksGetDocumentTypeByNameW (LPWSTR fileName);

Входные параметры:

fileName	- полное имя файла документа.

Возвращаемое значение:

int	- тип документа,
0	- в случае неудачи.

Примечание:

1. Документ при вызове этой функции не создается.
2. Независимо от используемого оформления возвращается:

- для чертежа - lt_DocSheetStandart,
- для спецификации - lt_DocSpc,
- для текстового документа - lt_DocTxtStandart.

ksGetDocumentSaveVersion – Текущая версия записи документов

Синтаксис:

int LIB_FUNC ksGetDocumentSaveVersion();

Возвращаемое значение:

	- номер версии из перечисления ksSaveDocumentVersionEnum.

Примечание:

Получать требуется в событии начала записи документа

ksGetDocumentOpenVersion – Версия файла, с которой документ был сохранен

Синтаксис:

int LIB_FUNC ksGetDocumentOpenVersion(reference docRef);

Входные параметры:

docRef	- reference документа или 0 для активного документа.

Возвращаемое значение:

	- номер версии из перечисления ksSaveDocumentVersionEnum.

ksSetMixDlgMaterialParam – Установить параметры материала для диалога МЦХ

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMixDlgMaterialParam.

Синтаксис:

int LIB_FUNC ksSetMixDlgMaterialParam(char * material, double density);

Входные параметры

material	- обозначение материала,
density	- плотность.

Возвращаемое значение:

TRUE	- в случае удачи.

ksSetMixDlgMaterialParamW – Установить параметры материала для диалога МЦХ (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetMixDlgMaterialParam.

Синтаксис:

int LIB_FUNC ksSetMixDlgMaterialParamW(LPWSTR material, double density);

Входные параметры

material	- обозначение материала,
density	- плотность.

Возвращаемое значение:

TRUE	- в случае удачи.

ksDistanceCurveCurve – Расстояние между двумя кривыми

Аналог данной функции при использовании Automation - метод ksMathematic2D::ksDistanceCurveCurve.

Синтаксис:

int LIB_FUNC ksDistanceCurveCurve(reference p1, reference p2, double * distanse, double * t1, double * t2);

Входные параметры:

p1	- указатель на первую кривую,
p2	- указатель на вторую кривую.

Выходные параметры:

distanse	- расстояние,
t1	- параметр на кривой 1 в точке с минимальным удалением,
t2	- параметр на кривой 2 в точке с минимальным удалением.

Возвращаемое значение:

1	- успешное завершение,
0	- построить касательную нельзя (кривые совпадают или одна кривая вложена в другую),
-1	- первый объект не существует,
-2	- второй объект не существует,
-3	- кривые расположены в разных видах,
-4	- не совпадают СК определения кривых (геометрическая и аннотационная),
-5	- первый объект не является кривой,
-6	- второй объект не является кривой,
-7	- ошибка.

ksSheetSetupDlg – Открыть диалог для задания формата листа

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_FORMAT_SHEET.htm`

Синтаксис:

int ksSheetSetupDlg (DocumentParam* docPar, void *HWindow);

Входные параметры:

docPar - указатель на структуру параметров документа,
Hwindow - дескриптор окна.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - при выходе из диалога по отмене.

Примечание:

1. Параметры docPar могут быть только параметрами чертежа, для любого другого типа документа функция ничего не сделает. Перед использованием функции необходимо правильно проинициализировать структуру docPar.
2. При использовании Unicode следует использовать функцию ksSheetSetupDlgW.

ksSheetSetupDlgW – Открыть диалог для задания формата листа (Unicode)

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_FORMAT_SHEET.htm`

Синтаксис:

int LIB_FUNC ksSheetSetupDlgW (DocumentParamW* docPar, void *HWindow);

Входные параметры:

docPar - указатель на структуру параметров документа,
Hwindow - дескриптор окна.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - при выходе из диалога по отмене.

Примечание:

1. Параметры docPar могут быть только параметрами чертежа, для любого другого типа документа функция ничего не сделает. Перед использованием функции необходимо правильно проинициализировать структуру docPar.
2. При использовании ANSI следует использовать функцию ksSheetSetupDlg.

ksReDrawDocPart – Перерисовать часть графического документа

Пример...

Синтаксис:

int ksReDrawDocPart (RectParam * rect, reference pView);

Входные параметры:

rect	- указатель на структуру размеров габаритного прямоугольника RectParam,
pView	- указатель на вид.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- при выходе из диалога по отмене.

Примечание:

Координаты габаритного прямоугольника определяются в виде pView. Функция перерисовывает часть вида во всех графических окнах документа, которому принадлежит данный вид.

ksReDrawDocPartEx – Перерисовка части 2D документа (Листа)

Синтаксис:

int LIB_FUNC ksReDrawDocPart (RectParam * rect, reference pView, int ParamType);

Входные параметры:

rect	- указатель на структуру размеров габаритного прямоугольника RectParam,
pView	- указатель на вид,
ParamType	- признак использования системы координат.

Возвращаемое значение:

TRUE	- в случае успешного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Если ParamType == 0, то используются координаты в системе координат текущей СК вида.
2. Если ParamType == VIEV_ALLPARAM, то используются координаты в системе координат вида.
3. Если ParamType == SHEET_ALLPARAM, то используются координаты в системе координат чертежа.

Функции вывода на печать

ksPrintPreviewWindow – Открыть окно предварительного просмотра документа перед печатью

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/1746_199_1_perehod_v_pred_prosmotr.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksPrintPreviewWindow.

Синтаксис:

int ksPrintPreviewWindow (reference docsArr, int inquiry);

Входные параметры:

docsArr	- массив типа CHAR_STR_ARR или CHAR_STR_ARR_W полных имен документов, которые нужно распечатать,
inquiry	- признак запроса документов: 1 - если docsArr = NULL или массив пуст, запросить документы у пользователя, 0 - показать документы без запроса.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

ksPrintKompasDocument – Напечатать КОМПАС-документ

Справка системы КОМПАС...

KOMPAS.chm::/ID_FILE_PRINT.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksPrintKompasDocument.

Синтаксис:

int ksPrintKompasDocument(const char * fileName,
const char * toFile,
double scale);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
scale	- масштаб вывода.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

При использовании Unicode следует использовать функцию ksPrintKompasDocumentW.

ksPrintKompasDocumentW – Напечатать КОМПАС-документ (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/ID_FILE_PRINT.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksPrintKompasDocument.

Синтаксис:

int LIB_FUNC ksPrintKompasDocumentW(LPCWSTR fileName,
LPCWSTR toFile,
double scale);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
scale	- масштаб вывода.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksPrintKompasDocument.

ksPrintKompasDocumentEx – Напечатать КОМПАС-документ

Справка системы КОМПАС...

KOMPAS.chm::/ID_FILE_PRINT.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksPrintKompasDocumentEx.

Синтаксис:

int ksPrintKompasDocumentEx(const char * fileName,
const char * toFile,

double scale,
int fKompasPrinter);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
Scale	- масштаб вывода,
fKompasPrint	- TRUE- используем принтер Компас,
	- FALSE - умолчательный принтер Windows.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

1. При использовании Unicode следует использовать функцию ksPrintKompasDocumentExW.
2. Функция является развитием ksPrintKompasDocument.

ksPrintKompasDocumentExW – Напечатать КОМПАС-документ (Unicode)

Справка системы КОМПАС..

`KOMPAS.chm::/ID_FILE_PRINT.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksPrintKompasDocumentEx.

Синтаксис:

int ksPrintKompasDocumentEx(LPCWSTR fileName,
LPCWSTR toFile,
double scale,
int fKompasPrinter);

Входные параметры:

fileName	- полное имя файла печатаемого документа,
toFile	- имя файла, в который требуется выводить документ (*.prn и т.д.), или NULL, если нужно вывести сразу на принтер,
Scale	- масштаб вывода,
fKompasPrint	- TRUE - используем принтер Компас,
	- FALSE - умолчательный принтер Windows.

Возвращаемое значение:

1	- в случае успешного завершения,

0 - в случае неудачи.

Примечание.

1. При использовании ANSI следует использовать функцию ksPrintKompasDocumentEx.
2. Функция является развитием ksPrintKompasDocumentW.

Функции настройки документа и работы со стилями

Настройки документов

Соответствие входных и выходных параметров...

ksGetSysOptions – Получить системные настройки

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SET_SAVECONFIG.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSysOptions.

Синтаксис:

int ksGetSysOptions (int optionsType, void * param, int sizePar);

Входной параметр:

optionsType - тип настройки,
sizePar - размер структуры параметров.

Выходной параметр:

param - указатель на структуру параметров.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Соответствие входных и выходных параметров...

См. также функции:

GetDocOptions - Получить настройки документа,

SetDocOptions - Заменить настройки документа.

ksGetWorkWindowColor – Получить цвет фона рабочего окна КОМПАС-ГРАФИК

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WINDOWCOLOR_SETUP.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetWorkWindowColor.

Синтаксис:

unsigned long ksGetWorkWindowColor();

Возвращаемое значение:

цвет фона рабочего окна.

ksSetSysOptions – Задать системные настройки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_SAVECONFIG.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksSetSysOptions.

Синтаксис:

int ksSetSysOptions (int optionsType, void * param, int sizePar);

Входной параметр:

optionsType - тип настройки (реализовано для привязок),
sizePar - размер структуры параметров.

Выходной параметр:

param - указатель на структуру параметров.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Стили

AddStyle – Добавить стиль

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/116_9_1_Stili_geometricheskikh_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksAddStyle.

Синтаксис:

unsigned short AddStyle (unsigned char type,

void *param,

unsigned int sizePar,

unsigned char flag);

Входные параметры:

type	- тип стиля,
param	- указатель на структуру параметров стиля,
flag	- признак создания стиля:
	0 - создать вручную,
	1- взять стиль из библиотеки,
sizePar	- размер структуры параметров стиля.

Возвращаемое значение:

идентификатор стиля	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. В настоящее время функция реализована для стилей кривых, текстов и штриховок.
2. При копировании стиля из библиотеки param - указатель на интерфейс ksLibStyle.
3. При создании стиля вручную:

- для обыкновенного стиля кривых type равен CURVE_STYLE, param - указатель на структуру CurveStyleParam.
- для стиля кривых, содержащего фрагменты, type равен CURVE_STYLE_EX, param - указатель на структуру CurveStyleParam.
- для стиля текста type равен TEXT_STYLE, param - указатель на структуру TextStyleParam.

GetStyleParam – Получить параметры стиля из документа

Пример...

Справка системы КОМПАС: стиль текста...

KOMPAS.chm::/546_65_11_Stili_teksta.htm

Стили геометрических объектов...

KOMPAS.chm::/116_9_1_Stili_geometricheskikh_.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetStyleParam.

Синтаксис:

int GetStyleParam (unsigned char type,

unsigned short styleNumber,

void *param,

unsigned int sizePar);

Входные параметры:

type	- тип стиля,
styleNumber	- номер стиля,
sizePar	- размер структуры параметров.

Выходные параметры:

param	- указатель на структуру параметров стиля.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. В случае неудачи структура param заполняется параметрами стиля по умолчанию.
2. Функция получает параметры стиля из документа. Различаются стили кривых, штриховок, текстов, оформления. В настоящее время функция реализована для стилей кривых и текстов.
3. Для обыкновенного стиля кривых type равен CURVE_STYLE, param - указатель на структуру CurveStyleParam.
4. Для стиля кривых, содержащего фрагменты, type равен CURVE_STYLE_EX, param - указатель на структуру CurveStyleParam.
5. Для стиля текста type равен TEXT_STYLE, param - указатель на структуру TextStyleParam.

ksDeleteStyleFromDocument – Удалить стиль из документа

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WORK_WITH_COLLECTION_AND_LIB.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDeleteStyleFromDocument.

Синтаксис:

int ksDeleteStyleFromDocument (unsigned char type,

void *param,

unsigned int size,

unsigned char flag);

Входные параметры:

type - тип стиля,
flag - признак используемой структуры стиля:
0 - создать вручную,
1- взять стиль из библиотеки,
size - размер структуры параметров.

Выходные параметры:

param - указатель на структуру параметров стиля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. В случае, если стиль задается вручную (flag = 0), требуется указать его параметры:

- для стиля кривой type = CURVE_STYLE, param - указатель на структуру CurveStyleParam,
- для расширенного стиля кривой type = CURVE_STYLE_EX, param - указатель на структуру CurveStyleParam,
- для стиля текста type = TEXT_STYLE, param - указатель на структуру TextStyleParam.

2. В случае, когда стиль берется из библиотеки (flag = 1), требуется задать имя библиотеки и номер стиля: для стиля кривых и для стиля текста param - указатель на структуру LibStyle.

ksGetLibraryStylesArray – Получить указатель на динамический массив стилей

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WORK_WITH_COLLECTION_AND_LIB.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetLibraryStylesArray.

Синтаксис:

reference ksGetLibraryStylesArray (char * libraryName,
unsigned char libraryType);

Входные параметры:

libraryName - полное имя библиотеки стилей,
libraryType - тип библиотеки стиля.

Возвращаемое значение:

указатель на динамический массив LIBRARY_STYLE_ARR стилей заданного типа. - в случае успеха,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksGetLibraryStylesArrayW.

ksGetLibraryStylesArrayW – Получить указатель на динамический массив стилей (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WORK_WITH_COLLECTION_AND_LIB.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksGetLibraryStylesArray.

Синтаксис:

reference LIB_FUNC ksGetLibraryStylesArrayW (LPWSTR libraryName,
unsigned char libraryType);

Входные параметры:

libraryName - полное имя библиотеки стилей,
libraryType - тип библиотеки стиля.

Возвращаемое значение:

указатель на динамический массив LIBRARY_STYLE_ARR стилей заданного типа. - в случае успеха,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksGetLibraryStylesArray.

ksGetObjectStyle – Получить стиль для объекта 2D документа

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetObjectStyle.

Синтаксис:

int ksGetObjectStyle(reference obj);

Входные параметры:

obj - указатель reference объекта 2D.

Примечание:

Метод позволяет получить стиль для кривых и эквидистанты.

ksIsStyleInDocument – Проверить, есть ли данный стиль в текущем документе

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_WORK_WITH_COLLECTION_AND_LIB.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsStyleInDocument.

Синтаксис:

int ksIsStyleInDocument (unsigned char type,

void * param,

unsigned int size,

unsigned char flag);

Входные параметры:

type	- тип стиля,
flag	- признак используемой структуры стиля:
	0 - создать вручную,
	1- взять стиль из библиотеки,
size	- размер структуры параметров.

Выходные параметры:

param	- указатель на структуру параметров стиля.

Возвращаемое значение:

1	- стиль в документе есть,
0	- стиля в документе нет.

Примечание:

1. В случае, если стиль задается вручную (flag = 0), требуется указать его параметры:
- для стиля кривой type = CURVE_STYLE, param - указатель на структуру CurveStyleParam,
- для расширенного стиля кривой type = CURVE_STYLE_EX, param - указатель на структуру CurveStyleParam,
- для стиля текста type = TEXT_STYLE, param - указатель на структуру TextStyleParam.
2. В случае, когда стиль берется из библиотеки (flag = 1), требуется задать имя библиотеки и номер стиля: для стиля кривых и для стиля текста param - указатель на структуру LibStyle.

ksSetObjectStyle – Установить стиль для объекта 2D документа

Аналог данной функции при использовании Automation - метод ksDocument2D::ksSetObjectStyle.

Синтаксис:

int ksSetObjectStyle(reference obj, unsigned int style);

Входные параметры:

obj - указатель reference объекта 2D,
style - номер стиля.

Примечание:

Метод позволяет установить стиль для кривых и эквидистанты.

Сервисные функции

Функции данного раздела обеспечивают выполнение системных команд.

EnableTaskAccess – Разрешить/запретить доступ к задаче

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksEnableTaskAccess.

Синтаксис:

void EnableTaskAccess (int enable);

Входной параметр:

enable - признак разрешения доступа к задаче со стороны пользователя:
1 - доступ разрешен,
0 - доступ запрещен.

Примечание:

См. также IsEnableTaskAccess.

IsEnableTaskAccess – Определить, разрешен ли доступ к задаче

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksIsEnableTaskAccess.

Синтаксис:

int IsEnableTaskAccess ();

Возвращаемое значение:

0 - доступ к задаче со стороны пользователя запрещен,
1 - доступ к задаче со стороны пользователя разрешен.

Примечание:

См. также EnableTaskAccess.

ksEnableUndo – Включить/отключить отмену предыдущих операций

Аналог данного метода при использовании Automation - ksDocument2D::ksEnableUndo.

Синтаксис:

void ksEnableUndo (unsigned char enable);

Входные параметры:

enable - признак отмены операций:
0 - отключить,
1 - включить.

Примечания:

1. Если enable=0, количество шагов отмены операций устанавливается равным нулю.
2. Если enable=1, восстанавливается старое значение количества шагов отмены операций, измененное вызовом этого метода с параметром enable=0.
3. Если метод ранее не вызывался, количество шагов отмены операций будет равно умолчательному значению.

ksExecuteKompasCommand – Выполнить команду системы КОМПАС

Аналог данного метода при использовании Automation – KompasObject::ksExecuteKompasCommand.

Синтаксис:

int LIB_FUNC ksExecuteKompasCommand (long commandID, int post);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum,
post - способ запуска команды:
1 - запуск команды через PostMessage,
0 - через SendMessage.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

Примечание:

1. Проверить доступность команды можно с помощью функции ksIsKompasCommandEnable.
2. Проверить, нажата ли в данный момент кнопка команды, можно с помощью ksIsKompasCommandCheck.

ksGetExternalInterface – Получить указатель внешнего интерфейса

Аналог данной функции при использовании Automation - метод KompasObject::ksGetExternalInterface.

Синтаксис:

LPDISPATCH ksGetExternalInterface();

Примечание:

1. Функция не выполняет Addref на выдаваемый интерфейс.
2. Внешний интерфейс можно использовать для обмена данными между двумя приложениями. Первое приложение передает интерфейс при вызове метода KompasObject::ksExecuteKompasLibraryCommandEx.
3. После выполнения KompasObject::ksExecuteKompasLibraryCommandEx, функция ksGetExternalInterface возвращает NULL.

ksGetLookStyle – Получить тип отрисовки визуальной части

Аналог данной функции при использовании Automation - метод KompasObject::lookStyle.

Синтаксис:

long ksGetLookStyle();

Примечание:

Функция возвращает значения из ksTypeLookStyle.

ksGetSystemProfileString – Получить строку из INI-файла системы или из Registry

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSystemProfileString.

Синтаксис:

int ksGetSystemProfileString (char *lpSection,
char *lpKey,
char *lpReturnedString,
int bufLen);

Выходные параметры:

lpSection - имя секции,
lpKey - имя ключа,
lpReturnedString - буфер для выходной строки,
bufLen - длина отведенного буфера в символах.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Пример:

lpSection = "Directories"

lpKey = "Sys"

Результат:

lpReturnedString = "c:\Program Files\Kompas54\Sys"

iSize = 29

Примечание:

1. Если lpReturnedString = 0 или bufLen = 0, то возвращается необходимая длина буфера в символах, иначе - возвращается количество переписанных байт (с учетом завершающего 0).
2. Если возвращаемое значение = 0, значит секция или ключ не были найдены ни в INI-файле, ни в реестре.

ksGetSystemProfileStringW – Получить строку из INI-файла системы или из Registry (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSystemProfileString.

Синтаксис:

int LIB_FUNC ksGetSystemProfileStringW (LPWSTRlpSection,
LPWSTRlpKey,
LPWSTRlpReturnedString,
int iSize);

Выходные параметры:

lpSection - имя секции,
lpKey - имя ключа,
lpReturnedString - буфер для выходной строки,
iSize - размер буфера.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Пример:

lpSection = "Directories"

lpKey = "Sys"

Результат:

lpReturnedString = "c:\Program Files\Kompas54\Sys"

iSize = 29

Примечание:

1. Если lpReturnedString = 0 или iSize = 0, то возвращается необходимый размер буфера, иначе - возвращается количество переписанных байт (с учетом завершающего 0).
2. Если возвращаемое значение = 0, значит секция или ключ не были найдены ни в INI-файле, ни в реестре.

ksGetSystemVersion – Получить версию системы

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSystemVersion.

Синтаксис:

void ksGetSystemVersion (int *iMajor, int *iMinor, int *iRelease, int *iBuild);

Выходные параметры:

iMajor	- старшее слово версии,
iMinor	- младшее слово версии,
iRelease	- номер выпуска внутри одной версии,
iBuild	- номер сборки внутри одного выпуска.

Примечание:

Любой из указателей может быть равен 0, тогда соответствующее ему значение не выдается.

ksGetLibraryStatus – Получить состояние защиты продукта

Синтаксис:

int LIB_FUNC ksGetLibraryStatus(unsigned int prodNumb);

Входные параметры:

prodNumb	- Номер продукта.

Возвращаемое значение

- возвращается значение из перечисления ksProtectProductStatusEnum.

ksIsActiveProcessRunnig – Проверить, запущен ли в текущем графическом документе процесс построения

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsActiveProcessRunnig.

Синтаксис:

int ksIsActiveProcessRunnig ();

Возвращаемое значение:

1	- процесс запущен,
0	- процесс не запущен,
-1	- в случае ошибки.

ksIsHomeVersion – Проверить, является ли версия домашней

Синтаксис:

int LIB_FUNC ksIsHomeVersion()

Возвращаемое значение:

1	- Если запущена домашняя версия Компас kHome.exe.
0	- кнопка отжата.

ksIsExportAvailable – Разрешен ли экспорт в другие форматы

Синтаксис:

int LIB_FUNC ksIsExportAvailable();

Возвращаемое значение

TRUE	- если экспорт в другие форматы разрешен.

Примечание:

Экспорт в другие форматы может быть запрещен приложением Компас-Защита.

ksIsKompasCommandCheck – Проверить нажата ли кнопка команды

Аналог данного метода при использовании Automation – KompasObject::ksIsKompasCommandCheck.

Синтаксис:

int LIB_FUNC ksIsKompasCommandCheck (long commandID);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum.

Возвращаемое значение:

1 - кнопка нажата,
0 - кнопка отжата.

ksIsKompasCommandEnable – Проверить доступность выполнения команды

Аналог данного метода при использовании Automation – KompasObject::ksIsKompasCommandEnable.

Синтаксис:

int LIB_FUNC ksIsKompasCommandEnable (long commandID);

Входные параметры:

commandID - константа из перечисления ProcessTypeEnum или ksKompasCommandEnum,
post - способ запуска команды:
1 - запуск команды через PostMessage,
0 - через SendMessage.

Возвращаемое значение:

1 - команда доступна,
0 - команда недоступна.

Примечание:

1. Проверить доступность команды можно с помощью функции ksIsKompasCommandEnable.
2. Проверить, нажата ли в данный момент кнопка команды, можно с помощью ksIsKompasCommandCheck.

ksIsModule2DActive – Проверить, разрешена ли работа со модулем 2D

Синтаксис:

int ksIsModule2DActive();

Возвращаемое значение:

1	- работа с модулем 2D разрешена,
0	- работа с модулем 2D не разрешена.

ksIsLibraryLocal – Признак локальный/сетевой продукт

Синтаксис:

int LIB_FUNC ksIsLibraryLocal(unsigned int prodNumb);

Входные параметры:

prodNumb	- номер продукта.

Возвращаемое значение

TRUE	- если продукт работает на локальном ключе.

ksIsLibraryProductKeyInfo – Получить информацию о текущей сессии

Синтаксис:

int LIB_FUNC ksIsLibraryProductKeyInfo(unsigned int prodNumb, char * keyInfo, int keyInfoLen);

Входные параметры:

prodNumb	- номер продукта,
keyInfoLen	- размер буфера.

Выходной параметр:

keyInfo	- буфер для информации о текущей сессии.

Возвращаемое значение

TRUE	- в случае успешного завершения.

ksIsLibraryProductKeyInfoW – Получить информацию о текущей сессии (Unicode)

Синтаксис:

int LIB_FUNC ksIsLibraryProductKeyInfoW(unsigned int prodNumb, LPWSTR keyInfo, int keyInfoLen);

Входные параметры:

prodNumb - номер продукта,
keyInfoLen - размер буфера.

Выходной параметр:

keyInfo - буфер для информации о текущей сессии.

Возвращаемое значение

TRUE - в случае успешного завершения.

ksIsLibraryProductName – Получить название продукта

Синтаксис:

int LIB_FUNC ksIsLibraryProductName(unsigned int prodNumb, char * productName, int nameLen);

Входные параметры:

prodNumb - номер продукта,
nameLen - размер буфера.

Выходной параметр:

productName - буфер для имени продукта.

Возвращаемое значение

TRUE - в случае успешного завершения.

ksIsLibraryProductNameW – Получить название продукта (Unicode)

Синтаксис:

int LIB_FUNC ksIsLibraryProductNameW(unsigned int prodNumb, LPWSTR productName, int nameLen);

Входные параметры:

prodNumb - номер продукта,
nameLen - размер буфера.

Выходной параметр:

productName - буфер для имени продукта.

Возвращаемое значение

TRUE - в случае успешного завершения.

ksIsLibraryTrial – Ознакомительный период

Синтаксис:

int LIB_FUNC ksIsLibraryTrial(unsigned int prodNumb);

Входные параметры:

prodNumb - номер продукта.

Возвращаемое значение

TRUE - если для продукта работает ознакомительный период.

ksIsPrintAvailable – Разрешена ли печать

Синтаксис:

int LIB_FUNC ksIsPrintAvailable();

Возвращаемое значение

TRUE - если печать разрешена.

Примечание:

Печать может быть запрещена приложением Компас-Защита.

ksRegisterLibraryNumber – Зарегистрировать номер продукта на сервере лицензий Компас

Синтаксис:

unsigned int LIB_FUNC ksRegisterLibraryNumber(unsigned int prodNumb);

Входные параметры:

prodNumb - номер продукта.

Возвращаемое значение

- идентификатор продукта.

Примечание

Идентификатор продукта используется для разрегистрации функцией ksUnRegisterLibraryNumber.

ksUnRegisterLibraryNumber – Разрегистрировать номер продукта на сервере лицензий Компас

Синтаксис:

unsigned int LIB_FUNC ksUnRegisterLibraryNumber(unsigned int prodNumbUnicueId);

Входные параметры:

prodNumbUnicueId - идентификатор продукта.

Возвращаемое значение

TRUE - в случае успешного завершения.

Примечание

Функция используется для разрегистрации продукта, зарегистрированного функцией ksRegisterLibraryNumber.

ksIsModule3DActive – Проверить, разрешена ли работа со модулем 3D

Аналог данного метода при использовании Automation - KompasObject::ksIsModule3DActive.

Синтаксис:

int ksIsModule3DActive();

Возвращаемое значение:

1 - работа с модулем 3D разрешена,
0 - работа с модулем 3D не разрешена.

ksIsSpdsVersion — проверить, используется ли версия Компас – Строитель

Синтаксис:

int LIB_FUNC ksIsSpdsVersion();

Возвращаемое значение:

1 - Если запущен Компас-строитель kSPDS.exe.

Экспортный номер - 1039.

ksIsStudyVersion – Проверить, является ли версия учебной

Синтаксис:

int LIB_FUNC ksIsStudyVersion();

Возвращаемое значение:

1 - Если запущена учебная версия Компас kStudy.exe.

ksKompasVariant – вернуть версию приложения Компас: Компас – Строитель, Компас – Студент, Иностранная версия и т.д.

Синтаксис:

int LIB_FUNC ksKompasVariant();

Возвращаемое значение:

Возвращается значение, состоящее из битовых флагов, заданных в перечислении ksKompasVariantEnum. В возвращаемом значении может быть взведено несколько флагов одновременно. Например, ksKompasSpds (Компас-строитель) и ksKompasGraphic (КОМПАС-График - Без 3D). Дополнительно может быть взведен флаг ksKompasLatin, если запущена локализованная иностранная версия.

ksModule3D – Подключить 3D модуль для режима сетевой работы системы

Аналог данного метода при использовании Automation - KompasObject::ksModule3D.

Синтаксис:

int ksModule3D (unsigned char attach);

Входные параметры:

attach	1 - включить работу с 3D модулем, 0 - отключить.

Возвращаемое значение:

1	- если был инициирован процесс подключения или отключения 3D модуля,
0	- в случае неудачи.

ksSetCriticalProcess – Установить критический процесс

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksSetCriticalProcess.

Синтаксис:

int ksSetCriticalProcess ();

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Работа процесса, вызванного прикладной библиотекой (например, Cursor или Placement), может быть принудительно прервана переходом к другому действию (команде). Чтобы запретить возможность принудительного прерывания такого процесса, перед его активизацией требуется вызвать данную функцию. Она сделает следующий за ней процесс "критическим", и при попытке его принудительного завершения будет выдано сообщение о необходимости корректного завершения процесса. После штатного завершения процесса признак "критического" процесса снимается, и для следующих процессов (при необходимости) функцию требуется вызывать заново.

ksSetLibraryEnable – Запретить/разрешить продукт (занять лицензию)

Синтаксис:

int LIB_FUNC ksSetLibraryEnable(unsigned int prodNumb,
unsigned char enable);

Входные параметры:

prodNumb	- Номер продукта.
enable	- TRUE - Занять лицензию FALSE - Отпустить лицензию

Возвращаемое значение

TRUE	- если попытка получить лицензию выполнялась.

ksSetLookStyle – Задать тип отрисовки визуальной части

Справка системы КОМПАС...

KOMPAS.chm::/1163_Glava133_Integracija_s_pri.htm

Аналог данной функции при использовании Automation - метод KompasObject::lookStyle.

Синтаксис:

long ksSetLookStyle (long style);

Примечание:

Функция принимает значения из ksTypeLookStyle.

ksSetProgressBar – Установить текущее значение индикатора прогресса

Аналог данного метода при использовании Automation – IProgressBarIndicator::SetProgress.

Синтаксис:

void LIB_FUNC ksSetProgressBar (long currentVal, char * newText, int resetText);

Входные параметры:

currentVal	- текущее значение значение индикатора,
newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния, 0 - не обновлять.

Примечание:

1. Метод выполняется, если индикатор прогресса запущен командой ksStartProgressBar.
2. При использовании Unicode следует использовать функцию ksSetProgressBarW.

ksSetProgressBarW – Установить текущее значение индикатора прогресса (Unicode)

Аналог данного метода при использовании Automation – IProgressBarIndicator::SetProgress.

Синтаксис:

void LIB_FUNC ksSetProgressBarW (long currentVal, LPWSTR newText, int resetText);

Входные параметры:

currentVal	- текущее значение значение индикатора,
newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния,
	0 - не обновлять.

Примечание:

1. Метод выполняется, если индикатор прогресса запущен командой ksStartProgressBarW.
2. При использовании ANSI следует использовать функцию ksSetProgressBar.

ksSetProgressText – Установить текст в строке состояния индикатора прогресса

Аналог данного метода при использовании Automation – IProgressBarIndicator::SetText.

Синтаксис:

void LIB_FUNC ksSetProgressText (char * newText);

Входные параметры:

newText	- текст в строке состояния.

Примечание:

1. Метод выполняется как при запущенном индикаторе прогресса командой ksStartProgressBar, так и при остановленном.
2. Если команда выполняется при запущенном индикаторе прогресса, устанавливаемый текст запоминается как последний текст перед запуском индикатора, и после остановки индикатора командой ksStopProgressBar в строке состояния будет отображаться текст newText.
3. При использовании Unicode следует использовать функцию ksSetProgressTextW.

ksSetProgressTextW – Установить текст в строке состояния индикатора прогресса (Unicode)

Аналог данного метода при использовании Automation – IProgressBarIndicator::SetText.

Синтаксис:

void LIB_FUNC ksSetProgressTextW (LPWSTR newText);

Входные параметры:

newText - текст в строке состояния.

Примечание:

1. Метод выполняется как при запущенном индикаторе прогресса командой ksStartProgressBar, так и при остановленном.
2. Если команда выполняется при запущенном индикаторе прогресса, устанавливаемый текст запоминается как последний текст перед запуском индикатора, и после остановки индикатора командой ksStopProgressBar в строке состояния будет отображаться текст newText.
3. При использовании ANSI следует использовать функцию ksSetProgressText.

ksStartProgressBar – Запустить индикатор прогресса

Аналог данного метода при использовании Automation – IProgressBarIndicator::Start.

Синтаксис:

void LIB_FUNC ksStartProgressBar (long minVal, long maxVal, char * newText, int resetText);

Входные параметры:

minVal	- минимальное значение шкалы,
maxVal	- максимальное значение шкалы,
newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния,
	0 - не обновлять.

Примечание:

1. После завершения работы индикатор процесса должен быть остановлен командой ksStopProgressBar.
2. При использовании Unicode следует использовать функцию ksStartProgressBarW.

ksStartProgressBarW – Запустить индикатор прогресса (Unicode)

Аналог данного метода при использовании Automation – IProgressBarIndicator::Start.

Синтаксис:

void LIB_FUNC ksStartProgressBar (long minVal, long maxVal, LPWSTR newText, int resetText);

Входные параметры:

minVal	- минимальное значение шкалы,
maxVal	- максимальное значение шкалы,
newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния,
	0 - не обновлять.

Примечание:

1. После завершения работы индикатор процесса должен быть остановлен командой ksStopProgressBar.
2. При использовании ANSI следует использовать функцию ksStartProgressBar.

ksStopProgressBar – Остановить индикатор прогресса

Аналог данного метода при использовании Automation – IProgressBarIndicator::Stop.

Синтаксис:

void LIB_FUNC ksStopProgressBar (char * newText, int resetTxt);

Входные параметры:

newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния,
	0 - не обновлять.

Примечание:

Метод выполняется, если индикатор прогресса запущен командой ksStartProgressBar.

ksStopProgressBarW – Остановить индикатор прогресса (Unicode)

Аналог данного метода при использовании Automation – IProgressBarIndicator::Stop.

Синтаксис:

void LIB_FUNC ksStopProgressBar (LPWSTR newText, int resetTxt);

Входные параметры:

newText	- текст в строке состояния,
resetText	1 - обновить текст в строке состояния,
	0 - не обновлять.

Примечание:

Метод выполняется, если индикатор прогресса запущен командой ksStartProgressBar.

ksSystemPath – Получить системный путь установленного типа

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_SET_PATHVIEW.htm`

Аналог данной функции при использовании Automation - метод KompasObject::ksSystemPath.

Синтаксис:

int ksSystemPath (char *buff, int bufLen, int pathType);

Типы системных путей...

Входные параметры:

bufLen - длина отведенного буфера для строки пути в символах,
pathType - тип системной папки.

Выходные параметры:

*buff - строка полного пути (включая завершающий 0).

Возвращаемое значение:

длина строки полного пути.

Примечание:

1. Строка пути не должна завершаться символом '\' (за исключением случая корневой папки устройства - "<drive>:\").
2. При использовании Unicode следует использовать функцию ksSystemPathW.

ksSystemPathW – Получить системный путь установленного типа (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/DLG_SET_PATHVIEW.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksSystemPath.

Синтаксис:

int LIB_FUNC ksSystemPathW (LPWSTR buff, int bufLen, int pathType);

Типы системных путей...

Входные параметры:

bufLen - длина отведенного буфера для строки пути в символах,
pathType - тип системной папки.

Выходные параметры:

*buff - строка полного пути (включая завершающий 0).

Возвращаемое значение:

длина строки полного пути.

Примечание:

1. Строка пути не должна завершаться символом '\' (за исключением случая корневой папки устройства - "<drive>:\").
2. При использовании ANSI следует использовать функцию ksSystemPath.

ksTransferInterface – Преобразовать интерфейсный объект одного типа API в интерфейсный объект API другого типа

Аналог данного метода при использовании Automation - API5 - KompasObject::TransferInterface.

Синтаксис:

LPUNKNOWN LIB_FUNC ksTransferInterface(LPUNKNOWN obj, long newApiType, long objNewType);

Входные параметры:

obj	- интерфейс 3D COM, API5 Auto или API7,
apiNewType	- тип API, к которому преобразуется исходный интерфейс, может принимать значения из ksAPITypeEnum,
objNewType	- тип объекта в интерфейсе, к которому преобразуется исходный объект (может быть задан 0).

Возвращаемое значение:

- указатель на интерфейс объекта в API заданного типа.

Описание:

Метод позволяет получить интерфейс объекта заданного типа objNewType в API заданного типа apiNewType по присланному объекту obj.

Примечание:

1. Исходным объектом может быть интерфейс 3D COM, API5 Auto или API7.
2. Если новый тип API совпадает с API присланного объекта, возвращается присланный объект.
3. Если задан новый тип API как неопределенный (0), возвращается присланный объект.
4. Если задан новый тип API значением вне допустимого множества значений, возвращается NULL.
5. Если задан новый тип объекта значением вне допустимого множества значений, возвращается NULL.
6. Если задан новый тип объекта неопределенным значением (0), возвращается интерфейс базового объекта, совпадающий по типу с требуемым объектом, либо требуемый объект может быть получен от базового через QueryInterface; рекомендуется явно указывать требуемый тип объекта.

ksTransferReference – Преобразовать объект по reference из API5 в интерфейсный объект API7

Аналог данного метода при использовании Automation – KompasObject::TransferReference.

Синтаксис:

LPUNKNOWN ksTransferReference(reference obj, reference doc);

Входные параметры:

obj - указатель на объект в API5,
doc - указатель на документ, где находится объект.

Возвращаемое значение:

- указатель на интерфейс объекта в API7 (Тип данных:LPUNKNOWN).

Описание:

Функция позволяет получить интерфейс объекта в API7 по присланному указателю obj в API5.

Примечание:

1. Функция реализована для документов и объектов вида графического документа. В остальных случаях возвращается NULL.
2. Если obj - указатель на документ, параметр doc игнорируется.
3. Если параметр doc = 0, берется текущий документ.

PumpWaitingMessages – Обработать список сообщений

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksPumpWaitingMessages.

Синтаксис:

void PumpWaitingMessages();

Описание:

Обработать все сообщения, имеющиеся в очереди сообщений задачи.

Функции работы с калькулятором

Функции данного раздела позволяют использовать калькулятор системы КОМПАС.

ksCalculate – Подсчитать значение выражения

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/490_Glava57_Zadanie_zavisimoste.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksCalculate.

Синтаксис:

int ksCalculate (char *s, double * rez);

Входные параметры:

s - строка с выражением.

Выходные параметры:

rez - результат расчета.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Поддерживаются функции и переменные:

   SIN, COS, TAN, ATAN - тригонометрические функции (аргумент в радианах),

   SIND, COSD, TAND, ATAND - тригонометрические функции (аргумент в градусах),

   SQRT, EXP, LN, ABS - корень квадратный, экспонента, натуральный логарифм, абсолютное значение.
2. Наименования функций можно писать в любом регистре (например, как COS, так и cos)
3. Если s = "A1 = 100 ", будет заведена переменная A1 с значением 100.
4. В именах переменных различается регистр букв (например, A1 и a1 - разные переменные).
5. Количество переменных не ограничено.
6. При использовании Unicode следует использовать функцию ksCalculateW.

ksCalculateW – Подсчитать значение выражения (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/490_Glava57_Zadanie_zavisimoste.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksCalculate.

Синтаксис:

int ksCalculateW (LPWSTR s, double * rez);

Входные параметры:

s - строка с выражением,

Выходные параметры:

rez - результат расчета.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Поддерживаются функции и переменные:
   SIN, COS, TAN, ATAN - тригонометрические функции (аргумент в радианах),
   SIND, COSD, TAND, ATAND - тригонометрические функции (аргумент в градусах),
   SQRT, EXP, LN, ABS - корень квадратный, экспонента, натуральный логарифм, абсолютное значение.
2. Наименования функций можно писать в любом регистре (например, как COS, так и cos)
3. Если s = "A1 = 100 ", будет заведена переменная A1 с значением 100.
4. В именах переменных различается регистр букв (например, A1 и a1 - разные переменные).
5. Количество переменных не ограничено.
6. При использовании ANSI следует использовать функцию ksCalculate.

ksCalculateReset – Очистить массив переменных калькулятора

Пример..

Аналог данной функции при использовании Automation - метод KompasObject::ksCalculateReset.

Синтаксис:

int ksCalculateReset();

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Функции экспорта документов

Функции данного раздела обеспечивают сохранение КОМПАС-документов в растровых форматах.

ksSaveAsToRasterFormat – Сохранить документ в растровом формате

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::SaveAsToRasterFormat.

Синтаксис:

int ksSaveAsToRasterFormat (reference sheet,

char * fileName,

RasterFormatParam * par);

Входные параметры:

sheet	- указатель на документ,
fileName	- полное имя файла документа,
par	- структура параметров записи в растровый формат.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами.
   Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.
2. Метод используется для чертежей, фрагментов, спецификаций, деталей и сборок.
3. Для трехмерной модели предварительно нужно получить reference, используя функцию ksGetReferenceFrom3dDocument для COM или свойство ksDocument3D::reference для Automation.
4. Если параметр sheet=0, сохраняется текущий документ.
5. При использовании Unicode следует использовать функцию ksSaveAsToRasterFormatW.

ksSaveAsToRasterFormatW – Сохранить документ в растровом формате (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::SaveAsToRasterFormat.

Синтаксис:

int LIB_FUNC ksSaveAsToRasterFormatW (reference sheet,

LPWSTR fileName,

RasterFormatParamW * par);

Входные параметры:

sheet	- указатель на документ,
fileName	- полное имя файла документа,
par	- структура параметров записи в растровый формат.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Метод позволяет сохранить присланный документ в растровом формате (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) с заданными свойствами. Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.
2. Метод используется для чертежей, фрагментов, спецификаций, деталей и сборок.
3. Для трехмерной модели предварительно нужно получить reference, используя функцию ksGetReferenceFrom3dDocument для COM или свойство ksDocument3D::reference для Automation.
4. Если параметр sheet=0, сохраняется текущий документ.
5. При использовании ANSI следует использовать функцию ksSaveAsToRasterFormat.

ksSaveAsToUncompressedRasterFormat – Сохранить документ в растровом формате без сжатия

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::SaveAsToUncompressedRasterFormat.

Синтаксис:

int ksSaveAsToUncompressedRasterFormat (reference pDoc,
char * fileName,
RasterFormatParam * par);

Входные параметры:

pDoc	- указатель на документ,
fileName	- имя файла документа, должно быть задано полное имя файла,
par	- структура параметров для конвертации в растровый формат.

Возвращаемое значение:

1	- в случае успешного завершения,

0 - в случае неудачи.

Примечание:

1. Функция позволяет сохранить документ в растровом формате без сжатия, с заданными свойствами: цветом, форматом (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) и т.п. Подробнее см. описание RasterFormatParam.
   Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.
2. Функция используется для графических документов (чертеж, фрагмент), документов деталей и сборок, для спецификаций и текстовых документов. Для документов деталей и сборок предварительно нужно получить reference, используя функцию ksGetReferenceFrom3dDocument для COM или свойство ksDocument3D::reference для Automation.
3. При использовании Unicode следует использовать функцию ksSaveAsToUncompressedRasterFormatW.

ksSaveAsToUncompressedRasterFormatW – Сохранить документ в растровом формате без сжатия (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/645_79_3_Sokhranenie_v_rastrovy.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::SaveAsToUncompressedRasterFormat.

Синтаксис:

int LIB_FUNC ksSaveAsToUncompressedRasterFormatW (reference pDoc,
char * fileName,
RasterFormatParamW * par);

Входные параметры:

pDoc	- указатель на документ,
fileName	- имя файла документа, должно быть задано полное имя файла,
par	- структура параметров для конвертации в растровый формат.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция позволяет сохранить документ в растровом формате без сжатия, с заданными свойствами: цветом, форматом (BMP, GIF, JPG, PNG, TIF, TGA, PCX, EMF и WMF) и т.п. Подробнее см. описание RasterFormatParamW.

Сохранение в WMF формат не поддерживается. При сохранении в WMF формате файл записывается в формате EMF.

2. Функция используется для графических документов (чертеж, фрагмент), документов деталей и сборок, для спецификаций и текстовых документов. Для документов деталей и сборок предварительно нужно получить reference, используя функцию ksGetReferenceFrom3dDocument для COM или свойство ksDocument3D::reference для Automation.
3. При использовании ANSI следует использовать функцию ksSaveAsToUncompressedRasterFormat.

Функции работы с прикладной библиотекой

Функции работы с меню

CommandWindow – Запрос к системе на создание окна с деревом команд

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCommandWindow.

Синтаксис:

int CommandWindow(RequestInfo *info);

Входные параметры:

Info - указатель на структуру параметров запроса к системе RequestInfo,

Возвращаемое значение:

- идентификатор выбранной команды, определенный в файле ресурсов, - в случае успешного завершения,
- порядковый номер в строке команд.
-1 - если команда не выбрана.

Описание.

Функция создает окно с деревом команд, определяемых строкой команд в структуре info или идентификатором меню из файла ресурсов. Команды в строке разделены пробелом или восклицательным знаком. Функции обратной связи управление передается после выбора команды или отказа.

В качестве функции обратной связи передается указатель на функцию типа CommandWindowCallBack. Если в качестве функции обратной связи задан NULL, то управление из CommandWindow возвращается немедленно, как только пользователь выберет команду в дереве команд. При этом возвращается идентификатор выбранной команды. В противном случае управление вернется, если пользователь закроет окно или функция обратной связи вернет FALSE. При этом возвращается -1.

Если в качестве commands задана строка в формате Компас 4.X, например "!Команда_1 !Команда_2 ...", то идентификатором команды является ее позиция в строке (начиная с 1), то есть идентификатором команды "Команда_1" является 1. Если в качестве commands задан идентификатор меню, то идентификатором команды является идентификатор соответствующего пункта меню.

Функция возвращает идентификатор выбранной команды, определенный в файле ресурсов, или порядковый номер в строке команд.

Примечание.

При использовании Unicode следует использовать функцию CommandWindowW.

CommandWindowW – Запрос к системе на создание окна с деревом команд (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCommandWindow.

Синтаксис:

int LIB_FUNC CommandWindow(RequestInfoW *info);

Входные параметры:

Info -указатель на структуру параметров запроса к системе RequestInfo,

Возвращаемое значение:

- идентификатор выбранной команды, определенный в файле ресурсов, - порядковый номер в строке команд.	- в случае успешного завершения,
-1	- если команда не выбрана.

Описание.

Функция создает окно с деревом команд, определяемых строкой команд в структуре info или идентификатором меню из файла ресурсов. Команды в строке разделены пробелом или восклицательным знаком. Функции обратной связи управление передается после выбора команды или отказа.

В качестве функции обратной связи передается указатель на функцию типа CommandWindowCallBack. Если в качестве функции обратной связи задан NULL, то управление из CommandWindow возвращается немедленно, как только пользователь выберет команду в дереве команд. При этом возвращается идентификатор выбранной команды. В противном случае управление вернется, если пользователь закроет окно, или функция обратной связи вернет FALSE. При этом возвращается -1.

Если в качестве commands задана строка в формате Компас 4.X, например "!Команда_1 !Команда_2 ...", то идентификатором команды является ее позиция в строке (начиная с 1), то есть идентификатором команды "Команда_1" является 1. Если в качестве commands задан идентификатор меню, то идентификатором команды является идентификатор соответствующего пункта меню.

Функция возвращает идентификатор выбранной команды, определенный в файле ресурсов, или порядковый номер в строке команд.

Примечание.

При использовании ANSI следует использовать функцию CommandWindow.

Функции ввода параметров

Функции данного раздела обеспечивает ввод параметров (целых, действительных, строковых), указание точек и вариантов действия. Функции указания точек в КОМПАС-ГРАФИК реализованы таким образом, что вместо ожидаемой точки пользователь может задать другой вариант построения объекта. Список вариантов указывается в строке приглашения.

Cursor – Указать положение объекта или определить вариант действия

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksCursor.

Синтаксис:

int Cursor (RequestInfo *info, double *x, double *y, void * phantom);

Входные параметры:

Info	- указатель на структуру параметров запроса к системе,
x,y	- координаты введенной точки,
Phantom	- указатель на структуру управления фантомом, определяющую тип движения курсора.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие функции прекращается после первого шага.

GetValidator – Получить валидатор

Пример...

Синтаксис:

void* GetValidator(void *min, void *max, unsigned char type);

Входные параметры:

min, max	- границы интервала,
type	- тип данных.

Возвращаемое значение:

указатель на Tvalidator	- в случае удачного завершения,
0	- в случае неудачи или если курсор находится не на поле чертежа.

Примечание:

Функция работает только в OWL.

ksExecDialPredefinedText – Получить предопределенный текст из файла текстовых шаблонов

Справка системы КОМПАС...

KOMPAS.chm::/538_65_9_6_Tekstovye_shablony.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksExecDialPredefinedText.

Синтаксис:

int ksExecDialPredefinedText (void *HWindow, char * str, int sizeStr);

Входные параметры:

HWindow	- несущее окно,
sizeStr	- размер строки str.

Выходной параметр:

str	- строка-буфер, в которую будет помещен текст.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

1. Все строки складываются в одну строку str с переводом строки "\n".
2. При использовании Unicode следует использовать функцию ksExecDialPredefinedTextW.

ksExecDialPredefinedTextW – Получить предопределенный текст из файла текстовых шаблонов (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/538_65_9_6_Tekstovye_shablony.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksExecDialPredefinedText.

Синтаксис:

int LIB_FUNC ksExecDialPredefinedTextW (void *HWindow, LPWSTR str, int sizeStr);

Входные параметры:

HWindow	- несущее окно,
sizeStr	- размер строки str.

Выходной параметр:

str	- строка-буфер, в которую будет помещен текст.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

1. Все строки складываются в одну строку str с переводом строки "\n".
2. При использовании ANSI следует использовать функцию ksExecDialPredefinedText.

ksExecDialPredefinedTextEx – Получить предопределенный текст из файла текстовых шаблонов

Справка системы КОМПАС...

KOMPAS.chm::/538_65_9_6_Tekstovye_shablony.htm

Аналог данной функции при использовании Automation - метод KompasObject::ksExecDialPredefinedText.

Синтаксис:

reference ksExecDialPredefinedTextEx (void *HWindow);

Входные параметры:

HWindow	- дескриптор окна.

Возвращаемое значение:

указатель на интерфейс ksDynamicArray массива строк, тип массива - TEXT_LINE_ARR. - в случае успеха,
NULL - в случае неудачи.

ksExecDialSymbol –Вызов диалога "Вставка символа"

Синтаксис:

int LIB_FUNC ksExecDialSymbol(void *HWindow, int * symb, LPSTR font, unsigned int len);

Входные параметры:

HWindow - дескриптор окна,
symb - номер символа на котором будет стоять курсор,
font - имя шрифта в диалоге,
len - размер буфера для имени шрифта.

Выходные параметры:

symb - номер выбранного символа,
font - имя выбранного шрифта.

Возвращаемое значение:

1 - в случае удачного завершения,
-1 - в случае неудачи или закрытии диалога по отмене.

ksExecDialSymbolW – Вызов диалога "Вставка символа" (Unicode)

Синтаксис:

int LIB_FUNC ksExecDialSymbol(void *HWindow, int * symb, LPSTR font, unsigned int len);

Входные параметры:

HWindow - дескриптор окна,
symb - номер символа на котором будет стоять курсор,
font - имя шрифта в диалоге,
len - размер буфера для имени шрифта.

Выходные параметры:

symb - номер выбранного символа,
font - имя выбранного шрифта.

Возвращаемое значение:

1 - в случае удачного завершения,
-1 - в случае неудачи или закрытия диалога по отмене.

См. также функцию ksGetSnapInfo - Получить текущую информацию о привязках.

ksExecDialSpecialSymbol – Вызов диалога "Вставка спецзнака"

Синтаксис:

int LIB_FUNC ksExecDialSpecialSymbol(void * HWindow);

Входные параметры:

HWindow - дескриптор окна.

Возвращаемое значение:

Номер спецзнака - в случае удачного завершения,
-1 - в случае неудачи или закрытии диалога по отмене.

ksGetCursorPosition – Получить координаты курсора

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetCursorPosition

Синтаксис:

int ksGetCursorPosition (double *x,

double *y,

int type);

Входные параметры:

x,y - координаты курсора в миллиметрах,
type - признак способа определения координат:
0 - без учета привязок,
1 - с учетом привязок.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи или если курсор находится не на поле чертежа.

ksGetCursorLimit – Получить радиус окружности, вписанной в "ловушку" курсора

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/SET_SNAP_CURSOR_DIALOG.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksGetCursorLimit.

Синтаксис:

double ksGetCursorLimit();

Возвращаемое значение:

- радиус окружности, вписанной в "ловушку" курсора.

ksIsCursorOrPlacementDocument – Проверить, запущен ли в текущем графическом документе процесс Cursor или Placement

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksIsCursorOrPlacementDocument.

Синтаксис:

int ksIsCursorOrPlacementDocument ();

Возвращаемое значение:

1	- процесс запущен,
0	- процесс не запущен.

ksMaterialDlg – Получить материал и его плотность из справочника материалов

Аналог данной функции при использовании Automation - метод KompasObject::ksMaterialDlg.

Синтаксис:

int ksMaterialDlg(void *HWindow,

char * material,

int sizeStr,

double* plt,

double* kod_size,

char *kod_tip);

Входные параметры:

Hwindow - дескриптор окна,
sizeStr - размер строки.

Выходные параметры:

material - строка-буфер, в которую будет помещено обозначение выбранного материала,
plt - плотность материала (г/куб.мм),
kod_size - четыре элемента:
kod_size[0] - код вида типоразмера:
1 - толщина,
2 - диаметр,
0 - вид не определен,
kod_size[1] - значение размера вида толщина, диаметр, диаметр вписанной окружности, значение A типоразмеров вида AxB или AxBxC,
kod_size[2] - значение B типоразмеров вида AxB или AxBxC,
kod_size[3] - значение C типоразмеров вида AxBxC,
kod_tip - строка кодов типов сортаментов для отображения, через запятую; если 0 - отображается все.

Возвращаемое значение:

-1 - справочник материалов не подключился,
0 - при выходе из диалога справочника материалов по отмене,
1 - в случае удачного завершения.

Примечание:

Функция устарела и не используется. Предназначалась для работы со справочником материалов, который в настоящее время не входит в комплект поставки.

ksPhantomShowHide – Включить или выключить отображение фантома на экране

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPhantomShowHide.

Синтаксис:

int ksPhantomShowHide (unsigned char type);

Входные параметры:

type - признак отрисовки фантома:
1 - включить,
0 - выключить.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Placement – Задать точку и угол

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksPlacement.

Синтаксис:

int Placement (RequestInfo *info, double *x, double *y, double *angle, void *phantom);

Входные параметры:

info - структуру параметров запроса к системе,
x,y - координаты введенной точки,
phantom - указатель на структуру управления фантомом, определяющую тип движения курсора,
angle - введенный угол.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Возможные варианты (команды) задаются в строке commands структуры info и разделяются восклицательными знаками или пробелами. Если вместо строки в качестве параметра передать идентификатор меню из файла ресурсов, то соответствующее меню будет выдано в окне приглашений. Если в качестве адреса _callBack передается NULL, то действие функции прекращается после первого шага.

ReadDouble – Ввести вещественное число с контролем попадания значения в заданный интервал

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksReadDouble.

Синтаксис:

int ReadDouble (char *smess, double def, double min, double max, double *value);

Входные параметры:

smess - строка приглашения,
def - значение, предлагаемое по умолчанию,
min,max - интервал возможных значений,

value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ReadDoubleW.

ReadDoubleW – Ввести вещественное число с контролем попадания значения в заданный интервал (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksReadDouble.

Синтаксис:

int LIB_FUNC ReadDoubleW (LPWSTR smess, double def, double min, double max, double *value);

Входные параметры:

smess - строка приглашения,
def - значение, предлагаемое по умолчанию,
min,max - интервал возможных значений,
value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ReadDouble.

ReadInt – Ввести целое число с контролем попадания значения в заданный интервал

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksReadInt.

Синтаксис:

int ReadInt (char *smess, int def, int min, int max, int *value);

Входные параметры:

smess - строка приглашения,
def - значение, предлагаемое по умолчанию,
min,max - интервал возможных значений,

value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ReadIntW.

ReadIntW – Ввести целое число с контролем попадания значения в заданный интервал (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksReadInt.

Синтаксис:

int LIB_FUNC ReadIntW (LPWSTR smess, int def, int min, int max, int *value);

Входные параметры:

smess - строка приглашения,
def - значение, предлагаемое по умолчанию,
min,max - интервал возможных значений,
value - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ReadInt.

ReadLong – Ввести длинное целое число с контролем попадания значения в заданный интервал

Пример...

Синтаксис:

int ReadLong (char *smess, long def, long min, long max, long *value);

Входные параметры:

smess - строка приглашения,
def - предлагаемое значение по умолчанию,
min,max - интервал возможных значений,
R - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ReadLongW.

ReadLongW – Ввести длинное целое число с контролем попадания значения в заданный интервал (Unicode)

Синтаксис:

int LIB_FUNC ReadLongW (LPWSTR smess, long def, long min, long max, long *value);

Входные параметры:

smess - строка приглашения,
def - предлагаемое значение по умолчанию,
min,max - интервал возможных значений,
R - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ReadLong.

ReadString – Ввести строку заданной длины

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksReadString.

Синтаксис:

int ReadString(char *smess, char *str, int maxlen);

Входные параметры:

smess - строка приглашения,
maxlen - максимально допустимая длина строки,
str - результат ввода.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ReadStringW.

ReadStringW – Ввести строку заданной длины (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksReadString.

Синтаксис:

int LIB_FUNC ReadString(LPWSTR smess, char *str, int maxlen);

Входные параметры:

smess	- строка приглашения,
maxlen	- максимально допустимая длина строки,
str	- результат ввода.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ReadString.

Функции вывода на экран

Функции данного раздела обеспечивают вывод на экран сообщений, подсказок, слайдов и обработку результата работы библиотечной функции.

CommandWindowCallBack – Прототип функции обратной связи для запроса окна с деревом команд

Синтаксис:

typedef int (WINAPI *CommandWindowCallBack)(int com,
RequestInfo * info);

Входные параметры:

com	- идентификатор команды из окна команд,
info	- указатель на область памяти для замены состава команд.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании Unicode следует использовать функцию CommandWindowCallBackW.

CommandWindowCallBackW – Прототип функции обратной связи для запроса окна с деревом команд (Unicode)

Синтаксис:

typedef int (WINAPI *CommandWindowCallBackW)(int com,
RequestInfoW * info,);

Входные параметры:

com	- идентификатор команды из окна команд,
info	- указатель на область памяти для замены состава команд.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании ANSI следует использовать функцию CommandWindowCallBack.

CursorCallBack – Прототип функции обратной связи для запроса точки

Синтаксис:

typedef int (WINAPI *CursorCallBack)(int com,
double * x,
double * y,
RequestInfo * info,
void *phantom,
int dynamic);

Входные параметры:

com	- идентификатор команды из окна команд,
x, y	- координаты точки привязки,
info	- указатель на область памяти для замены состава команд,
phantom	- указатель на фантомную группу,
dynamic	- признак динамического вызова.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании Unicode следует использовать функцию CursorCallBackW.

CursorCallBackW – Прототип функции обратной связи для запроса точки (Unicode)

Синтаксис:

```
typedef int ( WINAPI *CursorCallBackW )( int com,
double * x,
double * y,
RequestInfoW * info,
void *phantom,
int dynamic );
```

Входные параметры:

com	- идентификатор команды из окна команд,
x, y	- координаты точки привязки,
info	- указатель на область памяти для замены состава команд,
phantom	- указатель на фантомную группу,
dynamic	- признак динамического вызова.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании ANSI следует использовать функцию CursorCallBack.

DrawBitmap – Отрисовать растровый слайд

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawBitmap.

Синтаксис:

```
int DrawBitmap ( void *Hwindow, unsigned int BitmapID );
```

Входной параметр:

HWindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,
BitmapID	- идентификатор BITMAP в файле ресурсов.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

DrawSlide – Отрисовать векторный слайд

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawSlide.

Синтаксис:

int DrawSlide (void *Hwindow, unsigned int SlideID);

Входной параметр:

HWindow - дескриптор окна, в котором нужно отрисовать слайд,
SlideID - идентификатор слайда в файле ресурсов.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Error – Выдать сообщение об ошибке

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksError.

Синтаксис:

void Error (char * s);

Входной параметр:

s - строка с текстом сообщения.

Примечание::

1. После вывода сообщения будет ожидаться нажатие любой клавиши.
2. При использовании Unicode следует использовать функцию ErrorW.

ErrorW – Выдать сообщение об ошибке (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksError.

Синтаксис:

void LIB_FUNC ErrorW(LPWSTR s);

Входной параметр:

s - строка с текстом сообщения.

Примечание::

1. После вывода сообщения будет ожидаться нажатие любой клавиши.
2. При использовании ANSI следует использовать функцию Error.

FilePreviewFuncCallBack – Прототип функции обратной связи для функций выбора файлов

Пример...

Синтаксис:

typedef int (WINAPI *FilePreviewFuncCallBack)(HWND HWindow, char * fileName);

Входные параметры:

HWindow	- дескриптор окна просмотра,
fileName	- файл, который нужно показать в окне просмотра.

Возвращаемое значение:

1	- если файл отрисован,
0	- если файл не отрисован.

См. также:

ksInitFilePreviewFunc

Примечание.

При использовании Unicode следует использовать функцию FilePreviewFuncCallBackW.

FilePreviewFuncCallBackW – Прототип функции обратной связи для функций выбора файлов (Unicode)

Синтаксис:

typedef int (WINAPI *FilePreviewFuncCallBackW)(HWND HWindow, LPWSTR fileName);

Входные параметры:

HWindow	- дескриптор окна просмотра,
fileName	- файл, который нужно показать в окне просмотра.

Возвращаемое значение:

1	- если файл отрисован,
0	- если файл не отрисован.

См. также:

ksInitFilePreviewFuncW

Примечание.

При использовании ANSI следует использовать функцию FilePreviewFuncCallBack.

GetParentHWindow – Вернуть дескриптор скрытого окна

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawKompasText.

Синтаксис:

void * LIB_FUNC GetParentHWindow();

Примечание.

Можно передавать как Parent дескриптор главного окна.

GetHWindow – Получить дескриптор главного окна КОМПАС-ГРАФИК

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetHWindow.

Синтаксис:

void * far __export pascal GetHWindow (void);

Возвращаемое значение:

дескриптор главного окна, который используется функциями WINDOWS для работы с окнами.

ksDrawBitmapEx – Отрисовать растровый слайд в заданном окне

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawBitmapEx.

Синтаксис:

int ksDrawBitmapEx (void* HWindow,
unsigned int sldID,
HINSTANCE hInst)

Входной параметр:

HWindow	- дескриптор окна, в котором нужно отрисовать BITMAP-слайд,
sldID	- идентификатор BITMAP-слайда в файле ресурсов,
hInst	- hInstance текущей библиотеки или NULL, если она одна.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksDrawKompasDocument – Показать КОМПАС-документ в виде слайда в окне

Пример...

Синтаксис:

int ksDrawKompasDocument (void * HWindow, char * docFileName);

Входные параметры:

HWindow - дескриптор окна просмотра,
docFileName - полное имя файла документа.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Действие функции распространяется также на фрагменты и модели из библиотек. При этом имя файла должно иметь вид "c:\gr\lib1.l3d|детали|литье|фланец",
   где:
   - c:\gr\lib1.l3d - имя файла библиотеки,
   - |детали|литье| - разделы, подразделы внутри библиотеки,
   - фланец - имя фрагмента или модели.
2. При использовании Unicode следует использовать функцию ksDrawKompasDocumentW.

ksDrawKompasDocumentW – Показать КОМПАС-документ в виде слайда в окне (Unicode)

Синтаксис:

int LIB_FUNC ksDrawKompasDocumentW (void * HWindow, LPWSTR docFileName);

Входные параметры:

HWindow - дескриптор окна просмотра,
docFileName - полное имя файла документа.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Действие функции распространяется также на фрагменты и модели из библиотек. При этом имя файла должно иметь вид "c:\gr\lib1.l3d|детали|литье|фланец", где:

- c:\gr\lib1.l3d - имя файла библиотеки,
- |детали|литье| - разделы, подразделы внутри библиотеки,
- фланец - имя фрагмента или модели.

2. При использовании ANSI следует использовать функцию ksDrawKompasDocument.

ksDrawKompasGroup – Отрисовать группу в виде слайда в окне

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksDrawKompasGroup.

Синтаксис:

int ksDrawKompasGroup (void * HWindow, reference gr);

Входные параметры:

HWindow	- дескриптор окна,
gr	- указатель на группу.

Описание:

Функция выполняет отрисовку указанной группы в виде слайда в присланном окне.

ksDrawKompasText – Отрисовать текст в формате КОМПАС в окне

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawKompasText.

Синтаксис:

int LIB_FUNC ksDrawKompasText(void *HWindow, LPSTR text);

Входной параметр:

HWindow	- дескриптор окна для отрисовки слайда,
text	- текст.

Примечание.

При использовании Unicode следует использовать функцию ksDrawKompasTextW.

ksDrawKompasTextW – Отрисовать текст в формате КОМПАС в окне (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawKompasText.

Синтаксис:

int LIB_FUNC ksDrawKompasTextW(void *HWindow, LPWSTR text);

Входной параметр:

HWindow - дескриптор окна для отрисовки слайда,
text - текст.

Примечание.

При использовании ANSI следует использовать функцию ksDrawKompasText.

ksDrawSlideEx – Отрисовать слайд (расширенная функция)

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawSlideEx.

Синтаксис:

int ksDrawSlideEx (void *HWindow,

unsigned int SlideID,

HINSTANCE hInstance);

Входные параметры:

HWindow - дескриптор окна,
SlideID - номер слайда в файле ресурсов приложения,
hInstance - NULL - отрисовка в текущей библиотеке,
hInstance - в подключаемой.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

ksDrawSlideFromFile – Отрисовать слайд в окне из текстового файла, содержащего блок RCDATA

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawSlideFromFile.

Синтаксис:

int ksDrawSlideFromFile (void *Hwindow, char * fileName);

Входной параметр:

Hwindow - дескриптор окна,
fileName - полное имя файла.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Функция предназначена для отладки слайдов. Позволяет просмотреть отредактированный слайд в окне диалога без перетрансляции тестовой библиотеки.
2. При использовании Unicode следует использовать функцию ksDrawSlideFromFileW.

ksDrawSlideFromFileW – Отрисовать слайд в окне из текстового файла, содержащего блок RCDATA (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksDrawSlideFromFile.

Синтаксис:

int LIB_FUNC ksDrawSlideFromFileW (void *Hwindow, LPWSTR fileName);

Входной параметр:

Hwindow - дескриптор окна,
fileName - полное имя файла.

Возвращаемое значение:

1 - в случае удачного завершения.
0 - в случае неудачи.

Примечание:

1. Функция предназначена для отладки слайдов. Позволяет просмотреть отредактированный слайд в окне диалога без перетрансляции тестовой библиотеки.
2. При использовании ANSI следует использовать функцию ksDrawSlideFromFile.

ksInitFilePreviewFunc – Инициализировать адрес пользовательской функции просмотра пользовательского файла

Пример...

Аналог данной функции при использовании Automation - метод ksDocument2D::ksInitFilePreviewFunc.

Синтаксис:

int ksInitFilePreviewFunc (FilePreviewFuncCallBack func);

Входные параметры:

func - пользовательская функция просмотра пользовательского файла.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

См. также:

FilePreviewFuncCallBack

Примечание:

1. Если func вернет 1 - функция отрисовала файл, если 0 - функция не отрисовала файл, файл будет отрисовывать КОМПАС.
2. При использовании Unicode следует использовать функцию ksInitFilePreviewFuncW.

ksInitFilePreviewFuncW – Инициализировать адрес пользовательской функции просмотра пользовательского файла (Unicode)

Аналог данной функции при использовании Automation - метод ksDocument2D::ksInitFilePreviewFuncW.

Синтаксис:

int ksInitFilePreviewFuncW (FilePreviewFuncCallBackW func);

Входные параметры:

func - пользовательская функция просмотра пользовательского файла.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

См. также:

FilePreviewFuncCallBackW

Примечание:

1. Если func вернет 1 - функция отрисовала файл, если 0 - функция не отрисовала файл, файл будет отрисовывать КОМПАС.
2. При использовании ANSI следует использовать функцию ksInitFilePreviewFunc.

ksSetDebugMessagesMode – Включить/выключить режим автоматического вывода сообщений о результатах работы библиотеки

Аналог данной функции при использовании Automation - метод KompasObject::ksSetDebugMessagesMode

Синтаксис:

int LIB_FUNC ksSetDebugMessagesMode(int debugMode);

Входной параметр:

debugMode - флаг состояния режима (1 - включить, 0 - выключить).

Возвращаемое значение:

- предыдущее состояние флага режима.

Примечание

1. При включенном режиме сообщения выдаются сразу после возникновении ошибки.
2. Данной функцией рекомендуется пользоваться только в режиме отладки библиотеки.
3. Дополнительно о работе с ошибками библиотеки см. ReturnResult (для Automation - KompasObject::ksReturnResult).
4. После показа сообщения, если ошибка не является фатальной, флаг ошибки сбрасывается.

ksSlideBackground – Установить цвет фона по умолчанию для отрисовки слайда

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksSlideBackground.

Синтаксис:

void ksSlideBackground (COLORREF color);

Входной параметр:

color - цвет фона слайда.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Message – Выдать сообщение

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksMessage.

Синтаксис:

void Message (char * s);

Входной параметр:

s - строка с текстом сообщения.

Примечание.

При использовании Unicode следует использовать функцию MessageW.

MessageW – Выдать сообщение (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksMessage.

Синтаксис:

void LIB_FUNK MessageW (LPWSTR s);

Входной параметр:

s - строка с текстом сообщения.

Примечание.

При использовании ANSI следует использовать функцию Message.

MessageBoxResult – Вывести сообщение, соответствующее результату работы библиотеки (с кодом ошибки)

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksMessageBoxResult.

Синтаксис:

void MessageBoxResult (void);

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Функция сбрасывает флаг ошибки в случае, если она не является фатальной.

Pause – Получить сообщение с ожиданием нажатия клавиши

Пример...

Синтаксис:

void Pause (char * s);

Входной параметр:

s	- строка с текстом сообщения.

Примечание:

После вывода сообщения будет ожидаться нажатие любой клавиши.

PlacementCallBack – Прототип функции обратной связи для запроса точки и угла

Синтаксис:

typedef int (WINAPI *PlacementCallBack)(int com,
double * x,
double * y,
double * angle,
RequestInfo * info,
void *phantom,
int dynamic);

Входные параметры:

com	- идентификатор команды из окна команд,
x, y	- координаты точки привязки,
angle	- угол наклона,
info	- указатель на область памяти для замены состава команд,
phantom	- указатель на фантомную группу,
dynamic	- признак динамического вызова.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании Unicode следует использовать функцию PlacementCallBackW.

PlacementCallBackW – Прототип функции обратной связи для запроса точки и угла (Unicode)

Синтаксис:

```
typedef int ( WINAPI *PlacementCallBackW )( int com,
double * x,
double * y,
double * angle,
RequestInfoW * info,
void *phantom,
int dynamic );
```

Входные параметры:

com	- идентификатор команды из окна команд,
x, y	- координаты точки привязки,
angle	- угол наклона,
info	- указатель на область памяти для замены состава команд,
phantom	- указатель на фантомную группу,
dynamic	- признак динамического вызова.

Возвращаемое значение:

1	- если нужно продолжить запрос,
0	- если нужно прекратить запрос.

Примечание.

При использовании ANSI следует использовать функцию PlacementCallBack.

ReturnResult – Получить номер ошибки

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksReturnResult.

Синтаксис:

```
int ReturnResult ( void );
```

Возвращаемое значение:

	- код ошибки в зависимости от типа документа: графического или документа-модели при выполнении библиотечной программы
0	- при успешном завершении программы.

Примечание:

1. Коды ошибок документа-модели и графического документа частично совпадают, поэтому их нужно обрабатывать в зависимости от выполняемых операций.
2. Ошибка с номером>0 не является фатальной. Отрицательный номер ошибки приводит к завершению программы.
3. Текст ошибок можно получить, используя функцию StrResult.
4. Сообщение с текстом ошибки можно получить, используя функцию MessageBoxResult.

5. Сбросить ошибку, если она не фатальная, можно, используя функцию ResultNULL.

StrResult – Вывести строку, соответствующую результату работы библиотеки

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksStrResult.

Синтаксис:

void StrResult (char *str, int limit);

Входной параметр:

limit - ограничение по длине строки.

Возвращаемое значение:

str - строка сообщения, соответствующая результату работы библиотеки.

Примечание:

1. Функция сбрасывает флаг ошибки в случае, если она не является фатальной.
2. При использовании Unicode следует использовать функцию StrResultW.

StrResultW – Вывести строку, соответствующую результату работы библиотеки (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksStrResult.

Синтаксис:

void LIB_FUNC StrResultW (LPWSTR str, int limit);

Входной параметр:

limit - ограничение по длине строки.

Возвращаемое значение:

str - строка сообщения, соответствующая результату работы библиотеки.

Примечание:

1. Функция сбрасывает флаг ошибки в случае, если она не является фатальной.
2. При использовании ANSI следует использовать функцию StrResult.

WriteSlide – Записать выделенные объекты чертежа в формате векторного слайда КОМПАС

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksWriteSlide.

Синтаксис:

int WriteSlide (char *filename, unsigned int ID, double x, double y);

Входной параметр:

filename	- имя файла для записи,
ID	- идентификатор слайда в файле ресурсов,
x,y	- базовая точка.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию WriteSlideW.

WriteSlideW – Записать выделенные объекты чертежа в формате векторного слайда КОМПАС (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksWriteSlide.

Синтаксис:

int LIB_FUNC WriteSlideW (LPWSTR filename, unsigned int ID, double x, double y);

Входной параметр:

filename	- имя файла для записи,
ID	- идентификатор слайда в файле ресурсов,
x,y	- базовая точка.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию WriteSlide.

YesNo – Подтвердить действие или отказаться от него

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksYesNo.

Синтаксис:

int YesNo (char * s);

Входной параметр:

s - строка сообщения.

Возвращаемое значение:

1 - при подтверждении действия,
0 - при отказе,
-1 - при отмене действия.

Примечание:

1. Подтверждение означает нажатие в диалоге кнопки Да (Yes, OK); отказ - нажатие кнопки Нет (No); отмена действия - нажатие кнопки Отмена (Cansel).
2. При использовании Unicode следует использовать функцию YesNoW.

YesNoW – Подтвердить действие или отказаться от него (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksYesNo.

Синтаксис:

int LIB_FUNC YesNoW(LPWSTR s);

Входной параметр:

s - строка сообщения.

Возвращаемое значение:

1 - при подтверждении действия,
0 - при отказе,
-1 - при отмене действия.

Примечание:

1. Подтверждение означает нажатие в диалоге кнопки Да (Yes, OK); отказ - нажатие кнопки Нет (No); отмена действия - нажатие кнопки Отмена (Cansel).
2. При использовании ANSI следует использовать функцию YesNo.

Сервисные библиотечные функции

EditMacroMode – Получить режим работы функции библиотеки (создание нового или редактирование существующего макроэлемента)

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/Z_MACRO.htm

Аналог данной функции при использовании Automation - метод ksDocument2D::ksEditMacroMode.

Синтаксис:

reference EditMacroMode();

Возвращаемое значение:

указатель на редактируемый макроэлемент	- при редактировании макроэлемента,
0	- при создании макроэлемента.

ksConvertLangMenu – Конвертировать меню в соответствии с текущим словарем

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangMenu.

Синтаксис:

HMENU ksConvertLangMenu (HMENU hMenu);

Входные параметры:

hMenu	- дескриптор исходного меню.

Возвращаемое значение:

- дескриптор переведённого меню.

ksConvertLangStr – Конвертировать строку src в dst в соответствии с текущим словарем

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangStr.

Синтаксис:

int ksConvertLangStr (char* src, char* dst, int dstMaxLen);

Входные параметры:

src	- строка для конвертирования,
dstMaxLen	- максимальная длина переведённой строки.

Выходные параметры:

dst	- переведенная строка.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

1. Функция устарела, предпочтительнее использовать ksConvertLangStrEx
2. Словарь сохраняется в файле с расширением dic. Строки файла должны иметь следующий формат:

   "Вращение"="Rotation"

   "Параметры"="Parameters"

   ...

   ...

   "Перемещение"="Moving"

   Строки файла должны быть отсортированы по алфавиту.
3. При использовании Unicode следует использовать функцию ksConvertLangStrW.

ksConvertLangStrW – Конвертировать строку src в dst в соответствии с текущим словарем (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangStr.

Синтаксис:

int LIB_FUNC ksConvertLangStr (LPWSTR src, LPWSTR dst, int dstMaxLen);

Входные параметры:

src	- строка для конвертирования,
dstMaxLen	- максимальная длина переведённой строки.

Выходные параметры:

dst	- переведенная строка.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание.

1. Функция устарела, предпочтительнее использовать ksConvertLangStrEx
2. Словарь сохраняется в файле с расширением dic. Строки файла должны иметь следующий формат:

   "Вращение"="Rotation"

   "Параметры"="Parameters"

   ...

   ...

   "Перемещение"="Moving"

   Строки файла должны быть отсортированы по алфавиту.
3. При использовании ANSI следует использовать функцию ksConvertLangStr.

ksConvertLangStrEx – Конвертировать строку с идентификатором в соответствии с текущим словарем

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangStrEx.

Синтаксис:

```
int ksConvertLangStrEx (HINSTANCE hInstance,
unsigned int strID,
char* dst,
int dstMaxLen );
```

Входные параметры:

hInstance - HINSTANCE модуля, в котором находится переводимая строка,
strID - идентификатор строки,
dst - результат - переведенная строка,
dstMaxLen - максимально допустимая длина строки.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksConvertLangStrExW.

ksConvertLangStrExW – Конвертировать строку с идентификатором в соответствии с текущим словарем (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangStrEx.

Синтаксис:

int LIB_FUNC ksConvertLangStrEx (HINSTANCE hInstance,
unsigned int strID,
LPWSTR dst,
int dstMaxLen);

Входные параметры:

hInstance	- HINSTANCE модуля, в котором находится переводимая строка,
strID	- идентификатор строки,
dst	- результат - переведенная строка,
dstMaxLen	- максимально допустимая длина строки.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksConvertLangStrEx.

ksConvertLangWindow – Конвертировать окно с входящими дочерними окнами в соответствии с текущим словарем

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangWindow.

Синтаксис:

int ksConvertLangWindow (HWND hWnd);

Входные параметры:

hWnd	- дескриптор окна,

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

ksConvertLangWindowEx – Конвертировать окно с входящими дочерними окнами в соответствии с текущим словарем

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangWindowEx.

Синтаксис:

int ksConvertLangWindowEx (HWND hWnd,
HINSTANCE hInstance,
char* dlgID);

Входные параметры:

hWnd	- дескриптор окна,
hInstance	- HINSTANCE модуля, в котором находится переводимый диалог,
dlgID	- имя ресурса диалога.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksConvertLangWindowExW.

ksConvertLangWindowExW – Конвертировать окно с входящими дочерними окнами в соответствии с текущим словарем (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangWindowEx.

Синтаксис:

int LIB_FUNC ksConvertLangWindowExW (HWND hWnd,
HINSTANCE hInstance,
LPWSTR dlgID);

Входные параметры:

hWnd	- дескриптор окна,
hInstance	- HINSTANCE модуля, в котором находится переводимый диалог,
dlgID	- имя ресурса диалога.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksConvertLangWindowEx.

ksExecuteLibraryCommand – Выполнить команду другой библиотеки

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksExecuteLibraryCommand.

Синтаксис:

int ksExecuteLibraryCommand (char *fileName, int command);

Входные параметры:

fileName - имя файла прикладной библиотеки,
command - номер команды из библиотеки.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Если задано короткое имя файла прикладной библиотеки (без пути), то эта библиотека ищется в подпапке …\Libs главной папки КОМПАС.
2. При использовании Unicode следует использовать функцию ksExecuteLibraryCommandW.

ksExecuteLibraryCommandW – Выполнить команду другой библиотеки (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksExecuteLibraryCommand.

Синтаксис:

int LIB_FUNC ksExecuteLibraryCommandW (LPWSTR fileName, int command);

Входные параметры:

fileName - имя файла прикладной библиотеки,
command - номер команды из библиотеки.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Если задано короткое имя файла прикладной библиотеки (без пути), то эта библиотека ищется в подпапке …\Libs главной папки КОМПАС.
2. При использовании ANSI следует использовать функцию ksExecuteLibraryCommand.

ksGetLibraryTreeStruct – Получить структуру дерева библиотеки документов и библиотеки атрибутов

Аналог данной функции при использовании Automation - метод KompasObject::ksGetLibraryTreeStruct.

Синтаксис:

int ksGetLibraryTreeStruct (char * libName, TreeNodeParam * root);

Входные параметры:

libName - полное имя файла библиотеки моделей, фрагментов, атрибутов,
root - заполняется параметрами корня дерева.

Структура параметров узла дерева объектов библиотеки TreeNodeParam...

Возвращаемое значение:

0 - в случае неудачи.

Примечание.

При использовании Unicode следует использовать функцию ksGetLibraryTreeStructW.

ksGetLibraryTreeStructW – Получить структуру дерева библиотеки документов и библиотеки атрибутов (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksGetLibraryTreeStruct.

Синтаксис:

int LIB_FUNC ksGetLibraryTreeStructW (LPWSTR libName, TreeNodeParam * root);

Входные параметры:

libName - полное имя файла библиотеки моделей, фрагментов, атрибутов,
root - заполняется параметрами корня дерева.

Возвращаемое значение:

0 - в случае неудачи.

Примечание.

При использовании ANSI следует использовать функцию ksGetLibraryTreeStruct.

ksGetSystemControlStartResult – Проверить, запущен SystemControlStart или нет

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksGetSystemControlStartResult.

Синтаксис:

int ksGetSystemControlStartResult();

Входные параметры:

- Не используется.

Возвращаемое значение:

- тип выхода из режима работы под управлением системы КОМПАС.

ksOpenHelpFile – Открыть файл справки

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksOpenHelpFile

Синтаксис:

void ksOpenHelpFile (char * file, unsigned int command, unsigned int id);

Входные параметры:

file	- полное имя файла Справки,
command	- тип Справки (контекстная, по команде и т.д.),
id	- идентификатор страницы справки.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание :

1. Синтаксис соответствует::WinHelp.
2. Функция открывает файл Справки КОМПАС.
3. При закрытии системы окно Справки закроется автоматически.
4. При использовании Unicode следует использовать функцию ksOpenHelpFileW.

ksOpenHelpFileW – Открыть файл справки (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksOpenHelpFile

Синтаксис:

void LIB_FUNC ksOpenHelpFileW (LPWSTR file, unsigned int command, unsigned int id);

Входные параметры:

file	- полное имя файла Справки,
command	- тип Справки (контекстная, по команде и т.д.),
id	- идентификатор страницы справки.

Возвращаемое значение:

TRUE	- в случае удачного завершения,
FALSE	- в случае неудачи.

Примечание:

1. Синтаксис соответствует::WinHelp.
2. Функция открывает файл Справки КОМПАС.
3. При закрытии системы окно Справки закроется автоматически.
4. При использовании ANSI следует использовать функцию ksOpenHelpFile.

Примечание:

Функция обнуляет результат работы библиотеки, если ошибка не фатальная.

ksSetCurrentLibrary – Установить текущую библиотеку

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangMenu.

Синтаксис:

int ksSetCurrentLibrary (char * libname, long hModule);

Входные параметры:

libname	- имя библиотеки,
hModule	- идентификатор модуля Dll библиотеки.

Возвращаемое значение:

1	- библиотека стала текущей,
0	- в случае ошибки.

Примечание:

1. Назначить библиотеку текущей можно либо по идентификатору модуля Dll библиотеки, либо по имени библиотеки.
2. Данный метод следует использовать осторожно, иначе можно нарушить последовательность выполнения библиотек.
3. Метод может быть применен к подключенной библиотеке.
4. Метод позволяет остановить режим SystemControlStart, запущенный не из выполняемой (текущей) библиотеки.
5. При использовании Unicode следует использовать функцию ksSetCurrentLibraryW.

ksSetCurrentLibraryW – Установить текущую библиотеку (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksConvertLangMenu.

Синтаксис:

int LIB_FUNC ksSetCurrentLibrary (LPWSTR libname, long hModule);

Входные параметры:

libname - имя библиотеки,
hModule - идентификатор модуля Dll библиотеки.

Возвращаемое значение:

1 - библиотека стала текущей,
0 - в случае ошибки.

Примечание:

1. Назначить библиотеку текущей можно либо по идентификатору модуля Dll библиотеки, либо по имени библиотеки.
2. Данный метод следует использовать осторожно, иначе можно нарушить последовательность выполнения библиотек.
3. Метод может быть применен к подключенной библиотеке.
4. Метод позволяет остановить режим SystemControlStart, запущенный не из выполняемой (текущей) библиотеки.
5. При использовании ANSI следует использовать функцию ksSetCurrentLibrary.

ResultNULL– Обнулить результат работы библиотеки, если ошибка не фатальная

Аналог данной функции при использовании Automation - метод KompasObject::ksResultNULL.

Синтаксис:

int ResultNULL();

SystemControlStart – Перейти под управление КОМПАС-ГРАФИК

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksSystemControlStart.

Синтаксис:

int SystemControlStart (char *menuCommand);

Входные параметры:

menuCommand - указатель на строку, помещаемую в меню для возврата в библиотеку.

Возвращаемое значение:

- типы выхода из режима работы под управлением системы КОМПАС.

Примечание:

1. Библиотечная функция отдает управление КОМПАС-ГРАФИК для интерактивной доработки документа. Возврат в библиотечную функцию будет осуществлен после нажатия команды в меню Библиотеки, обозначенной строкой menuCommand.

   Если строка не задана, то ей автоматически будет присвоено значение "Вернуться в библиотеку" (функция вернет scStoppedByMenuCommand).
2. При использовании Unicode следует использовать функцию SystemControlStartW.

SystemControlStartW – Перейти под управление КОМПАС-ГРАФИК (Unicode)

Аналог данной функции при использовании Automation - метод KompasObject::ksSystemControlStart.

Синтаксис:

int LIB_FUNC SystemControlStartW (LPWSTR menuCommand);

Входные параметры:

menuCommand - указатель на строку, помещаемую в меню для возврата в библиотеку.

Возвращаемое значение:

тип выхода из режима работы под управлением системы КОМПАС.

Примечание:

1. Библиотечная функция отдает управление КОМПАС-ГРАФИК для интерактивной доработки документа. Возврат в библиотечную функцию будет осуществлен после нажатия команды в меню Библиотеки, обозначенной строкой menuCommand.

   Если строка не задана, то ей автоматически будет присвоено значение "Вернуться в библиотеку" (функция вернет scStoppedByMenuCommand).
2. При использовании ANSI следует использовать функцию SystemControlStart.

SystemControlStop – Отдать управление библиотеке

Пример...

Аналог данной функции при использовании Automation - метод KompasObject::ksSystemControlStop.

Синтаксис:

void SystemControlStop ();

Примечание:

Функция принудительно передает управление библиотечной функции, запущенной в режиме модельного диалога (в этом случае управление может переходить от библиотеки к системе и наоборот).

Функции работы с базами данных

Функции работы с базами данных обеспечивают работу с базами данных, доступ к которым осуществляется через интерфейс ODBCODBC, и базами данных, сохраненными в формате текстового файла loa.

Работа с базой данных начинается с создания ее блока заголовка функцией CreateDB. Блок заголовка базы данных является управляющей структурой, содержащей имя базы данных и ее параметры. Конкретная привязка блока заголовка к файлу базы данных осуществляется функцией ConnectDB, причем в процессе работы возможно его отсоединение и повторное связывание.

Перед выполнением запроса необходимо создать специальный буфер, называемый отношением. Структура задается блоком функций Relation.....EndRelation, внутри которого определяются перечень полей и их типы. Сам запрос задается функцией DoStatement, а его условие можно определить с помощью функции Condition или ConditionW (Unicode). После выполнения запроса можно последовательно получить с помощью функции ReadRecord все записи базы данных, соответствующие условию запроса.

CloseTextFile – Закрыть текстовый файл запросов

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCloseTextFile.

Синтаксис:

void CloseTextFile (reference f);

Входной параметр:

f	- указатель на текстовый файл.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Condition – Задать новое условие запроса

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCondition.

Синтаксис:

int Condition (reference bd, reference rel, char * str);

Входные параметры:

bd	- указатель на объект БД,
rel	- указатель на отношение,
str	- запрос.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. При работе с базой данных через ODBC-интерфейс происходит переопределение составляющей SQL-запроса, начинающейся с ключевого слова WHERE. Состав обрабатываемых полей записи, определенный в отношении, не изменяется. При работе с текстовыми файлами использование функции Condition является единственной возможностью определения условия запроса, так как в функции DoStatement определяется только список обрабатываемых полей записи. Следует отметить, что при работе с текстовыми базами данных не обрабатывается вложенность условий (например, "where d > 10 and d < 14").
2. Действительно для запроса Select....
3. При использовании Unicode следует использовать функцию ConditionW.

ConditionW – Задать новое условие запроса (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCondition.

Синтаксис:

int LODBC_FUNC ConditionW (reference bd, reference rel, LPWSTR str);

Входные параметры:

bd	- указатель на объект БД,
rel	- указатель на отношение,
str	- запрос.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. При работе с базой данных через ODBC-интерфейс происходит переопределение составляющей SQL-запроса, начинающейся с ключевого слова WHERE. Состав обрабатываемых полей записи, определенный в отношении, не изменяется. При работе с текстовыми файлами использование функции Condition является единственной возможностью определения условия запроса, так как в функции DoStatement определяется только список обрабатываемых полей записи. Следует отметить, что при работе с текстовыми базами данных не обрабатывается вложенность условий (например, "where d > 10 and d < 14").
2. Действительно для запроса Select....
3. При использовании ANSI следует использовать функцию Condition.

ConnectDB – Связать заголовок и базу данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksConnectDB.

Синтаксис:

int ConnectDB (reference bd, char * dataBaseName);

Входные параметры:

bd	- указатель на блок заголовка, созданный при помощи функции CreateDB.
dataBaseName	- имя БД (для базы данных, доступной через интерфейс ODBC - имя БД в администраторе ODBC, для базы данных текстового формата - имя файла).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция связывает блок заголовка с указателем bd и базу данных с именем dataBaseName. Если база данных представлена в виде текстового файла, то функция работает аналогично стандартной функции WINDOWS OpenFile. При режиме работы через ODBC-интерфейс имя базы данных задается в файле ODBC.INI, содержащем список определенных баз данных.
2. При использовании Unicode следует использовать функцию ConnectDBW.

ConnectDBW – Связать заголовок и базу данных (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksConnectDB.

Синтаксис:

int LODBC_FUNC ConnectDBW (reference bd, LPWSTR dataBaseName);

Входные параметры:

bd	- указатель на блок заголовка, созданный при помощи функции CreateDB.
dataBaseName	- имя БД (для базы данных, доступной через интерфейс ODBC - имя БД в администраторе ODBC, для базы данных текстового формата - имя файла).

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Функция связывает блок заголовка с указателем bd и базу данных с именем dataBaseName. Если база данных представлена в виде текстового файла, то функция работает аналогично стандартной функции WINDOWS OpenFile. При режиме работы через ODBC-интерфейс имя базы данных задается в файле ODBC.INI, содержащем список определенных баз данных.
2. При использовании ANSI следует использовать функцию ConnectDB.

CreateDB – Создать блок заголовка базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCreateDB.

Синтаксис:

reference CreateDB(char *s);

Входной параметр:

s - тип базы данных:
TXT_DB - база данных текстового формата,
ODBC_DB - база данных, доступная через интерфейс ODBC.

Возвращаемое значение:

указатель на блок заголовка базы данных - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция позволяет открыть БД. Одновременно может быть открыто несколько БД.
2. При использовании Unicode следует использовать функцию CreateDBW.

CreateDBW – Создать блок заголовка базы данных (Unicode)

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCreateDB.

Синтаксис:

reference CreateDBW(LPWSTR s);

Входной параметр:

s - тип базы данных:
TXT_DB - база данных текстового формата,
ODBC_DB - база данных, доступная через интерфейс ODBC.

Возвращаемое значение:

указатель на блок заголовка базы данных - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция позволяет открыть БД. Одновременно может быть открыто несколько БД.
2. При использовании ANSI следует использовать функцию CreateDB.данных с доступом через интерфейс ODBC.

DeleteDB – Удалить блок заголовка базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksDeleteDB.

Синтаксис:

int DeleteDB(reference bd);

Входной параметр:

bd - указатель на объект БД.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Функция позволяет закрыть БД. После ее выполнения происходит автоматическое отсоединение базы данных и удаление всех созданных ранее запросов.

DisconnectDB – Отсоединить блок заголовка от базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksDisconnectDB.

Синтаксис:

int DisconnectDB(reference bd);

Входной параметр:

bd - указатель на блок заголовка, созданный при помощи функции CreateDB.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

При удалении блока заголовка (функция DeleteDB) отсоединение выполняется автоматически, поэтому использовать ее рекомендуется только при переопределении базы данных.

DoStatement – Выполнить запрос базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksDoStatement.

Синтаксис:

int DoStatement (reference bd, reference rel, char * str);

Входные параметры:

bd - указатель на объект БД,

rel	- действительный указатель на отношение,
str	- запрос.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Пример запроса выборки из БД:

Select d, s, p from bolt where d = 10

Где d, s, p - названия колонок или * для всех колонок или номера колонок "2, 4, 7" для текстового файла, начиная с единицы слева направо, bolt - имя таблицы в БД или "" для всех колонок текстового файла, d - имя колонки в отношении.

Нужен действительный указатель r на отношение.

Пример запроса для вставки строки в таблицу bolt.

Insert into bolt (d,p,s) values(10, 1.5, 14)

Пример запроса для удаления строки из таблицы bolt.

Delete from bolt where d = 10

Пример запроса для замены данных в строке таблицы bolt.

Update bolt set p = 2.5, s = 20 where d =10

Для ODBC баз отношение не обязательно в случае Insert, Delete, Update. Для текстового файла отношение в этом случае необходимо, чтобы определить имена колонок

Примечание:

Параметр str при работе через ODBC-интерфейс содержит строку SQL-запроса, а при работе с текстовыми файлами - номера полей (колонок) или пустую строку (если обрабатываются все поля таблицы). Для текстовых баз данных созданный запрос будет обрабатывать все записи, а непосредственно условие выборки определяется функцией Condition.

DoStatementW – Выполнить запрос базы данных (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksDoStatement.

Синтаксис:

int LODBC_FUNC DoStatementW (reference bd, reference rel, LPWSTR str);

Входные параметры:

bd	- указатель на объект БД,
rel	- действительный указатель на отношение,
str	- запрос.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Пример запроса выборки из БД:

Select d, s, p from bolt where d = 10

Где d, s, p - названия колонок или * для всех колонок или номера колонок "2, 4, 7" для текстового файла, начиная с единицы слева направо, bolt - имя таблицы в БД или "" для всех колонок текстового файла, d - имя колонки в отношении.

Нужен действительный указатель r на отношение.

Пример запроса для вставки строки в таблицу bolt.

Insert into bolt (d,p,s) values(10, 1.5, 14)

Пример запроса для удаления строки из таблицы bolt.

Delete from bolt where d = 10

Пример запроса для замены данных в строке таблицы bolt.

Update bolt set p = 2.5, s = 20 where d =10

Для ODBC баз отношение не обязательно в случае Insert, Delete, Update. Для текстового файла отношение в этом случае необходимо, чтобы определить имена колонок.

Примечание:

Параметр str при работе через ODBC-интерфейс содержит строку SQL-запроса, а при работе с текстовыми файлами - номера полей (колонок) или пустую строку (если обрабатываются все поля таблицы). Для текстовых баз данных созданный запрос будет обрабатывать все записи, а непосредственно условие выборки определяется функцией Condition.

EndRelation – Завершить описание отношения

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksEndRelation.

Синтаксис:

void EndRelation();

Примечание:

Функция завершает описание отношения, начатое функцией Relation.

FreeStatement – Освободить запрос базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksFreeStatement.

Синтаксис:

int FreeStatement (reference bd, reference rel, unsigned short int fOption);

Входные параметры:

bd	- указатель на объект БД,
rel	- указатель на отношение,
fOption	- тип освобождения.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Происходит освобождение памяти, отведенной для буфера записи функцией Relation. Параметр fOption имеет значение при работе через ODBC-интерфейс и означает тип освобождения (описан в помощи модуля ODBC).

GetColumnName – Считать имя колонки таблицы из базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksGetColumnName.

Синтаксис:

int GetColumnName (reference db,

char * tableName,

char * columnName,

int size,

unsigned char firstOrNext);

Входные параметры:

db	- указатель на базу данных,
tableName	- ODBC - имя таблицы, текстовая БД - имя файла,
size	- размер буфера,
firstOrNext	- признак колонки:
	F - первая колонка,
	N - следующая колонка.

Выходной параметр:

columnName	- имя колонки.

Возвращаемое значение:

1 - если в указанной базе еще существуют несчитанные имена колонок таблиц,

0 - если все имена колонок таблиц считаны.

Примечание.

При использовании Unicode следует использовать функцию GetColumnNameW.

GetColumnNameW – Считать имя колонки таблицы из базы данных (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksGetColumnName.

Синтаксис:

int LODBC_FUNC GetColumnName (reference db,

LPWSTR tableName,

LPWSTR columnName,

int size,

unsigned char firstOrNext);

Входные параметры:

db	- указатель на базу данных,
tableName	- ODBC - имя таблицы, текстовая БД - имя файла,
size	- размер буфера,
firstOrNext	- признак колонки:
	F - первая колонка,
	N - следующая колонка.

Выходной параметр:

columnName - имя колонки.

Возвращаемое значение:

1 - если в указанной базе еще существуют несчитанные имена колонок таблиц,
0 - если все имена колонок таблиц считаны.

Примечание:

При использовании ANSI следует использовать функцию GetColumnName.

GetTableName – Считать имя таблицы из базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksGetTableName.

Синтаксис:

int GetTableName (reference db,

char * tableName,

int size,

unsigned char firstOrNext);

Входные параметры:

db	- указатель на объект БД,
firstOrNext	- признак таблицы: F - первая таблица, N - следующая таблица,
size	- размер буфера.

Выходной параметр:

tableName	- имя таблицы.

Возвращаемое значение:

1	- если в указанной базе еще существуют несчитанные имена таблиц,
0	- если все имена таблиц считаны.

Примечание.

При использовании Unicode следует использовать функцию GetTableNameW.

GetTableNameW – Считать имя таблицы из базы данных (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksGetTableName.

Синтаксис:

int LODBC_FUNC GetTableName (reference db,

LPWSTR tableName,

int size,

unsigned char firstOrNext);

Входные параметры:

db	- указатель на объект БД,
firstOrNext	- признак таблицы: F - первая таблица, N - следующая таблица,
size	- размер буфера.

Выходной параметр:

tableName - имя таблицы.

Возвращаемое значение:

1 - если в указанной базе еще существуют несчитанные имена таблиц,
0 - если все имена таблиц считаны.

Примечание.

При использовании ANSI следует использовать функцию GetTableName.

IsODBCOkey – Проверить подключение ODBC

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksIsODBCOkey.

Синтаксис:

int IsODBCOkey();

Возвращаемое значение:

1 - если соединение с ODBC установлено,
0 - если соединения нет.

ksOpenTextFileEx – Открыть текстовый файл, в котором хранятся SQL запросы

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCloseTextFile.

Синтаксис:

reference ksOpenTextFileEx(char * fileName, int textFileType);

Входные параметры:

fileName - имя файла,
textFileType - тип текстового файла:
0 - текстовая база данных
1 - простой текстовый файл
2 - текстовый файл с комментариями (комментарии не выдаются).

Примечание:

Комментарии могут иметь следующие варианты формата:

```
/*
  многострочный
  комментарий
*/
```

// однострочный комментарий

ksOpenTextFileExW – Открыть текстовый файл, в котором хранятся SQL запросы (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksCloseTextFile.

Синтаксис:

reference ksOpenTextFileExW(LPWSTR fileName, int textFileType);

Входные параметры:

fileName - имя файла,
textFileType - тип текстового файла:
0 - текстовая база данных
1 - простой текстовый файл
2 - текстовый файл с комментариями (коментарии не выдаются).

Примечание:

Комментарии могут иметь следующие варианты формата:

/*

многострочный

комментарий

*/

// однострочный комментарий

OpenTextFile – Открыть текстовый файл запросов

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksOpenTextFile.

Синтаксис:

reference OpenTextFile(char * fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

указатель на открытый файл - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция работает по правилам стандартной процедуры WINDOWS OpenFile.
2. Реализованные в рамках данного раздела функции работы с текстовыми файлами обеспечивают их построчную обработку и позволяют вынести обращения к базе данных во внешний текстовый файл. Это дает возможность работать с разными платформами СУБД без изменения выполняемого кода приложения.
3. При использовании Unicode следует использовать функцию OpenTextFileW.

OpenTextFileW – Открыть текстовый файл запросов (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksOpenTextFile.

Синтаксис:

reference LODBC_FUNC OpenTextFileW(LPWSTR fileName);

Входной параметр:

fileName - имя файла.

Возвращаемое значение:

указатель на открытый файл - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Функция работает по правилам стандартной процедуры WINDOWS OpenFile.
2. Реализованные в рамках данного раздела функции работы с текстовыми файлами обеспечивают их построчную обработку и позволяют вынести обращения к базе данных во внешний текстовый файл. Это дает возможность работать с разными платформами СУБД без изменения выполняемого кода приложения.
3. При использовании ANSI следует использовать функцию OpenTextFile.

RChar – Определить строковое поле в отношении

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRChar.

Синтаксис:

int RChar (char *name, int size, long int type);

Входные параметры:

name - имя колонки таблицы,
size - размер буфера,

type - тип данных, хранящихся в БД (действительно для ODBC-баз).

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен при работе с текстовой базой данных.
2. Параметр type имеет смысл только для баз данных, связанных через ODBC, и описывает действительный тип колонки базы данных, с которой будет связано поле при выполнении операции DoStatement. Это позволяет получать строковое представление записи произвольного типа. Описания типов для ODBC хранятся в файле SQL.h, который поставляется вместе с модулями ODBC.
3. При использовании Unicode следует использовать функцию RCharW.

RCharW – Определить строковое поле wchar_t[size] в отношении (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRChar.

Синтаксис:

int LODBC_FUNC RCharW (LPWSTR name, int size, long int type);

Входные параметры:

name - имя колонки таблицы,
size - размер буфера,
type - тип данных, хранящихся в БД (действительно для ODBC-баз).

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен при работе с текстовой базой данных.
2. Параметр type имеет смысл только для баз данных, связанных через ODBC, и описывает действительный тип колонки базы данных, с которой будет связано поле при выполнении операции DoStatement. Это позволяет получать строковое представление записи произвольного типа. Описания типов для ODBC хранятся в файле SQL.h, который поставляется вместе с модулями ODBC.
3. При использовании ANSI следует использовать функцию RChar.

RDouble – Определить double-поле в отношении

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRDouble.

Синтаксис:

int RDouble (char *name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

RDoubleW – Определить double-поле в отношении (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRDouble.

Синтаксис:

int LODBC_FUNC RDouble (LPWSTR name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

Relation – Создать новое отношение

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRelation.

Синтаксис:

reference Relation (reference bd);

Входной параметр:

bd - указатель на объект БД.

Возвращаемое значение:

указатель на отношение - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Отношение характеризует один запрос и представляет собой описание памяти, необходимой для размещения очередной записи выборки, выполняемого функцией ReadRecord. Является составным объектом, каждое поле которого описывает тип и имя поля (колонки) в таблице базы данных. Имя действительно только в случае работы с текстовым файлом, так как при обмене через ODBC-интерфейс имена уже описаны в блоке заголовка. Количество отношений, определенных для базы данных, не ограничивается.

RFloat – Определить в отношении поле типа float

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRFloat.

Синтаксис:

int RFloat (char *name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

RFloatW – Определить в отношении поле типа float (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRFloat.

Синтаксис:

int LODBC_FUNC RFloat (LPWSTR name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Параметр name действителен только при работе с текстовыми базами данных.

RInt – Определить в отношении поле типа short int

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRInt.

Синтаксис:

int RInt (char *name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен только при работе с текстовыми базами данных.
2. При использовании Unicode следует использовать функцию RIntW.

RIntW – Определить в отношении поле типа short int (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRInt.

Синтаксис:

int LODBC_FUNC RIntW (LPWSTR name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен только при работе с текстовыми базами данных.
2. При использовании ANSI следует использовать функцию RInt.

RLong – Определить в отношении поле типа int или long int

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRLong.

Синтаксис:

int RLong (char *name);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен только при работе с текстовыми базами данных.
2. При использовании Unicode следует использовать функцию RLongW.

RLongW – Определить в отношении поле типа int или long int (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksRLong.

Синтаксис:

int LODBC_FUNC RLongW (LPWSTRname);

Входной параметр:

name - имя поля.

Возвращаемое значение:

1 - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

1. Параметр name действителен только при работе с текстовыми базами данных.
2. При использовании ANSI следует использовать функцию RLong.

ReadRecord – Получить запись базы данных

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksReadRecord.

Синтаксис:

int ReadRecord (reference bd, reference rel, void *v);

Входные параметры:

bd	- указатель на объект БД,
rel	- указатель на отношение,
v	- объем памяти для хранения одной записи.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

Количество и типы полей в структуре v должны соответствовать отношению rel.

ReadStrFromTextFile – Считать строку из текстового файла запросов

Пример...

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksReadStrFrFile.

Синтаксис:

int ReadStrFromTextFile (reference f, char * buff, int numb);

Входные параметры:

f	- указатель на текстовый файл,
numb	- номер строки.

Выходной параметр:

buff	- результат работы метода.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Строка обращения к базе данных начинается с номера, далее после двоеточия идет текст обращения, например:

   3:select d,f from Table1 where d=10 and f>d

2. При использовании Unicode следует использовать функцию ReadStrFromTextFileW.

ReadStrFromTextFileW – Считать строку из текстового файла запросов (Unicode)

Аналог данной функции при использовании Automation - метод ksDataBaseObject::ksReadStrFrFile.

Синтаксис:

int LODBC_FUNC ReadStrFromTextFileW (reference f, LPWSTR buff, int numb);

Входные параметры:

f	- указатель на текстовый файл,
numb	- номер строки.

Выходной параметр:

buff	- результат работы метода.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Строка обращения к базе данных начинается с номера, далее после двоеточия идет текст обращения, например:

   3:select d,f from Table1 where d=10 and f>d

2. При использовании ANSI следует использовать функцию ReadStrFromTextFile.

Функции работы с динамическими массивами

Функции данного раздела позволяют использовать динамические массивы.

Такие массивы используются для работы с массивами параметров неопределенной длины. Поддерживаются как стандартные массивы (со строго определенной структурой элементов), так и пользовательские динамические массивы с произвольной структурой элементов. Применение динамических массивов позволяет избежать резервирования памяти из расчета максимально возможной размерности массива. Нумерация элементов массива начинается с 0.

AddArrayItem – Добавить элемент в массив

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksAddArrayItem.

Синтаксис:

int AddArrayItem (reference p, int numb, void * value, int size);

Входные параметры:

p	- указатель на массив,
numb	- индекс элемента, перед которым нужно вставить новый элемент, нумерация начинается с 0, при numb=-1 элемент добавляется в конец массива,
value	- указатель на структуру параметров - источник значений,
size	- размер структуры параметров - источника значений.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

1. В пользовательском динамическом массиве хранятся только указатели на элементы.
2. При добавлении элемента в пользовательский динамический массив параметр size игнорируется.

ClearArray – Очистить массив

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksClearArray.

Синтаксис:

int ClearArray (reference p);

Входной параметр:

p	- указатель на массив.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае ошибки.

Примечание:

1. Из массива удаляются все элементы, но сам массив не удаляется.
2. При удалении пользовательского элемента автоматически вызывается пользовательская функция удаления элемента, заданная пользователем при создании массива.

CreateArray – Создать стандартный или пользовательский динамический массив неопределенной длины

Пример...

Синтаксис:

reference CreateArray (int type, DeleteFunc f);

Входной параметр:

f - функция удаления для пользовательского динамического массива, void WINAPI f(void *), определяется пользователем,
type - Типы динамических массивов.

Возвращаемое значение:

указатель на созданный массив - в случае успеха,
0 - в случае ошибки.

DeleteArray – Удалить массив

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksDeleteArray.

Синтаксис:

int DeleteArray (reference p);

Входной параметр:

p - указатель на массив.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае ошибки.

Примечание:

Может быть удален непустой массив, то есть перед вызовом функции DeleteArray вызывать ClearArray необязательно.

ExcludeArrayItem – Удалить элемент массива

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksExcludeArrayItem.

Синтаксис:

int ExcludeArrayItem (reference p, int numb);

Входной параметр:

p	- указатель на массив,
numb	- номер элемента массива, нумерация начинается с 0.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

GetArrayCount – Получить количество элементов в динамическом массиве

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksGetArrayCount.

Синтаксис:

int GetArrayCount (reference p);

Входной параметр:

p	- указатель на массив.

Возвращаемое значение:

- количество элементов массива.

GetArrayItem – Получить элемент массива

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksGetArrayItem.

Синтаксис:

int GetArrayItem (reference p, int numb, void * value, int size);

Входные параметры:

p - указатель на массив,
numb - номер элемента массива, нумерация начинается с 0,
size - длина области памяти.

Входные параметры:

value - указатель на область памяти, куда будет помещено значение.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

GetArrayType – Получить тип массива

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksGetArrayType.

Синтаксис:

int GetArrayType (reference p);

Входной параметр:

p - указатель на массив.

Возвращаемое значение:

тип динамического массива.

GetUserArrayItem – Получить указатель на элемент пользовательского динамического массива

Пример...

Синтаксис:

int GetUserArrayItem (reference p, int numb, void ** value);

Входной параметр:

p - указатель на массив,
numb - номер элемента массива, нумерация начинается с 0.

Выходной параметр:

value - указатель на пользовательскую структуру параметров элемента массива.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

SetArrayItem – Установить параметры элемента динамического массива

Пример...

Аналог данной функции при использовании Automation - метод ksDynamicArray::ksSetArrayItem.

Синтаксис:

int SetArrayItem (reference p, int numb, void * value, int size);

Входные параметры:

p - указатель на массив,
numb - номер элемента массива, нумерация начинается с 0,
size - длина структуры-источника.

Выходной параметр:

value - указатель на структуру параметров элемента массива.

Возвращаемое значение:

1 - в случае успеха,
0 - в случае неудачи.

Функции работы с атрибутами

Функции данного раздела позволяют работать с атрибутами объектов.

ChoiceAttr – Вывести диалог просмотра атрибутов объекта

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksChoiceAttr.

Синтаксис:

reference ChoiceAttr (reference pObj);

Входной параметр:

pObj - указатель на объект, к которому подключены атрибуты.

Возвращаемое значение:

указатель на атрибут - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Указатель на объект reference pObj может быть получен следующими способами:

- с помощью Итератора - методами CreateIterator, MoveIterator.
- с помощью методов создания объектов, возвращающих указатель reference на созданный объект, например, reference LineSeg.

ChoiceAttrTypes – Выдать диалог для просмотра в библиотеке атрибутов списка типов атрибутов и выбора нужного типа

Пример...

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksChoiceAttrTypes.

Синтаксис:

double ChoiceAttrTypes (char * libname);

Входной параметр:

libName - имя библиотеки типов атрибутов,
если libname = NULL, то тип атрибута берется из текущего документа.

Возвращаемое значение:

уникальный номер типа атрибута - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Диалог "привязан" к главному окну.

CreateAttr – Создать атрибут объекта

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Замечание:

Функция устарела, рекомендуется использовать ksCreateAttr

Синтаксис:

reference CreateAttr (reference pObj,
Attribute * attr,
double attrID,
char *libname);

Входные параметры:

pObj	- указатель на объект (группа, вид, отдельный объект), атрибут которого создается. Если pObj = 0, то создается атрибут документа,
attr	- указатель на структуру параметров табличного атрибута Attribute,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется из текущего документа.

Возвращаемое значение:

указатель на атрибут	- в случае удачного завершения,
0	- в случае неудачи.

CreateAttrIterator – Создать итератор для перебора атрибутов объекта

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksCreateAttrIterator.

Синтаксис:

reference CreateAttrIterator(reference obj,
unsigned int key1,
unsigned int key2,
unsigned int key3,
unsigned int key4,
double numb);

Входные параметры:

obj	- указатель на объект,
key1, key2, key3, key4	- ключи для поиска по ключам,
numb	- номер типа атрибута для поиска по номеру.

Возвращаемое значение:

1	- в случае успешного завершения,
0	- в случае неудачи.

Примечание:

1. Если obj = 0, то производится перемещение по объектам с заданным атрибутом внутри графического документа.
2. Если obj не равен 0, то объектом может быть объект вида, вид, группа, слой, тогда движение будет происходить по атрибутам этого объекта.
3. Если obj не равен 0, то объектом может быть документ, тогда движение будет происходить по атрибутам документа.
4. Итератор работает с атрибутами активного документа.

CreateAttrType – Создать описание типа атрибута

Пример...

Синтаксис:

double CreateAttrType (AttributeType * attrType, char * libname);

Входные параметры:

attrType	- указатель на структура параметров типа атрибута ksAttributeType.
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута создается в текущем документе.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Функция устарела, рекомендуется использовать ksCreateAttrType.

DeleteAttr – Удалить атрибут объекта

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksDeleteAttr.

Синтаксис:

int DeleteAttr (reference pObj, reference PAttr, char * password);

Входные параметры:

pObj	- указатель на объект, к которому подключен атрибут. Если pObj = 0 - атрибут текущего документа,
pAttr	- указатель на удаляемый атрибут,

password - пароль атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Указатель на объект reference pObj может быть получен следующими способами:

- с помощью функций Итератора CreateIterator, MoveIterator.
- с помощью методов создания объектов, возвращающих указатель reference на созданный объект, например, reference LineSeg.

2. Указатель на атрибут reference pAttr может быть получен:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr, или выбора атрибутов ChoiceAttr.

DeleteAttrType – Удалить описание атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksDeleteAttrType.

Синтаксис:

int DeleteAttrType (double attrID, char *libname, char * password);

Входные параметры:

attrID - уникальный номер типа,
libName - имя библиотеки типов атрибутов,
если libname = NULL, то тип атрибута удаляется из текущего документа,
password - пароль типа атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam, которая может быть получена с помощью функции ksGetLibraryAttrTypesArray, или с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

GetAttrColumnInfo – Получить информацию по столбцу

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrColumnInfo.

Синтаксис:

int GetAttrColumnInfo (reference pAttr, unsigned int columnNumb, ColumnInfo *columnInfo);

Входные параметры:

pAttr	- указатель на атрибут,
columnNumb	- номер колонки.

Выходной параметр:

columnInfo	- адрес структуры для записи результата.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

GetAttrKeysInfo – Получить информацию для поиска

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrKeysInfo.

Синтаксис:

int GetAttrKeysInfo (reference pAttr,

unsigned int *key1,

unsigned int *key2,

unsigned int *key3,

unsigned int *key4,

double attrID);

Входной параметр:

pAttr	- указатель на атрибут,

Выходные параметры:

key1, key2, key3, key4	- ключи атрибута для поиска по значению,
attrID	- уникальный номер типа атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

GetAttrRow – Получить строку атрибута

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Синтаксис:

int GetAttrRow (reference pAttr,
unsigned int rowNumb,
unsigned char *flagVisible,
void *value,
unsigned int size);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения,
flagVisible	- указатель на массив флагов видимости ячеек строки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Функция устарела, рекомендуется вместо нее использовать функцию ksGetAttrRow.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

3. Перед использованием указателя flagVisible должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной flagVisible.
4. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
5. Для получения размера данных строки атрибута может быть использована функция ksGetSizeAttrRow.
6. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

GetAttrTabInfo – Получить информацию по табличному атрибуту

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrTabInfo.

Синтаксис:

int GetAttrTabInfo (reference pAttr, unsigned int * rowsCount, unsigned int * columnsCount);

Входной параметр:

pAttr	- указатель на атрибут.

Выходные параметры:

rowsCount	- количество строк атрибута,
columnsCount	- количество столбцов атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

GetAttrType – Получить описание типа атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Функция устарела, рекомендуется использовать ksGetAttrType.

Синтаксис:

int GetAttrType (double attrID, char *libname, AttributeType * attrType);

Входной параметр:

attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута из текущего документа.

Выходной параметр:

attrType	- указатель на структуру типа атрибута....

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

GetAttrValue – Получить значение атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Замечание. Функция устарела, вместо нее использовать ksGetAttrValue.

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrValue.

Синтаксис:

int GetAttrValue (reference pAttr,
unsigned int rowNumb,
unsigned int columnNumb,
unsigned char *flagVisible,
void *value, unsigned int size);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

GetSizeAttrRow – Получить длину строки табличного атрибута

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm`

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrRow.

Синтаксис:

int GetSizeAttrRow (reference pAttr);

Входной параметр:

pAttr	- указатель на атрибут.

Возвращаемое значение:

длина строки атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Функция устарела, рекомендуется вместо нее использовать функцию ksGetSizeAttrRow.

GetSizeAttrValue – Получить длину поля атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrValue.

Синтаксис:

int GetSizeAttrValue (reference pAttr, unsigned int columnNumb);

Входные параметры:

pAttr	- указатель на атрибут,
columnNumb	- номер колонки.

Возвращаемое значение:

длина поля атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Функция устарела, рекомендуется вместо нее использовать функцию ksGetSizeAttrValue.

ksAddAttrRow – Добавить строку к табличному атрибуту

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksAddAttrRow.

Синтаксис:

int ksAddAttrRow (reference pAttr,
int rowNumb,
unsigned char *flagVisible,
void *value,
unsigned int size,
char * password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
value	- указатель на область памяти, откуда копируются данные,

Size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.
2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.
5. При использовании Unicode следует использовать функцию ksAddAttrRowW.

ksAddAttrRowW – Добавить строку к табличному атрибуту (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksAddAttrRow.

Синтаксис:

```
int ksAddAttrRowW (reference pAttr,
int rowNumb,
unsigned char *flagVisible,
void *value,
unsigned int size,
char * password);
```

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
value	- указатель на область памяти, откуда копируются данные,
Size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечания:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfoW для каждой колонки атрибута.
5. При использовании ANSI следует использовать функцию ksAddAttrRow.

ksChoiceAttr3D – Просмотреть атрибуты объекта документа-модели

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksChoiceAttr3D.

Синтаксис:

LPATTRIBUTE3D ksChoiceAttr3D (LPUNKNOWN pObj);

Входные параметры:

pObj - указатель на объект.

Возвращаемое значение:

указатель на интерфейс атрибута IAttribute3D - в случае удачного завершения,
0 - в случае неудачи.

ksCreateAttr – Создать атрибут объекта

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksCreateAttr.

Синтаксис:

reference ksCreateAttr (reference pObj,
ksAttribute * attr,
double attrID,
char *libname);

Входные параметры:

pObj	- указатель на объект (группа, вид, отдельный объект), атрибут которого создается. Если pObj = 0, то создается атрибут документа,
attr	- указатель на структуру параметров атрибута ksAttribute,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется из текущего документа.

Возвращаемое значение:

указатель на атрибут	- в случае удачного завершения,
0	- в случае неудачи.

Примечания:

1. Указатель на объект reference pObj может быть получен следующими способами:

- с помощью функций Итератора CreateIterator, MoveIterator.
- с помощью функций создания объектов, возвращающих указатель reference на созданный объект, например, reference LineSeg.

2. Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam, которая может быть получена с помощью функции ksGetLibraryAttrTypesArray, или с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

ksCreateAttrType – Создать новый тип атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksCreateAttrType.

Синтаксис:

double ksCreateAttrType (ksAttributeType * attrType, char * libname);

Входные параметры:

attrType - attrType - указатель на структуру параметров типа атрибута ksAttributeType,
libName - имя библиотеки типов атрибутов,
если libname = NULL, то тип атрибута создается в текущем документе.

Возвращаемое значение:

уникальный номер типа атрибута - в случае удачного завершения,
0 - в случае неудачи.

ksCreateAttr3D – Создать атрибут по номеру типа атрибута из библиотеки

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksCreateAttr3D.

Синтаксис:

LPATTRIBUTE3D_IFUNC ksCreateAttr3D (LPUNKNOWN pObj, ksAttribute * attr, double attrID, char *libname);

Входные параметры:

pObj - указатель на объект для которого создается атрибут,
attr - указатель на структуру параметров атрибута ksAttribute,
attrID - уникальный номер типа атрибута,
libName - имя библиотеки типов атрибутов,
если libname = NULL, то тип атрибута берется в документе.

Возвращаемое значение:

Указатель на интерфейс атрибута IAttribute3D - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam, которая может быть получена с помощью функции ksGetLibraryAttrTypesArray. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

ksCreateAttr3DEx – Создать атрибут по номеру типа атрибута из библиотеки libname

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksCreateAttr3DEx.

Синтаксис Automation:

ksAttribute3D * ksCreateAttr3DEx (LPDISPATCH pObj,
LPDISPATCH pSourcePart,
LPDISPATCH attr,
double attrID,
BSTR libname);

Входные параметры:

pObj	- указатель на объект для которого создается атрибут,
sourcePart	- указатель на интерфейс вставки детали IPart,
attr	- указатель на структуру параметров атрибута ksAttribute,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется в документе.

Возвращаемое значение:

Указатель на атрибут IAttribute3D	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если libname = NULL - тип атрибута берется в документе.
2. pObj - может быть равен:

- NULL или указателю на интерфейс 3d документа IDocument3D - создается атрибут у документа;
- указателю на интерфейс коллекции объектов дерева IFeatureCollection - атрибут групповой;
- указателю на интерфейс объекта дерева IFeature - атрибут у определенного объекта.

3. Атрибут можно добавить ко всем объектам дерева построений, кроме верхнего компонента, сопряжений, группы сопряжений.
4. sourcePart - может быть:

- NULL - атрибут будет создан в текущем документе,
- указатель на интерфейс детали или подсборки вставленной в сборку IPart - атрибут будет создан в документе-источнике,
- если sourcePart == pObj создается атрибут документа в источнике.

5. При использовании Unicode следует использовать функцию ksCreateAttr3DExW.

ksCreateAttr3DExW – Создать атрибут по номеру типа атрибута из библиотеки libname (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksCreateAttr3DEx.

Синтаксис Automation:

```
ksAttribute3D * LIB_FUNC ksCreateAttr3DExW (LPUNKNOWN pObj,
IPart * pSourcePart,
ksAttributeW * attr,
double attrID,
LPWSTR libname);
```

Входные параметры:

pObj	- указатель на объект для которого создается атрибут,
sourcePart	- указатель на интерфейс вставки детали IPart,
attr	- указатель на структуру параметров атрибута ksAttribute,
attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута берется в документе.

Возвращаемое значение:

Указатель на атрибут IAttribute3D	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Если libname = NULL - тип атрибута берется в документе.
2. pObj - может быть равен:

- NULL или указателю на интерфейс 3d документа IDocument3D - создается атрибут у документа,
- указателю на интерфейс коллекции объектов дерева IFeatureCollection - атрибут групповой,
- указателю на интерфейс объекта дерева IFeature - атрибут у определенного объекта.

3. Атрибут можно добавить ко всем объектам дерева построений, кроме верхнего компонента, сопряжений, группы сопряжений.
4. sourcePart - может быть:

- NULL - атрибут будет создан в текущем документе,
- указатель на интерфейс детали или подсборки вставленной в сборку IPart - атрибут будет создан в документе-источнике,
- если sourcePart == pObj, создается атрибут документа в источнике.

5. При использовании ANSI следует использовать функцию ksCreateAttr3DEx.

ksDeleteAttrRow – Удалить строку табличного атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksDeleteAttrRow.

Синтаксис:

int ksDeleteAttrRow (reference pAttr, unsigned int rowNumb, char * password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

ksDeleteAttr3D – Удалить атрибут

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksDeleteAttr3D.

Синтаксис:

int ksDeleteAttr3D (LPUNKNOWN pObj,
LPATTRIBUTE3D pAttr,
char* password);

Входные параметры:

pObj	- указатель на объект, у которого удаляется атрибут,
pAttr	- указатель на атрибут IAttribute3D,

password - пароль.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksDeleteAttr3DW – Удалить атрибут (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksDeleteAttr3D.

Синтаксис:

int LIB_FUNC ksDeleteAttr3DW (LPUNKNOWN pObj,
LPATTRIBUTE3D pAttr,
LPWSTR password);

Входные параметры:

pObj - указатель на объект, у которого удаляется атрибут,
pAttr - указатель на атрибут IAttribute3D,
password - пароль.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

ksGetAttrRow – Получить строку атрибута

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании Automation - метод ksAttributeObject::ksGetAttrRow.

Синтаксис:

int ksGetAttrRow (reference pAttr,
unsigned int rowNumb,
unsigned char * flagVisible,
unsigned char * columnKeys,
void * value,
unsigned int size);

Входные параметры:

pAttr - указатель на атрибут,

rowNumb	- номер строки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя columnKeys должен быть выделен буфер памяти, и его адрес присвоен переменной columnKeys.
4. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
5. Для получения размера данных ячейки атрибута может быть использована функция ksGetSizeAttrValue.
6. При использовании Unicode следует использовать функцию ksGetAttrRowW.
7. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

ksGetAttrRowW – Получить строку атрибута (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данного метода при использовании Automation - метод ksAttributeObject::ksGetAttrRow.

Синтаксис:

```
int ksGetAttrRowW (reference pAttr,
unsigned int rowNumb,
unsigned char * flagVisible,
unsigned char * columnKeys,
void * value,
unsigned int size );
```

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки.

Возвращаемое значение:

1	- в случае удачного завершения.
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя columnKeys должен быть выделен буфер памяти, и его адрес присвоен переменной columnKeys.
4. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
5. Для получения размера данных ячейки атрибута может быть использована функция ksGetSizeAttrValue.
6. При использовании ANSI следует использовать функцию ksGetAttrRow.
7. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfoW для каждой колонки атрибута.

ksGetAttrType – Получить описание типа табличного атрибута из библиотеки

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrType.

Синтаксис:

int ksGetAttrType (double attrID, char * libname, ksAttributeType * attrType);

Входной параметр:

attrID - уникальный номер типа атрибута,
libName - имя библиотеки типов атрибутов,
если libname = NULL, то тип атрибута из текущего документа.

Выходной параметр:

attrType - указатель на структуру типа атрибута AttributeType....

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam, которая может быть получена с помощью функции ksGetLibraryAttrTypesArray, или с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

ksGetAttrTypeInfo – Выдать информацию о типе атрибута

Синтаксис:

int LIB_FUNC ksGetAttrTypeInfo (reference pAttr,

double * numb,

char * attrTypeName,

unsigned int attrTypeNameSize,

char * attrTypeFileName,

unsigned int attrTypeFileNameSize,

char * attrTypePathInFile,

unsigned int attrTypePathInFileSize);

Входные параметры:

pAttr - указатель на атрибут,
attrTypeNameSize - размер буфера имени типа атрибута,
attrTypeFileNameSize - размер буфера имени файла с описанием типа атрибута,
attrTypePathInFileSize - размер буфера для пути к описанию типа атрибута внутри библиотеки типов.

Выходные параметры:

numb - номер типа атрибута,
attrTypeName - имя типа атрибута,
attrTypeFileName - имя файла, в котором описан тип атрибута,
attrTypePathInFile - путь к описанию типа атрибута внутри библиотеки типов.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Если присылается NULL, соответствующее поле не заполняется.

ksGetAttrTypeInfoW – Выдать информацию о типе атрибута (Unicode)

Синтаксис:

int LIB_FUNC ksGetAttrTypeInfoW(reference pAttr,

double * numb,

LPWSTR attrTypeName,

unsigned int attrTypeNameSize,

LPWSTR * attrTypeFileName,

unsigned int attrTypeFileNameSize,

LPWSTR * attrTypePathInFile,

unsigned int attrTypePathInFileSize);

Входные параметры:

pAttr - указатель на атрибут,
attrTypeNameSize - размер буфера имени типа атрибута,
attrTypeFileNameSize - размер буфера имени файла с описанием типа атрибута,
attrTypePathInFileSize - размер буфера для пути к описанию типа атрибута внутри библиотеки типов.

Выходные параметры:

numb - номер типа атрибута,
attrTypeName - имя типа атрибута,
attrTypeFileName - имя файла, в котором описан тип атрибута,
attrTypePathInFile - путь к описанию типа атрибута внутри библиотеки типов.

Возвращаемое значение:

TRUE - в случае удачного завершения,
FALSE - в случае неудачи.

Примечание:

Если присылается NULL, соответствующее поле не заполняется.

ksGetAttrValue – Получить значение ячейки из таблицы атрибута

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrValue.

Синтаксис:

int ksGetAttrValue (reference pAttr,

unsigned int rowNumb,

unsigned int columnNumb,

unsigned char *flagVisible,

unsigned char *columnKeys,

void *value,

unsigned int size);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Для нетабличного атрибута номер колонки равен нулю.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

3. .Для получения размера данных ячейки атрибута может быть использована функция ksGetSizeAttrValue.

4. Перед использованием указателей value, flagVisible, columnKeys для каждого указателя должен быть выделен буфер памяти, и его адрес присвоен соответствующей переменной.
5. При использовании Unicode следует использовать функцию ksGetAttrValueW.
6. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

ksGetAttrValueW – Получить значение ячейки из таблицы атрибута (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetAttrValue.

Синтаксис:

int ksGetAttrValueW (reference pAttr,

unsigned int rowNumb,

unsigned int columnNumb,

unsigned char *flagVisible,

unsigned char *columnKeys,

void *value,

unsigned int size);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,
size	- размер выделенной под value памяти.

Выходной параметр:

value	- указатель на область памяти, куда копируются значения,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Для нетабличного атрибута номер колонки равен нулю.
2. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

3. .Для получения размера данных ячейки атрибута может быть использована функция ksGetSizeAttrValue.
4. Перед использованием указателей value, flagVisible, columnKeys для каждого указателя должен быть выделен буфер памяти, и его адрес присвоен соответствующей переменной.
5. При использовании ANSI следует использовать функцию ksGetAttrValue.
6. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfoW для каждой колонки атрибута.

ksGetLibraryAttrTypesArray – Получить массив типов атрибутов, находящихся в указанной библиотеке

Пример...

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetLibraryAttrTypesArray.

Синтаксис:

reference ksGetLibraryAttrTypesArray (char * libname);

Входной параметр:

libName	- имя библиотеки типов атрибутов. Если libname = NULL, то типы атрибутов из текущего документа.

Возвращаемое значение:

указатель на динамический массив ksDynamicArray LIBRARY_ATTR_TYPE_ARR - массив типов атрибутов, находящихся в заданной библиотеке типов	- в случае удачного завершения,
0	- в случае неудачи.

ksGetSizeAttrRow – Получить длину строки указанного табличного атрибута с указателем

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrRow.

Синтаксис:

int ksGetSizeAttrRow (reference pAttr, int * count);

Входной параметр:

pAttr - указатель на атрибут.

Выходной параметр:

count - число ячеек с учетом записей.

Возвращаемое значение:

длина строки атрибута - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr..
2. При использовании Unicode следует использовать функцию ksGetSizeAttrRowW.

ksGetSizeAttrRowW – Получить длину строки указанного табличного атрибута с указателем (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrRowW.

Синтаксис:

int ksGetSizeAttrRowW (reference pAttr, int * count);

Входной параметр:

pAttr - указатель на атрибут.

Выходной параметр:

count - число ячеек с учетом записей.

Возвращаемое значение:

длина строки атрибута - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr..
2. При использовании ANSI следует использовать функцию ksGetSizeAttrRow.

ksGetSizeAttrValue – Получить размер данных ячейки атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrValue.

Синтаксис:

int ksGetSizeAttrValue (reference pAttr, unsigned int columnNumb, int *count);

Входные параметры:

pAttr	- указатель на атрибут,
columnNumb	- номер колонки.

Выходной параметр:

count	- число ячеек с учетом записей.

Возвращаемое значение:

длина строки ячейки атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr..
2. При использовании Unicode следует использовать функцию ksGetSizeAttrValueW.

ksGetSizeAttrValueW – Получить размер данных ячейки атрибута (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/505_62_1_2_Opisanie_struktury.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksGetSizeAttrValueW.

Синтаксис:

int ksGetSizeAttrValueW (reference pAttr, unsigned int columnNumb, int *count);

Входные параметры:

pAttr	- указатель на атрибут,
columnNumb	- номер колонки.

Выходной параметр:

count	- число ячеек с учетом записей.

Возвращаемое значение:

длина строки ячейки атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr..
2. При использовании ANSI следует использовать функцию ksGetSizeAttrValue.

ksSetAttrRow – Установить строку атрибута

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrRow.

Синтаксис:

int ksSetAttrRow (reference pAttr,

unsigned int rowNumb,

unsigned char *flagVisible,

unsigned char *columnKeys,

void *value,

unsigned int size,

char *password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки,
value	- указатель на область памяти, откуда копируются данные,
size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также:

ksLtVariant

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.
5. При использовании Unicode следует использовать функцию ksSetAttrRowW.

ksSetAttrRowW – Установить строку атрибута (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrRow.

Синтаксис:

int ksSetAttrRowW (reference pAttr,

unsigned int rowNumb,

unsigned char *flagVisible,

unsigned char *columnKeys,

void *value,

unsigned int size,

char *password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- указатель на массив ключей ячеек строки,
value	- указатель на область памяти, откуда копируются данные,
Size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Смотрите также:

ksLtVariant

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfoW для каждой колонки атрибута.
5. При использовании ANSI следует использовать функцию ksSetAttrRow.

ksSetAttrType – Изменить тип атрибута в библиотеке типов атрибутов

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/510_62_3_Upravlenie_tipami_atri.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrType.

Синтаксис:

double ksSetAttrType (double attrID,
char * libname,
ksAttributeType * attrType,
char * password);

Входные параметры:

attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то тип атрибута изменяется в текущем документе,
attrType	- Указатель на структуру параметров типа атрибута...,
password	- пароль типа атрибута.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam, которая может быть получена с помощью функции ksGetLibraryAttrTypesArray, или с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

ksSetAttrValue – Установить значение атрибута

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrValue.

Синтаксис:

int ksSetAttrValue(reference pAttr,

unsigned int rowNumb,

unsigned int columnNumb,

unsigned char *flagVisible,

unsigned char *columnKeys,

void *value,

unsigned int size,

char *password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,

flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- ключ поля колонки (одно значение, если не запись и массив значений, если запись),
value	- указатель на область памяти, откуда копируется значение ячейки,
size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:
- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.
2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. При использовании Unicode следует использовать функцию ksSetAttrValueW.
5. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

ksSetAttrValueW – Установить значение атрибута (Unicode)

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrValue.

Синтаксис:

```
int ksSetAttrValueW(reference pAttr,
unsigned int rowNumb,
unsigned int columnNumb,
unsigned char *flagVisible,
unsigned char *columnKeys,
void *value,
unsigned int size,
char *password);
```

Входные параметры:

pAttr	- указатель на атрибут,

rowNumb	- номер строки,
columnNumb	- номер колонки,
flagVisible	- указатель на массив флагов видимости ячеек строки,
columnKeys	- ключ поля колонки (одно значение, если не запись и массив значений, если запись),
value	- указатель на область памяти, откуда копируется значение ячейки,
size	- размер выделенной под value памяти,
password	- пароль атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

1. Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

2. Перед использованием указателя flagVisible должен быть выделен буфер памяти, и его адрес присвоен переменной flagVisible.
3. Перед использованием указателя value должен быть выделен буфер памяти, например, char buffer[MAX_TEXT_LENGTH], и его адрес присвоен переменной value.
4. При использовании ANSI следует использовать функцию ksSetAttrValue.
5. Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfoW для каждой колонки атрибута.

MoveAttrIterator – Перебирать атрибуты по итератору

Пример...

Аналог данной функции при использовании Automation - метод ksIterator::ksMoveAttrIterator.

Синтаксис:

reference MoveAttrIterator (reference iterator,

unsigned char ch,

reference * pObj);

Входные параметры:

iterator	- указатель на итератор,
ch	- направление перемещения итератора: F - первый атрибут, N - следующий атрибут,
pObj	- указатель на группу.

Возвращаемое значение:

указатель на атрибут - в случае успешного завершения,
0 - в случае неудачи.

Примечание:

Если итератор создан для движения по элементам с определенным атрибутом, то pObj - указатель на объект с данным атрибутом. Если pObj = NULL, pObj не заполняется.

SetAttrRow – Установить строку табличного атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Замечание. Функция устарела, вместо неё использовать ksSetAttrRow.

Синтаксис:

int SetAttrRow (reference pAttr,
unsigned int rowNumb,
unsigned char *flagVisible,
void *value,
unsigned int size,
char * password);

Входные параметры:

pAttr - указатель на атрибут,
rowNumb - номер строки,
flagVisible - указатель на массив флагов видимости ячеек строки,
value - указатель на область памяти, откуда копируются данные,
size - размер выделенной под value памяти,
password - пароль атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

SetAttrType – Установить описание типа атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Функция устарела, рекомендуется использовать ksSetAttrType.

Синтаксис:

int SetAttrType (double attrID,
char *libname,
AttributeType * attrType,
char * password);

Входные параметры:

attrID	- уникальный номер типа атрибута,
libName	- имя библиотеки типов атрибутов, если libname = NULL, то обрабатывается локальный атрибут текущего документа.
attrType	- Структура типа атрибута...,
password	- пароль типа атрибута.

Возвращаемое значение:

уникальный номер типа атрибута	- в случае удачного завершения,
0	- в случае неудачи.

SetAttrValue – Установить значение атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/502_Glava61_Obshchie_svedenija_.htm

Замечание. Функция устарела, вместо неё использовать ksSetAttrValue.

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksSetAttrValue.

Синтаксис:

int SetAttrValue (reference pAttr,
unsigned int rowNumb,
unsigned int columnNumb,
unsigned char *flagVisible,
void *value,
unsigned int size, char * password);

Входные параметры:

pAttr	- указатель на атрибут,
rowNumb	- номер строки,
columnNumb	- номер колонки,
flagVisible	- флаг видимости ячейки (одно значение, если не запись и массив значений, если запись),
value	- указатель на область памяти, откуда копируется значение ячейки,

size - размер выделенной памяти под value,
password - пароль атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Тип данных, которые находятся в области памяти с указателем value, описывается структурой параметров ColumnInfo для каждой колонки атрибута.

ViewEditAttr – Вывести диалог для просмотра или редактирования атрибута

Пример...

Справка системы КОМПАС...

`KOMPAS.chm::/DLG_ATR_LST_BODY_ATR.htm`

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksViewEditAttr.

Синтаксис:

int ViewEditAttr (reference pAttr, unsigned int type, char * password);

Входные параметры:

pAttr - указатель на атрибут,
type - режим работы:
1 - режим просмотра,
2 - режим редактирования,
password - пароль типа атрибута.

Возвращаемое значение:

1 - в случае удачного завершения,
0 - в случае неудачи.

Примечание:

Указатель на атрибут reference pAttr может быть получен следующими способами:

- с помощью функций итератора CreateAttrIterator, MoveAttrIterator,
- с помощью функций создания атрибутов ksCreateAttr или выбора атрибутов ChoiceAttr.

ViewEditAttrType – Вывести диалог для просмотра или редактирования типа атрибута

Пример...

Справка системы КОМПАС...

KOMPAS.chm::/DLG_ATR_NEW_TYPE.htm

Аналог данной функции при использовании Automation - метод ksAttributeObject::ksViewEditAttrType.

Синтаксис:

int ViewEditAttrType (char * libname, unsigned int type, double attrId, char * password);

Входные параметры:

libName	- имя библиотеки типов атрибутов. Если libname = NULL, типы атрибутов берутся из текущего документа,
type	- режим работы: - 1 - режим просмотра, 2 - режим редактирования,
attrId	- уникальный номер типа атрибута,
password	- пароль типа атрибута.

Возвращаемое значение:

1	- в случае удачного завершения,
0	- в случае неудачи.

Примечание:

Уникальный номер типа атрибута attrID может быть получен с помощью поля typeid структуры параметров типа атрибута LibraryAttrTypeParam или с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.

Функции работы с событиями

Функции данного раздела обеспечивают обработку событий.

ksConnectionAdvise – Подписаться на событие

Синтаксис:

int ksConnectionAdvise (NotifyConnectionParam * param, LPUNKNOWN object);

Входные параметры:

param	- указатель на структуру параметров подписки на события,
object	- указатель интерфейса событий.

Возвращаемое значения:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Система выполнит AddRef на object и запомнит в текущей библиотеке или в текущей notify-библиотеке.

ksConnectionUnAdvise – Снять подписку на событие

Синтаксис:

int ksConnectionUnadvise (NotifyConnectionParam * param);

Входные параметры:

param	- указатель на структуру параметров подписки на события.

Возвращаемое значение:

1	- в случае успеха,
0	- в случае неудачи.

Примечание:

Для текущей библиотеки или текущей notify-библиотеки выполнит Release на соответствующий указатель событий.

Интерфейсы событий

Структуры параметров и константы (список)

A...B

C

D...FG...KLM...OPQ-RSTU...Z

Структуры параметров атрибутов

Attribute – Структура параметров табличного атрибута

unsigned int	key1	рекомендуется как код разработчика
unsigned int	key2	рекомендуется как код атрибута
unsigned int	key3	рекомендуется как код разработчика
unsigned int	key4	системный код атрибута
unsigned char	*flagVisible	массив, определяющий для каждой колонки атрибута видимость или невидимость: (0 - видимое поле, 1- невидимое поле)
void	*values	массив значений ячеек таблицы атрибутов (сначала все значения для первой строки, затем все значения для второй строки и т.д.)
unsigned int	valSize	размер массива значений ячеек
char	password [10]	пароль, если не пустая строка - защищает от несанкционированного изменения информации в атрибуте

Примечания:

1. Значения параметра key4 от 0 до 1000 зарезервированы за ОАО "АСКОН"
2. Эта структура и использующая ее функция CreateAttr устарели. Рекомендуется вместо них использовать структуру ksAttribute и функцию ksCreateAttr.

ksAttribute – Структура параметров атрибута

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksAttributeParam.

unsigned int	key1	ключ (дополнительный идентификатор) для поиска атрибута
unsigned int	key2	ключ (дополнительный идентификатор) для поиска атрибута
unsigned int	key3	ключ (дополнительный идентификатор) для поиска атрибута

unsigned int	key4	ключ (дополнительный идентификатор) для поиска атрибута
unsigned char	*flagVisible	массив, определяющий для каждой колонки атрибута видимость или невидимость (0 - видимое поле, 1- невидимое поле)
void	*values	массив значений ячеек таблицы атрибутов (сначала все значения для первой строки, затем все значения для второй строки и т.д.)
unsigned int	valSize	размер массива значений ячеек
char	password[10]	пароль, если не пустая строка - защищает от несанкционированного изменения информации в атрибуте
unsigned char	*columnKeys	массив ключей колонок

Примечание:

1. При создании атрибута, например, с помощью функции ksCreateAttr, ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам CreateAttrIterator. При присвоении значений ключам рекомендуется для ключей key1, key3 присваивать код, идентифицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ОАО"АСКОН".
2. Значения ключей атрибута могут быть получены с помощью функции GetAttrKeysInfo.
3. Перед использованием каждого из указателей: flagVisible, values, columnKeys должен быть выделен буфер памяти, и его адрес присвоен соответствующей переменной из указанных.
4. При задании значений параметров flagVisible и columnKeys необходимо учитывать типы данных элементов атрибута. Колонка атрибута может содержать данные типа Запись. Запись может, в свою очередь, состоять из нескольких колонок. Массивы флагов и ключей должны содержать количество элементов, равное суммарному количеству колонок, включая колонки, содержащие данные типа Запись.

   Например, если атрибут состоит из пяти колонок, то массив будет состоять из пяти элементов. Если в одной из колонок данные имеют тип Запись, которая содержит 2 колонки, то массив будет состоять из семи элементов. Флаг видимости колонки типа Запись позволяет управлять отображением всей колонки. Если колонка отображается (значение для нее равно 1), то флаги для каждой из колонок записи позволяют управлять ее отображением. Если колонка атрибута не отображается (значение для нее равно 0), то вне зависимости от значений флагов для колонок записи, колонка не будет отображаться.
5. При использовании Unicode следует использовать структуру параметров ksAttributeW.

ksAttributeW – Структура параметров атрибута (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksAttributeParam.

unsigned int	key1	ключ (дополнительный идентификатор) для поиска атрибута
unsigned int	key2	ключ (дополнительный идентификатор) для поиска атрибута
unsigned int	key3	ключ (дополнительный идентификатор) для поиска атрибута
unsigned int	key4	ключ (дополнительный идентификатор) для поиска атрибута
unsigned char	*flagVisible	массив, определяющий для каждой колонки атрибута видимость или невидимость (0 - видимое поле, 1- невидимое поле)
void	*values	массив значений ячеек таблицы атрибутов (сначала все значения для первой строки, затем все значения для второй строки и т.д.)
unsigned int	valSize	размер массива значений ячеек
wchar_t	password[10]	пароль, если не пустая строка - защищает от несанкционированного изменения информации в атрибуте
unsigned char	*columnKeys	массив ключей колонок

Примечание:

1. При создании атрибута, например, с помощью функции ksCreateAttr, ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам CreateAttrIterator. При присвоении значений ключам рекомендуется для ключей key1, key3 присваивать код, идентифицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ОАО "АСКОН".
2. Значения ключей атрибута могут быть получены с помощью функции GetAttrKeysInfo.
3. Перед использованием каждого из указателей: flagVisible, values, columnKeys должен быть выделен буфер памяти, и его адрес присвоен соответствующей переменной из указанных.
4. При задании значений параметров flagVisible и columnKeys необходимо учитывать типы данных элементов атрибута. Колонка атрибута может содержать данные типа Запись. Запись может, в свою очередь, состоять из нескольких колонок. Массивы флагов и ключей должны содержать количество элементов, равное суммарному количеству колонок, включая колонки, содержащие данные типа Запись.

   Например, если атрибут состоит из пяти колонок, то массив будет состоять из пяти элементов. Если в одной из колонок данные имеют тип Запись, которая содержит 2 колонки, то массив будет состоять из семи элементов. Флаг видимости колонки типа Запись позволяет управлять отображением всей колонки. Если колонка отображается (значение для нее равно 1), то флаги для каждой из колонок записи позволяют управлять ее отображением. Если колонка атрибута не отображается (значение для нее равно 0), то вне зависимости от значений флагов для колонок записи, колонка не будет отображаться.
5. При использовании ANSI следует использовать структуру параметров ksAttribute.

AttributeType – Структура параметров типа табличного атрибута

char	header [80]	заголовок-комментарий типа,
unsigned int	rowsCount	количество строк в таблице,
unsigned char	flagVisible	флаг видимости атрибута в таблице: (0 - невидимый, 1 - видимый),
char	password[10]	пароль, если не пустая строка - защищает от несанкционированного изменения типа,
reference	columns	массив неопределенной длины информации о столбцах.

Примечания:

1. Эта структура и использующая ее функция CreateAttrType устарели. Рекомендуется вместо них использовать структуру ksAttributeType и функцию ksCreateAttrType.
2. Массив columns содержит структуры параметров столбцов ColumnInfo.

ksAttributeType – Структура параметров типа атрибута

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksAttributeTypeParam.

unsigned int	key1	рекомендуется как код разработчика
unsigned int	key2	рекомендуется как код атрибута
unsigned int	key3	рекомендуется как код разработчика
unsigned int	key4	системный код атрибута
char	header[80]	заголовок-комментарий типа
unsigned int	rowsCount	количество строк в таблице
unsigned char	flagVisible	флаг видимости атрибута в таблице: (0 - видимый, 1 - невидимый)
char	password[10]	пароль, если не пустая строка - защищает от несанкционированного изменения типа
reference	columns	массив неопределенной длины информации о столбцах

Примечание:

1. Параметры типа атрибута могут быть получены с помощью функции ksGetAttrType.
2. При создании типа атрибута ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам CreateAttrIterator. При присвоении значений

ключам рекомендуется для ключей key1, key3 присваивать код, идентифицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ОАО "АСКОН".

3. Значения ключей атрибута могут быть получены с помощью функции GetAttrKeysInfo.
4. Массив колонок columns содержит структуры параметров колонок ColumnInfo и может быть создан с помощью функции CreateArray с параметром ATTR_COLUMN_ARR и AddArrayItem.
5. При использовании Unicode следует использовать структуру параметров ksAttributeTypeW.

ksAttributeTypeW – Структура параметров типа атрибута (Unicode)

Аналог данных параметров при использовании Automation - интерфейс ksAttributeTypeParam.

unsigned int	key1	рекомендуется как код разработчика,
unsigned int	key2	рекомендуется как код атрибута,
unsigned int	key3	рекомендуется как код разработчика,
unsigned int	key4	системный код атрибута,
wchar_t	header[80]	заголовок-комментарий типа,
unsigned int	rowsCount	количество строк в таблице,
unsigned char	flagVisible	флаг видимости атрибута в таблице: (0 - видимый, 1 - невидимый),
wchar_t	password[10]	пароль, если не пустая строка - защищает от несанкционированного изменения типа,
reference	columns	массив неопределенной длины информации о столбцах.

Примечание:

1. Параметры типа атрибута могут быть получены с помощью функции ksGetAttrType.
2. При создании типа атрибута ключам key1 - key4 могут быть присвоены нулевые значения. Если заданы ненулевые значения ключей или одного из ключей, эти значения могут быть использованы в дальнейшем как дополнительные идентификаторы для поиска атрибута с помощью итератора по атрибутам CreateAttrIterator. При присвоении значений ключам рекомендуется для ключей key1, key3 присваивать код, идентифицирующий разработчика, для ключа key2 - код атрибута, для ключа key4 - системный код. Значения параметра key4 от 0 до 1000 зарезервированы за ОАО "АСКОН".
3. Значения ключей атрибута могут быть получены с помощью функции GetAttrKeysInfo.
4. Массив колонок columns содержит структуры параметров колонок ColumnInfo и может быть создан с помощью функции CreateArray с параметром ATTR_COLUMN_ARR и AddArrayItem.
5. При использовании ANSI следует использовать структуру параметров ksAttributeType.

ColumnInfo – Структура параметров одного столбца табличного атрибута

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksColumnInfoParam.

char	header [80]	заголовок-комментарий столбца,
unsigned char	type	тип данных в столбце табличного атрибута,
unsigned char	key	дополнительный признак ("ключ"), который позволит отличить разные переменные с одинаковым типом,
char	def [TEXT_LENGTH]	значение в столбце по умолчанию,
unsigned char	flagEnum	флаг, включающий режим, когда значение поля атрибута будет заполнятся из массива перечисленных значений (1 - включен, 0 - отключен),
reference	fieldEnum	массив неопределенной длины перечислений (строки),
reference	columns	динамический массив неопределенной длины информации о колонках записи.

Примечание:

1. Если тип данных в столбце запись (RECORD), то массив columns, в свою очередь, содержит структуры, определяющие поля этой записи. Напомним, что запись соответствует табличному атрибуту (только запись, в отличие от табличного атрибута, не может содержать вложенную запись).
2. Параметры структуры могут быть получены с помощью функции GetAttrColumnInfo или поля “columns” структуры параметров типа атрибута AttributeType.
3. При использовании Unicode следует использовать структуру параметров ColumnInfoW.

ColumnInfoW – Структура параметров одного столбца табличного атрибута (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksColumnInfoParam.

wchar_t	header [80]	заголовок-комментарий столбца,
unsigned char	type	тип данных в столбце табличного атрибута,
unsigned char	key	дополнительный признак ("ключ"), который позволит отличить разные переменные с одинаковым типом,
wchar_t	def [TEXT_LENGTH]	значение в столбце по умолчанию,

unsigned char	flagEnum	флаг, включающий режим, когда значение поля атрибута будет заполнятся из массива перечисленных значений (1 - включен, 0 - отключен),
reference	fieldEnum	массив неопределенной длины перечислений (строки),
reference	columns	динамический массив неопределенной длины информации о колонках записи.

Примечание:

1. Если тип данных в столбце запись (RECORD), то массив columns, в свою очередь, содержит структуры, определяющие поля этой записи. Напомним, что запись соответствует табличному атрибуту (только запись, в отличие от табличного атрибута, не может содержать вложенную запись).
2. Параметры структуры могут быть получены с помощью функции GetAttrColumnInfo или поля “columns” структуры параметров типа атрибута AttributeType.
3. При использовании ANSI следует использовать структуру параметров ColumnInfo.

LibraryAttrTypeParam – Структура параметров для типа атрибута в библиотеке типов атрибутов

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibraryAttrTypeParam.

char	name [MAX_TEXT_LENGTH]	имя типа атрибута,
double	typeId	номер типа атрибута в библиотеке/

Примечание:

1. Параметры структуры могут быть получены с помощью метода ksGetLibraryAttrTypesArray.
2. Уникальный номер типа атрибута typeid может быть получен с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.
3. При использовании Unicode следует использовать структуру параметров LibraryAttrTypeParamW.

LibraryAttrTypeParamW – Структура параметров для типа атрибута в библиотеке типов атрибутов (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibraryAttrTypeParam.

wchar_t	name [MAX_TEXT_LENGTH]	имя типа атрибута,
double	typeId	номер типа атрибута в библиотеке/

Примечание:

1. Параметры структуры могут быть получены с помощью метода ksGetLibraryAttrTypesArray.
2. Уникальный номер типа атрибута typeid может быть получен с помощью функции GetAttrKeysInfo. Тип атрибута должен быть предварительно создан, например, с помощью функции ksCreateAttrType.
3. При использовании ANSI следует использовать структуру параметров LibraryAttrTypeParam.

Структуры параметров размеров

ABreakDimParam – Структура параметров углового размера с обрывом

Справка системы КОМПАС...

CM_CUT_DIMA.htmАналог данных параметров при использовании Automation - интерфейс ksABreakDimParam.

DimText	tPar	параметры размерной надписи,
BreakDimDrawing	dPar	параметры изображения размера,
ADimSource	sPar	параметры привязки углового размера.

Структура параметров размерной надписи DimText...

Структура параметров изображения размера с обрывом BreakDimDrawing...

Структура параметров привязки углового размера ADimSource...

ADimParam – Структура параметров углового размера

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksADimParam

DimText	tPar	параметры размерной надписи,
DimDrawing	dPar	параметры изображения размера,
ADimSource	sPar	параметры привязки углового размера.

Структура параметров размерной надписи DimText...

Структура параметров изображения размера DimDrawing...

Структура параметров привязки углового размера ADimSource...

ADimSource – Структура параметров привязки углового размера

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksADimSourceParam.

double	xc, yc	координаты центра размерной дуги,
double	rad	радиус размерной дуги,
double	ang1	начальный угол размерной дуги,
double	ang2	конечный угол размерной дуги,
int	dir	направление: (1 - размерная линия против часовой стрелки; -1 - по часовой стрелке),
double	x1, y1	координаты точки выхода первой выносной линии,
double	x2, y2	координаты точки выхода второй выносной линии.

Примечание:

Радиус размерной дуги - аннотационный параметр (он не зависит от масштаба).

BreakDimDrawing – Структура параметров отрисовки линейного или углового размера с обрывом

Справка системы КОМПАС...

CM_CUT_DIML.htmСправка системы КОМПАС...

CM_CUT_DIMA.htmАналог данных параметров при использовании Automation - интерфейс ksBreakDimDrawing.

unsigned char	pl	признак отрисовки выносной линии (0 - отрисовка включена, 1 - отрисовка выключена),
unsigned char	pt	Тип стрелки у первой выносной линии,
int	textPos	положение текста: 0 - автоматическое размещение текста, > 0 - значение расстояния от выносной линии до текста в направлении от первой точки ко второй),

int	shelfDir	признак отрисовки полки (0 - нет полки, -1 - полка направлена влево, 1 - полка направлена вправо, 2 - полка направлена вверх, 3 - полка направлена вниз).
double	ang	угол наклона "ножки" полки,
int	length	длина "ножки" полки.

Примечание:

Параметры textPos и length являются аннотационными (не зависят от масштаба, показывают расстояние и длину "на бумаге").

DimDrawing – Структура параметров отрисовки линейного и углового размеров

Справка системы КОМПАС: линейный размер...

CM_DIML.htm

Угловой размер...

CM_DIMA.htmАналог данных параметров при использовании Automation - интерфейс ksDimDrawingParam.

unsigned char	pl1	признак отрисовки первой выносной линии: (0 - включена, 1 - выключена),
unsigned char	pl2	признак отрисовки второй выносной линии: (0 - включена, 1 - выключена),
unsigned char	pt1	тип стрелки у первой выносной линии,
unsigned char	pt2	тип стрелки у второй выносной линии,
int	textPos	положение текста: (0 - автоматическое размещение текста, > 0 - на указанное расстояние в направлении от первой точки ко второй; < 0 - на указанное расстояние в направлении от второй точки к первой;
unsigned char	extBase	параметр отрисовки текста: (0 - в центре, 1- textPos относительно 1 точки; 2 - textPos относительно 2 точки; 3 - общая размерная линия),
int	shelfDir	наличие выносной полки: (0 - нет выносной полки, -1 - полка направлена влево 1 - полка направлена вправо, 2 - полка направлена вверх, 3 - полка направлена вниз),

double	ang	угол наклона "ножки" выносной полки,
int	length	длина "ножки" выносной полки.

Типы отрисовки стрелок в размерах...

Примечания:

1. Значение параметра textPos задается в миллиметрах для линейных размеров и в градусах для угловых. Этот параметр - аннотационный (не зависит от масштаба).
2. Параметр length - аннотационный (не зависит от масштаба).

DimText – Структура параметров размерной надписи

Справка системы КОМПАС...

CM_DIM_TEXT_EDITOR.htm#label_paramАналог данных параметров при использовании Automation - интерфейс ksDimTextParam.

unsigned short	style	стиль текста размера, (0 - стиль по умолчанию)
unsigned int	sign	номер условного значка перед номиналом (0 - нет значка, 1 - диаметр, 2 - квадрат, 3 - радиус, > 3 - номер значка из шрифта Symbol type A 4 - M - метрическая резьба 210 - символ сферы,
unsigned int	bitFlag	0 - ручное задание или набор битовых полей, задающих признаки размерной надписи,
reference	pText	динамический массив строк,
unsigned char	stringFlag	флаг используемого типа массива строк: 0 - динамический массив строк символов CHAR_STR_ARR или CHAR_STR_ARR_W, 1 - динамический массив строк текста TEXT_LINE_ARR

Признаки размерной надписи...

Структура параметров TextLineParam...

Примечание:

1. Строки в pText должны лежать в последовательности с учетом включения битовых флагов в bitFlag: текст перед номиналом, номинал, квалитет, верхнее отклонение, нижнее отклонение, единицы измерения, текст после размера, последующие строки.
2. Верхнее отклонение, нижнее отклонение принимаются, если флаг TOLERANCE не включен, а DEVIATION включен (это признак ручной простановки отклонений).
3. Для функции GetObjParam нужно определить флаг stringFlag. Если он равен нулю, выдается динамический массив строк символов.

4. Динамический массив TEXT_LINE_ARR содержит структуру TextLineParam.

DimensionsOptions – Структура параметров для определения настроек размеров

Справка системы КОМПАС...

CM_CHANGELEADER.htm

Аналог данных параметров при использовании Automation - интерфейс ksDimensionsOptions.

double	proLineExtension	выход выносных линий за размерную,
double	textDistanceFromDimLine	расстояние от размерной линии до текста,
double	textDistanceFromProLine	расстояние от выносных линий до текста,
double	dimLineExtension	выход размерной линии за текст,
double	arrowLength	длина стрелки размера,
unsigned short	style	стиль текста,
unsigned char	decimalsCount	количество знаков после запятой (от 0 до 9),
int	anglePrecisionLevel	точность углового размера: 0 - градусы, 1- минуты, 2- секунды,
int	hiddenToleranceNumber	максимальный номер показываемого квалитета (от 1 до 17) -1 - показ квалитета не включен.

DimensionPartsParam – Структура параметров объектов, составляющих размер

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksDimensionPartsParam.

reference	line1	первая выносная линия,
reference	line2	вторая выносная линия,
reference	dimLine	размерная линия,
reference	dimLine1	продолжение размерной линии,
reference	leg	"ножка",
reference	shelf	полка,
reference	gr	временная группа всех объектов размера, включая тексты,

reference	curveExt	- продолжение базовой кривой (у радиального размера дуги), - линия-"указатель", проведенная от размерной надписи к дуге (у размера дуги).

Примечание:

Данная структура содержит ссылки на все составляющие размера. Для их получения "рассыпается" копия размера в памяти. Размер в документе остается единым объектом.

LDimParam – Структура параметров линейного размера

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLDimParam.

DimText	tPar	параметры размерной надписи,
DimDrawing	dPar	параметры изображения размера,
LDimSource	sPar	параметры привязки линейного размера.

Структура параметров размерной надписи DimText...

Структура параметров изображения размера DimDrawing...

Структура параметров привязки линейного размера LDimSource...

LDimSource – Структура параметров привязки линейного размера

Справка системы КОМПАС...

191_Glava24_Linejnye_razmery.htmАналог данных параметров при использовании Automation - интерфейс ksLDimSourceParam.

unsigned char	ps	признак ориентации размерной линии:(0 - горизонтально, 1-вертикально, 2 - параллельно отрезку, соединяющему точки привязки 3 - по dx, dy, 4 - параллельно отрезку с выносными линиями по dx, dy.
double	x1, y1	координаты первой точки привязки,
double	x2, y2	координаты второй точки привязки,
double	dx, dy	вектор, определяющий положение размерной линии,
unsigned char	basePoint	признак, указывающий, от какой точки откладывать dx, dy:(1 - от первой точки, 2 - от второй точки).

Примечание:

Вектор, определяющий положение размерной линии, - аннотационный (не зависит от масштаба).

LBreakDimParam – Структура параметров линейного размера с обрывом

Справка системы КОМПАС...

CM_CUT_DIML.htmАналог данных параметров при использовании Automation - интерфейс ksLBreakDimParam.

DimText	tPar	параметры размерной надписи,
BreakDimDrawing	dPar	параметры изображения размера,
LBreakDimSource	sPar	параметры привязки линейного размера.

Структура параметров размерной надписи DimText...

Структура параметров изображения размера с обрывом BreakDimDrawing...

Структура параметров привязки линейного размера с обрывом LBreakDimSource...

LBreakDimSource – Структура параметров привязки линейного размера с обрывом

Справка системы КОМПАС...

CM_CUT_DIML.htmАналог данных параметров при использовании Automation - интерфейс ksLBreakDimSource.

double	x1, y1	координаты первой точки привязки,
double	x2, y2	координаты точки выхода стрелки,
double	x3, y3	координаты точки на размерной линии.

OrdinatedDimParam – Структура параметров размера высоты

Справка системы КОМПАС...

CM_ORDINATE_DIM.htmАналог данных параметров при использовании Automation - интерфейс ksOrdinatedDimParam.

DimText	tPar	размерная надпись,
OrdinatedDrawing	dPar	параметры изображения размера,
OrdinatedSource	sPar	параметры привязки размера.

Структура параметров изображения размера высоты OrdinatedDrawing...

Структура параметров привязки размера высоты OrdinatedSource...

OrdinatedDrawing – Структура параметров изображения размера высоты

Справка системы КОМПАС...

CM_ORDINATE_DIM.htmАналог данных параметров при использовании Automation - интерфейс ksOrdinatedDrawingParam.

unsigned char	type	тип размера высоты.

Типы размеров высоты...

OrdinatedSource – Структура параметров привязки размера высоты

Справка системы КОМПАС...

CM_ORDINATE_DIM.htmАналог данных параметров при использовании Automation - интерфейс ksOrdinatedSourceParam.

double	x0, y0	координаты точки, задающей нулевой уровень,
double	x1, y1	координаты точки, задающей измеряемый уровень,
double	x2, y2	координаты точки, задающей положение размерной надписи.

RBreakDimParam – Структура параметров радиального размера с изломом

Справка системы КОМПАС...

CM_DIMR_WITH_BREAK.htmАналог данных параметров при использовании Automation - интерфейс ksRBreakDimParam.

RDimSource	sPar	параметры привязки углового размера,
RBreakDrawing	dPar	параметры изображения размера,
DimText	tPar	параметры размерной надписи.

Структура параметров RDimSource...

Структура параметров RBreakDrawing...

Структура параметров DimText...

RBreakDrawing – Структура параметров изображения радиального размера с изломом

Справка системы КОМПАС...

CM_DIMR_WITH_BREAK.htmАналог данных параметров при использовании Automation - интерфейс ksRBreakDrawingParam.

unsigned char	pt	тип стрелки,
double	ang	угол наклона размерной линии,
unsigned int	pb	длина излома.

Типы отрисовки стрелок в размерах...

Примечание:

Длина излома - аннотационный параметр (он не зависит от масштаба).

RDimDrawing – Структура параметров отрисовки диаметрального и радиального размеров

Справка системы КОМПАС: диаметральный размер...

CM_DIMD.htm

Простой радиальный размер...

CM_DIMR.htmАналог данных параметров при использовании Automation - интерфейс ksRDimDrawingParam.

unsigned char	pt1	тип стрелки у первой выносной линии,
unsigned char	pt2	диаметральный размер: тип стрелки у второй выносной линии, радиальный размер: 0 - размер от центра, 1 - не от центра,
int	textPos	параметр отрисовки текста или длина ножки; аннотационный (не зависит от масштаба),
int	shelfDir	наличие горизонтальной выносной полки: (0 - нет выносной полки, -1 - полка направлена влево, 1 - полка направлена вправо, 2 - полка направлена вверх, 3 - полка направлена вниз),
double	ang	угол наклона размерной линии.

struct RDimDrawing {

Типы отрисовки стрелок в размерах...

RDimParam – Структура параметров диаметрального и обычного радиального размера

Справка системы КОМПАС: диаметральный размер...

CM_DIMD.htm

Простой радиальный размер...

CM_DIMR.htmАналог данных параметров при использовании Automation - интерфейс ksRDimParam.

DimText	tPar	параметры размерной надписи,
RDimDrawing	dPar	параметры изображения размера,
RDimSource	sPar	параметры привязки диаметрального и радиального размеров.

Структура параметров DimText...

Структура параметров RDimDrawing...

Структура параметров RDimSource...

RDimSource – Структура параметров привязки диаметрального и радиального размеров

Справка системы КОМПАС: диаметральный размер...

CM_DIMD.htm

Простой радиальный размер...

CM_DIMR.htmАналог данных параметров при использовании Automation - интерфейс ksRDimSourceParam.

double	xc, yc	координаты центра дуги или окружности,
double	rad	радиус дуги или окружности.

ShelfPar – Структура параметров выносной полки

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksShelfPar.

int	psh	направление полки параллельно оси X (0 - нет выносной полки, -1 - полка направлена влево, 1 - полка направлена вправо, 2 - полка направлена вверх, 3 - полка направлена вниз),
double	ang	угол наклона размерной линии для диаметрального и радиального размеров,
int	length	длина "ножки" выносной линии.

Структуры параметров документов

AssociationViewParam – Структура параметров ассоциативного вида

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksAssociationViewParam.

ViewParam	viewPar	- Параметры обычного вида, структура ViewParam
char	fileName [MAX_TEXT_LENGTH]	- Имя файла документа-модели
char	projectionName [TEXT_LENGTH]	- Имя проекции (из списка проекций в документе-источнике)
unsigned char	viewType	- Тип вида из LtViewType (параметр только для чтения)
unsigned char	dimensionLayoutScaling	- Признак масштабирования аннотационных объектов вида (только для SetObjParam)
unsigned char	projectionLink	- Проекционная связь
unsigned char	disassembly	- Разнести
long	visibleLinesStyle	- Номер стиля отрисовки видимых ребер и очерков, если 0 - умолчательный стиль
long	hiddenLinesStyle	- Номер стиля отрисовки всех невидимых линий, если 0 - умолчательный стиль
long	tangentEdgesStyle	- Номер стиля отрисовки видимых линий перехода, если 0 - умолчательный стиль
unsigned char	hiddenLinesShow	- Признак отрисовки невидимых линий
unsigned char	tangentEdgesShow	- Признак отрисовки видимых линий перехода
unsigned char	projBodies	- Признак проецирования тел
unsigned char	projSurfaces	- Признак проецирования поверхностей
unsigned char	projThreads	- Признак проецирования резьбы
unsigned char	reserve[30]	- Резерв
HatchParamEx	hatchPar	- Параметры штриховки, структура HatchParamEx (используется только в виде разрез\сечение)
unsigned char	sameHatch	- Одинаковая штриховка всех деталей сборки
unsigned char	section	- Признак разрез/сечение

Примечание:

1. Параметры могут быть получены с помощью метода GetObjParam с параметром ASSOCIATION_VIEW_PARAM.
2. При использовании Unicode следует использовать структуру параметров AssociationViewParamW.

AssociationViewParamW – Структура параметров ассоциативного вида (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksAssociationViewParam.

ViewParam	viewPar	- Параметры обычного вида, структура ViewParam
wchar_t	fileName [MAX_TEXT_LENGTH]	- Имя файла документа-модели
wchar_t	projectionName [TEXT_LENGTH]	- Имя проекции (из списка проекций в документе-источнике)
unsigned char	viewType	- Тип вида из LtViewType (параметр только для чтения)
unsigned char	dimensionLayoutScaling	- Признак масштабирования аннотационных объектов вида (только для SetObjParam)
unsigned char	projectionLink	- Проекционная связь
unsigned char	disassembly	- Разнести
long	visibleLinesStyle	- Номер стиля отрисовки видимых ребер и очерков, если 0 - умолчательный стиль
long	hiddenLinesStyle	- Номер стиля отрисовки всех невидимых линий, если 0 - умолчательный стиль
long	tangentEdgesStyle	- Номер стиля отрисовки видимых линий перехода, если 0 - умолчательный стиль
unsigned char	hiddenLinesShow	- Признак отрисовки невидимых линий
unsigned char	tangentEdgesShow	- Признак отрисовки видимых линий перехода
unsigned char	projBodies	- Признак проецирования тел
unsigned char	projSurfaces	- Признак проецирования поверхностей
unsigned char	projThreads	- Признак проецирования резьбы
unsigned char	reserve[30]	- Резерв
HatchParamEx	hatchPar	- Параметры штриховки, структура HatchParamEx (используется только в виде разрез\сечение)
unsigned char	sameHatch	- Одинаковая штриховка всех деталей сборки
unsigned char	section	- Признак разрез/сечение

Примечание:

1. Параметры могут быть получены с помощью метода GetObjParam с параметром ASSOCIATION_VIEW_PARAM.
2. При использовании ANSI следует использовать структуру параметров AssociationViewParam.

CopyObjectParam – Структура параметров копирования объекта графического документа

Справка системы КОМПАС...

CM_COPY_PROPERTY.htmАналог данных параметров при использовании Automation - интерфейс ksCopyObjectParam.

reference	p	указатель на объект, группу, вид, слой
double	xOld	координаты базовой точки объекта
double	yOld	
double	xNew	координаты точки вставки
double	yNew	
double	scale	масштаб
double	angle	угол поворота в градусах
unsigned char	attrCopy	признак копирования атрибутов
unsigned char	dimLineScale	признак масштабирования выносных линий
	spcObjCopy	копировать объекты спецификации
unsigned char	storagesCopy	копировать пользовательские данные и свойства
unsigned char	hyperLinksCopy	копировать ссылки

DocumentParam – Структура параметров документа

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksDocumentParam.

unsigned char	regim	режим (0 - видимый, 1 - "слепой"),
unsigned char	type	тип документа,
char	fileName [MAX_TEXT_LENGTH]	имя файла документа,
char	comment [TEXT_LENGTH]	комментарий к документу,
char	author [TEXT_LENGTH]	автор документа,
SheetPar	sheet	структура параметров оформления документов.

Примечание:

1. Структура параметров sheet используется для чертежей и спецификаций, т.е. когда параметр type имеет значения 1, 2, 4.
2. При использовании Unicode следует использовать структуру параметров DocumentParamW.

DocumentParamW – Структура параметров документа (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksDocumentParam.

unsigned char	regim	режим (0 - видимый, 1 - "слепой"),
unsigned char	type	тип документа,
wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла документа,
wchar_t	comment [TEXT_LENGTH]	комментарий к документу,
wchar_t	author [TEXT_LENGTH]	автор документа,
SheetParW	sheet	структура параметров оформления документов.

Примечание:

1. Структура параметров sheet используется для чертежей и спецификаций, т.е. когда параметр type имеет значения 1, 2, 4.
2. При использовании ANSI следует использовать структуру параметров DocumentParam.

LayerParam – Структура параметров слоя

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLayerParam.

unsigned short	state	состояние слоя,
unsigned long	color	цвет слоя в активном состоянии,
char	name [TEXT_LENGTH]	имя слоя.

Состояния видов и слоев...

Примечание.

При использовании Unicode следует использовать структуру параметров LayerParamW.

LayerParamW – Структура параметров слоя (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLayerParam.

unsigned short	state	состояние слоя,
unsigned long	color	цвет слоя в активном состоянии,
wchar_t	name [TEXT_LENGTH]	имя слоя.

Состояния видов и слоев...

Примечание.

При использовании ANSI следует использовать структуру параметров LayerParam.

OverlapObjectOptions – Параметры перекрывающихся объектов

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksOverlapObjectOptions.

unsigned short	overlap	1 - перекрывать штриховки и линии при пересечении, 0 - не перекрывать,
double	gap	Зазор при перекрывании объектов.

PlacementParam – Структура параметров местоположения (привязки)

Аналог данных параметров при использовании Automation - интерфейс ksPlacementParam.

double	xBase, yBase	координаты базовой точки в СК вида,
double	scale	масштаб,
double	ang	угол поворота в СК вида.

RasterFormatParam – Структура параметров записи в растровый формат

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRasterFormatParam.

unsigned char	format	формат растра
unsigned char	colorBPP	глубина цвета (бит на пиксел)
unsigned char	greyScale	признак сохранения в оттенках серого (0 - цветной растр, >0 - оттенки серого)
int	extResolution	разрешение растра (0 - текущее разрешение)
double	extScale	масштаб
unsigned char	colorType	цвет вывода объектов
unsigned char	onlyThinLine	признак вывода всех линий тонкими (0 - установленные для объектов толщины линий, >0 - выводит все тонкими линиями)
char	Pages [MAX_TEXT_LENGTH]	список диапазонов выводимых страниц в формате "beg1-end1, beg2-end2, beg3-end3, ..."
unsigned char	rangeIndex	признак выбора стороны страниц (0 - все, 1- нечетные, 2 - четные),

unsigned char	multiPageOutput	признак сохранения всех листов в одном файле (0 - сохранять листы в отдельных файлах, >0 сохранять все листы в одном файле) (признак используется только для формата TIFF)

Форматы растра...

Цвета вывода объектов...

Глубина цвета растра...

Примечание.

При использовании Unicode следует использовать структуру параметров RasterFormatParamW.

RasterFormatParamW – Структура параметров записи в растровый формат (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRasterFormatParam.

unsigned char	format	формат растра
unsigned char	colorBPP	глубина цвета (бит на пиксел)
unsigned char	greyScale	признак сохранения в оттенках серого (0 - цветной растр, >0 - оттенки серого)
int	extResolution	разрешение растра (0 - текущее разрешение)
double	extScale	масштаб
unsigned char	colorType	цвет вывода объектов
unsigned char	onlyThinLine	признак вывода всех линий тонкими (0 - установленные для объектов толщины линий, >0 - выводит все тонкими линиями)
wchar_t	Pages [MAX_TEXT_LENGTH]	список диапазонов выводимых страниц в формате "beg1-end1, beg2-end2, beg3-end3, ..."
unsigned char	rangeIndex	признак выбора стороны страниц (0 - все, 1- нечетные, 2 - четные),
unsigned char	multiPageOutput	признак сохранения всех листов в одном файле (0 - сохранять листы в отдельных файлах, >0 сохранять все листы в одном файле) (признак используется только для формата TIFF)

Форматы растра...

Цвета вывода объектов...

Глубина цвета растра...

Примечание.

При использовании ANSI следует использовать структуру параметров RasterFormatParam.

RasterParam – Структура параметров растрового объекта

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRasterParam.

PlacementParam	place	структура местоположения,
char	fileName [MAX_TEXT_LENGTH]	полный путь к файлу,
unsigned char	embeded	признак внедрения или ссылки (1 - внедрять данные в объект, 0 - связывать объект файлом на диске).

Примечание:

1. Значение параметра embeded = 0 в данный момент не используется (связь растрового объекта с файлом на диске в КОМПАС-ГРАФИК не реализована).
2. При использовании Unicode следует использовать структуру параметров RasterParamW.

RasterParamw – Структура параметров растрового объекта (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRasterParam.

PlacementParam	place	структура местоположения,
wchar_t	fileName [MAX_TEXT_LENGTH]	полный путь к файлу,
unsigned char	embeded	признак внедрения или ссылки (1 - внедрять данные в объект, 0 - связывать объект файлом на диске).

Примечание:

1. Значение параметра embeded = 0 в данный момент не используется (связь растрового объекта с файлом на диске в КОМПАС-ГРАФИК не реализована).
2. При использовании ANSI следует использовать структуру параметров RasterParam.

SheetOptions – Структура параметров оформления

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSheetOptions.

SheetPar	sheetPar	Структура параметров оформления SheetPar,
unsigned char	docType	Тип документа.

Примечание.

При использовании Unicode следует использовать структуру параметров SheetOptionsW.

SheetOptionsW – Структура параметров оформления (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSheetOptions.

SheetParW	sheetPar	Структура параметров оформления SheetParW,
unsigned char	docType	Тип документа.

Примечание.

При использовании ANSI следует использовать структуру параметров SheetOptions.

SheetPar – Структура параметров оформления

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSheetPar.

char	layoutName [MAX_TEXT_LENGTH]	для чертежа: - имя библиотеки оформления, для спецификации: - имя библиотеки стилей спецификации, пустая строка - библиотека Graphic.lyt
unsigned int	shtType	для чертежа - тип штампа из указанной библиотеки, для спецификации - номер стиля из указанной библиотеки
StandartSheet	stPar	- структура параметров листа чертежа стандартного формата,
SheetSize	usPar	- структура параметров листа чертежа нестандартного формата/

Примечание:

1. Один из параметров - stPar или usPar - используется только для чертежа. Стандартный чертеж или нестандартный, определяется значением параметра type из структуры параметров документа DocumentParam.
2. При использовании Unicode следует использовать структуру параметров SheetParW.

SheetParW – Структура параметров оформления (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSheetPar.

wchar_t	layoutName [MAX_TEXT_LENGTH]	для чертежа: - имя библиотеки оформления, для спецификации: - имя библиотеки стилей спецификации, пустая строка - библиотека Graphic.lyt,
unsigned int	shtType	для чертежа - тип штампа из указанной библиотеки, для спецификации - номер стиля из указанной библиотеки,
StandartSheet	stPar	- структура параметров листа чертежа стандартного формата,
SheetSize	usPar	- структура параметров листа чертежа нестандартного формата.

Примечание:

1. Один из параметров - stPar или usPar - используется только для чертежа. Стандартный чертеж или нестандартный, определяется значением параметра type из структуры параметров документа DocumentParamW.
2. При использовании ANSI следует использовать структуру параметров SheetPar.

SheetSize – Структура параметров нестандартного листа

Справка системы КОМПАС...

`362_41_1_Listy.htm`Аналог данных параметров при использовании Automation - интерфейс ksSheetSize.

double	width	длина листа,
double	height	высота листа.

SnapOptions – Структура параметров привязок в графическом документе

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSnapOptions.

struct	SnapOptions	Глобальные привязки
unsigned char	nearestPoint	Ближайшая точка
unsigned char	nearestMiddle	Середина
unsigned char	intersect	Пересечение
unsigned char	tangentToCurve	Касание
unsigned char	normalToCurve	Нормаль
unsigned char	grid	По сетке
unsigned char	xyAlign	Выравнивание
unsigned char	angSnap	Угловая привязка
unsigned char	pointOnCurve	Точка на кривой
unsigned int	commonOpt	Общие настройки привязок: динамически отслеживать, отображать текст, с учетом фоновых слоев, подавить привязки
double	angleStep	Угловой шаг для привязок
unsigned char	localSnap	Тип локальной привязки

Типы локальной привязки...

Общие настройки привязок...

StandartSheet – Структура параметров стандартного листа

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksStandartSheet

unsigned char	format	формат листа 0 (A0) ... 4(A4),
unsigned char	multiply	кратность формата,
unsigned char	direct	расположение штампа: - 0 - вдоль короткой стороны, - 1 - вдоль длинной.

Стандартные форматы листа...

ViewColorParam – Параметры цвета фона

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksViewColorParam.

long	color	Цвет фона, -1 если используется цвет окна, установленный в Windows,
unsigned char	useGradient	1 - использовать градиентный переход при полутоновом отображении 0 - не использовать (доступно только для документов деталей и сборок),
long	topColor	Верхний цвет перехода (доступно только для документов деталей и сборок),
long	bottomColor	Нижний цвет перехода (доступно только для документов деталей и сборок).

ViewParam – Структура параметров вида

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksViewParam.

unsigned short	state	состояние вида,
double	x,y	точка привязки вида,
double	scale	масштаб вида,
double	ang	угол поворота вида,
unsigned long	color	цвет вида в активном состоянии,
char	name [TEXT_LENGTH]	имя вида.

Состояния видов и слоев...

Примечание.

1. Параметры могут быть получены с помощью метода GetObjParam с параметром ALLPARAM.
2. При использовании Unicode следует использовать структуру параметров ViewParamW.

ViewParamW – Структура параметров вида (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksViewParam.

unsigned short	state	состояние вида,
double	x,y	точка привязки вида,
double	scale	масштаб вида,
double	ang	угол поворота вида,
unsigned long	color	цвет вида в активном состоянии
wchar_t	name [TEXT_LENGTH]	имя вида.

Состояния видов и слоев...

Примечание.

1. Параметры могут быть получены с помощью метода GetObjParam с параметром ALLPARAM.
2. При использовании ANSI следует использовать структуру параметров ViewParam.

Структуры параметров графических объектов

ArcParam – Структура параметров дуги окружности по центру, радиусу и углам

Справка системы КОМПАС...

1436_Postroenie_dug.htm

Аналог данных параметров при использовании Automation - интерфейс ksArcByAngleParam.

double	xc,yc	координаты центра дуги,
double	rad	радиус дуги,
double	ang1	начальный угол,
double	ang2	конечный угол,
short	dir	направление построения дуги,
unsigned short	style	стиль линии.

Системные стили линий...

ArcParam1 – Структура параметров дуги по точкам

Справка системы КОМПАС...

1436_Postroenie_dug.htm

Аналог данных параметров при использовании Automation - интерфейс ksArcByPointParam.

double	xc,yc	координаты центра дуги,
double	rad	радиус дуги,
double	x1,y1	начальная точка дуги,
double	x2,y2	конечная точка дуги,
short	dir	направление построения дуги,
unsigned short	style	стиль линии.

Системные стили линий...

AxisLineParam – Структура параметров осевой линии

Аналог данных параметров при использовании Automation - интерфейс ksAxisLineParam.

MathPointParam	begPoint	координаты начальной точки осевой линии,
MathPointParam	endPoint	координаты конечной точки осевой линии.

BezierParam – Структура параметров кривой Безье

Справка системы КОМПАС...

spline_postroenie.htm#BEZIERАналог данных параметров при использовании Automation - интерфейс ksBezierParam.

reference	pMathPoint	массив неопределенной длины математических точек сплайна,
unsigned char	closed	признак замкнутости кривой (0 - не замкнута 1 - замкнута),
unsigned short	style	стиль линии.

Системные стили линий...

BezierPointParam – Структура параметров узла кривой Безье

Справка системы КОМПАС...

spline_postroenie.htm#BEZIERАналог данных параметров при использовании Automation - интерфейс ksBezierPointParam.

double	x, y	координаты базовой точки,
double	ang	угол наклона касательной к кривой в базовой точке,
double	left	расстояние от базовой точки к левой точке узла,
double	right	расстояние от базовой точки к правой точке узла.

CentreParam – Структура параметров обозначения центра

Справка системы КОМПАС...

CM_CENTRE_MARKER.htmАналог данных параметров при использовании Automation - интерфейс ksCentreParam.

reference	baseCurve	- указатель на базовую геометрическую кривую (например, окружность) или 0, если создается отдельное обозначение центра,
double	x, y	- координаты точки привязки, (если задана базовая кривая, эти параметры не используются),
double	ang	- угол наклона обозначения центра (если задана базовая кривая, этот параметр не используется),

unsigned char	type	- тип обозначения центра: 0 - маленький крестик, 1 - одна ось, 2 - две оси,
unsigned char unsigned char unsigned char unsigned char	standXpTail standXmTail standYpTail, standYmTai	- признаки длины для каждого "хвостика" обозначения центра (0 – длина "хвостика" задается 1 - "хвостик" стандартный),
double double double double	lenXpTail lenXmTail lenYpTail lenYmTail	- длина для каждого "хвостика", если он нестандартный.

Примечание:

1. Индексация полуосей обозначения центра следующая:

- Xp- по оси абсцисс системы координат обозначения центра,
- Xm- против оси абсцисс системы координат обозначения центра,
- Yp- по оси ординат системы координат обозначения центра,
- Ym- против оси ординат системы координат обозначения центра.

2. Система координат обозначения центра – система, оси которой совпадают с осями обозначения, а сама она повернута относительно текущей системы координат на угол ang.

CircleParam – Структура параметров окружности

Справка системы КОМПАС...

1432_Okruznosti.htm#okr_po_centru_i_tochke
Аналог данных параметров при использовании Automation - интерфейс ksCircleParam.

double	xc,yc	координаты центра окружности,
double	rad	радиус окружности,
unsigned short	style	стиль линии.

Системные стили линий...

CON – Структура параметров сопряжения двух кривых окружностью

Справка системы КОМПАС...

1432_Okruznosti.htm#okr_kasateln_k_dvum_krivumАналог данных параметров при использовании Automation - интерфейс ksCON.

double	xc, yc	координаты центра сопрягающей окружности,
double	x1, y1	координаты первой точки сопряжения,
double	x2, y2	координаты второй точки сопряжения.

ConicArcParam – Структура параметров конического сечения

Аналог данных параметров при использовании Automation - интерфейс ksConicArcParam.

double	A, B, C, D, E, F	коэффициенты канонического уравнения,
double	x1, y1	координаты начальной точки,
double	x2, y2	координаты конечной точки,
double	style	стиль линии.

Системные стили линий...

CornerParam – Структура параметров скругленных (или усеченных) углов прямоугольников и правильных многоугольников

Справка системы КОМПАС: скругление...

173_22_3_Skruglenie.htmСправка системы КОМПАС:фаска...

172_22_1_Faska.htm

Аналог данных параметров при использовании Automation - интерфейс ksCornerParam.

int	index	индекс (номер) угла (0, 1, 2, ...),
unsigned char	fillet	признак фаски/скругления: 0 - фаска, 1 -скругление,
double	l1	длина фаски первого сегмента или радиус скругления,
double	l2	длина фаски второго сегмента.

Примечание:

Данная структура используется внутри структуры параметров прямоугольника RectangleParam или правильного многоугольника RegularPolygonParam для задания скруглений и фасок на определенных углах этого многоугольника. Если скругления и фаски не нужны, то структура CornerParam не используется.

Пример работы со структурой CornerParam и массивом CORNER_ARR

CurvePattern – Структура параметров участка штриховой кривой

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurvePattern.

double double	visibleSeg invisibleSeg	длина видимого участка, длина невидимого участка.

CurvePatternEx – Структура параметров участка штриховой кривой (расширенная)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurvePatternEx.

double double double	visibleSeg invisibleSeg dx, dy	длина видимого участка, длина невидимого участка, положение базовой точки ("нуля")
unsigned char	pictureType	картинки относительно начала образца, тип картинки (0 - картинка в виде массивов ломаных линий, 1 - картинка в виде фрагмента),
CurvePicture	picture	структура параметров картинки, состоящей из ломаных линий,
char	frwName[TEXT_LENGTH]	имя фрагмента-источника картинки.

Примечания:

1. Какой из параметров - picture или frwName используется, определяется значением параметра pictureType.
2. Картинка в виде фрагмента используется только в функции AddStyle (при создании стиля).
3. При использовании Unicode следует использовать структуру параметров CurvePatternExW.

CurvePatternExW – Структура параметров участка штриховой кривой (расширенная), Unicode

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurvePatternEx.

double double double	visibleSeg invisibleSeg dx, dy	длина видимого участка, длина невидимого участка, положение базовой точки ("нуля") картинки относительно начала образца,

unsigned char	pictureType	тип картинки (0 - картинка в виде массивов ломаных линий, 1 - картинка в виде фрагмента),
CurvePicture	picture	структура параметров картинки, состоящей из ломаных линий,
wchar_t	frwName[TEXT_LENGTH]	имя фрагмента-источника картинки.

Примечания:

1. Какой из параметров - picture или frwName используется, определяется значением параметра pictureType.
2. Картинка в виде фрагмента используется только в функции AddStyle (при создании стиля).
3. При использовании ANSI следует использовать структуру параметров CurvePatternEx.

CurvePicture – Структура параметров "картинки", включаемой в стиль линии

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurvePicture.

reference	polygon	динамический массив POLYLINE_ARR ломаных линий, описывающий картинку,
reference	fill	динамический массив POLYLINE_ARR точек, описывающий границу заливки.

Примечание:

Координаты точек в массивах POLYLINE_ARR и POLYLINE_ARR задаются в масштабе листа относительно нулевой точки картинки.

CurveStyleParam – Структура параметров стиля кривой

Пример...

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurveStyleParam.

char	name[TEXT_LENGTH]	имя стиля,
unsigned long	color	цвет,
double	paperWidth	толщина линии на бумаге ("толщина пера"),
unsigned char	screenWidth	толщина линии на экране (в пикселах),

unsigned char	curveType	набор полей, определяющих тип кривой (0 - сплошная, 1 - прерывистая 2 - прерывистая, содержащая картинки),
reference	pattern	для прерывистой кривой - динамический массив параметров участков штриховой кривой,
unsigned char	even	для прерывистой кривой: 1 - кривая всегда оканчивается штрихами 0 - кривая оканчивается "как получится".

Структура параметров CurvePattern...

Структура параметров CurvePatternEx...

Параметры пера пользовательского стиля линии...

Примечания:

1. В качестве параметра pattern для прерывистой кривой используется динамический массив CURVE_STYLE, содержащий структуры параметров участков CurvePattern, а для прерывистой кривой, содержащей картинки, - динамический массив CURVE_STYLE_EX со структурами CurvePatternEx.
2. Параметр curveType представляет собой набор полей, задающих тип кривой и толщину пера при отрисовке этой кривой.
3. Например, если curveType = 1|LIKE_BASIC_LINE, то линия прерывистая с параметрами пера, как у системной основной линии.

   При использовании Unicode следует использовать структуру параметров CurveStyleParamW .

CurveStyleParamW – Структура параметров стиля кривой (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksCurveStyleParam.

wchar_t	name[TEXT_LENGTH]	имя стиля,
unsigned long	color	цвет,
double	paperWidth	толщина линии на бумаге ("толщина пера"),
unsigned char	screenWidth	толщина линии на экране (в пикселах),

unsigned char	curveType	набор полей, определяющих тип кривой (0 - сплошная, 1 - прерывистая 2 - преры вистая, содержащая картинки),
reference	pattern	для прерывистой кривой - динамический массив параметров участков штриховой кривой,
unsigned char	even	для прерывистой кривой: 1 - кривая всегда оканчивается штрихами 0 - кривая оканчивается "как получится".

Структура параметров CurvePattern...

Структура параметров CurvePatternExW...

Параметры пера пользовательского стиля линии...

Примечания:

1. В качестве параметра pattern для прерывистой кривой используется динамический массив CURVE_STYLE, содержащий структуры параметров участков CurvePattern, а для прерывистой кривой, содержащей картинки - динамический массив CURVE_STYLE_EX со структурами CurvePatternEx.
2. Параметр curveType представляет собой набор полей, задающих тип кривой и толщину пера при отрисовке этой кривой.

   Например, если curveType = 1|LIKE_BASIC_LINE, то линия прерывистая с параметрами пера как у системной основной линии.
3. При использовании ANSI следует использовать структуру параметров CurveStyleParam.

EquidistantParam – Структура параметров эквидистанты

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksEquidistantParam.

reference	geoObj	графический объект - базовая кривая эквидистанты,
unsigned char	side	указание, с какой стороны строить эквидистанту (0 - слева по направлению, 1 - справа по направлению, 2 - с двух сторон),
unsigned char	cutMode	тип обхода углов контура (0 - обход срезом, 1 - обход дугой),

unsigned char	degState	флаг построения вырожденных сегментов эквидистанты (0 - вырожденные сегменты запрещены, 1 - вырожденные сегменты разрешены),
double	radRight	радиус эквидистанты справа по направлению кривой,
double	adLeft	радиус эквидистанты слева по направлению кривой,
unsigned short	style	стиль линии.

Системные стили линий...

EllipseArcParam – Структура параметров дуги эллипса

Аналог данных параметров при использовании Automation - интерфейс ksEllipseArcParam.

double	xc;yc	координаты центра эллипса,
double	a, b	длина полуосей эллипса,
double	ang	угол наклона оси a эллипса к оси X,
double	angFirst	начальный угол дуги к оси a,
double	angSecond	конечный угол дуги к оси a,
short	dir	направление построения,
unsigned short	style	стиль линии.

Системные стили линий...

EllipseArcParam1 – Структура параметров дуги эллипса (при параметрическом построении)

Аналог данных параметров при использовании Automation - интерфейс ksEllipseArcParam1.

double	xc, yc	координаты центра эллипса,
double	a, b	длина полуосей эллипса,
double	ang	угол наклона полуоси a к оси OX,
double	parFirst	начальное значение параметра,
double	parSecond	конечное значение параметра,
short	dir	направление построения дуги,
unsigned short	style	стиль линии.

Системные стили линий...

EllipseParam – Структура параметров эллипса

Справка системы КОМПАС...

139_Glava14_Ehllipsy.htm

Аналог данных параметров при использовании Automation - интерфейс ksEllipseParam.

double	xc,yc	координаты центра эллипса,
double	a, b	длина полуосей эллипса,
double	ang	угол наклона оси a эллипса к оси X,
unsigned short	style	стиль линии.

Системные стили линий...

HatchLineParam – Структура параметров линии штриховки

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksHatchLineParam.

double	x, y	точка привязки линии,
double	dx, dy	смещение для следующей линии; dy должен быть отличен от нуля!
double	ang	угол наклона линии,
unsigned char	typeCurvStyle	признак стиля линии (0 - системный 1 - пользовательский),
unsigned short	style	системный стиль линии,
CurveStyleParam	curPar	параметры пользовательского стиля линии.

Системные стили линий...

Примечание.

При использовании Unicode следует использовать структуру параметров HatchLineParamW.

HatchLineParamW – Структура параметров линии штриховки (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksHatchLineParam.

double	x, y	точка привязки линии,
double	dx, dy	смещение для следующей линии; dy должен быть отличен от нуля!
double	ang	угол наклона линии,
unsigned char	typeCurvStyle	признак стиля линии (0 - системный 1 - пользовательский),

unsigned short CurveStyleParamW	style curPar	системный стиль линии, структура параметров пользовательского стиля линии.

Системные стили линий...

Примечание.

При использовании ANSI следует использовать структуру параметров HatchLineParam.

HatchParam – Структура параметров штриховки

Справка системы КОМПАС...

CM_HATCH.htmАналог данных параметров при использовании Automation - интерфейс ksHatchParam.

unsigned short	style	- стиль штриховки,
double	ang	- угол штриховки,
double	step	- шаг штриховки,
double	width	- ширина полосы штриховки (0 - штриховать всю область),
double	x, y	- базовая точка (нужна только для отрисовки штриховки),
reference	pBoundaries	- временная группа - границы штриховки.

Системные стили штриховок...

Примечание:

Параметр pBoundaries используется для функций GetObjParam и ksHatch.

HatchParamEx – Расширенная структура параметров штриховки

Справка системы КОМПАС...

CM_HATCH.htmАналог данных параметров при использовании Automation - интерфейс ksHatchParam.

unsigned short	style	- стиль штриховки,
double	ang	- угол штриховки,
double	step	- шаг штриховки,
double	width	- ширина полосы штриховки (0 - штриховать всю область),
double	x, y	- координаты базовой точки (нужна только для отрисовки штриховки),
reference	pBoundaries	- временная группа - границы штриховки,
unsigned long	color	Цвет, по умолчанию FREE_COLOR (0xff000000l),
unsigned char	sheetAng	0 - угол собственный (накатка); 1 - угол листовой (обычная штриховка).

Системные стили штриховок...

Примечание:

Параметр pBoundaries используется для функций GetObjParam и ksHatch.

HatchStyleParam – Структура параметров стиля штриховки

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksHatchStyleParam.

char	name [TEXT_LENGTH]	имя стиля штриховки,
double	step	шаг штриховки по умолчанию,
double	ang	угол наклона,
MathPointParam	refPoint	базовая точка штриховки,
double	width	ширина полосы штриховки,
unsigned long	color	цвет штриховки (по умолчанию FREE_COLOR (0xff000000l)),
unsigned char	mayChangeAngle	признак, показывающий, разрешено ли менять угол наклона штриховки,
unsigned char	mayChangeWidth	признак, показывающий, разрешено ли менять ширину полосы штриховки,
unsigned char	mayChangeSpace	признак, показывающий, разрешено ли менять шаг (масштаб) штриховки,
unsigned char	isScalable	признак изменения расстояния между линиями штриховки (1 - по масштабу, 0 - по шагу),
reference	arrLineParam	массив структур параметров линий, участвующих в штриховке HatchLineParam.

Структура параметров MathPointParam...

Примечание.

При использовании Unicode следует использовать структуру параметров HatchStyleParamW.

HatchStyleParamW – Структура параметров стиля штриховки (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksHatchStyleParam.

wchar_t	name [TEXT_LENGTH]	имя стиля штриховки,
double	step	шаг штриховки по умолчанию,
double	ang	угол наклона,
MathPointParam	refPoint	базовая точка штриховки,
double	width	ширина полосы штриховки,

unsigned long	color	цвет штриховки (по умолчанию FREE_COLOR (0xff000000l)),
unsigned char	mayChangeAngle	признак, показывающий, разрешено ли менять угол наклона штриховки,
unsigned char	mayChangeWidth	признак, показывающий, разрешено ли менять ширину полосы штриховки,
unsigned char	mayChangeSpace	признак, показывающий, разрешено ли менять шаг (масштаб) штриховки,
unsigned char	isScalable	признак изменения расстояния между линиями штриховки (1 - по масштабу, 0 - по шагу),
reference	arrLineParam	массив структур параметров линий, участвующих в штриховке HatchLineParam .

Структура параметров MathPointParam...

Примечание.

При использовании ANSI следует использовать структуру параметров HatchStyleParam.

HotPointDescription – Структура параметров характерной точки

Справка системы КОМПАС...

double	x, y	координаты точки в СК объекта,
char*	text	текст, расположенный рядом с точкой,
int	cursorId	идентификатор стандартного курсора,
HINSTANCE	cursorInst	описание курсора при прохождении над точкой.

Примечание:

Текст text должен быть статическим, хотя бы на время работы с библиотечным элементом.

HotPointDescription1 – Структура параметров характерной точки

Справка системы КОМПАС...

LPOLESTR	text	текстовая строка, расположенная около hot-точки для UNICODE вместо HotPointDescription::text,
Описание курсора при прохождении над точкой		
int	bitmapId	идентификатор bmp из ресурсов,
HINSTANCE	bitmapInst	Модуль,

int	bitmapCO	система координат, в которой отрисовывается битмап: - 0 - СК листа, - 1 - СК вида, - 2 - СК владельца,
int	bitmapId	идентификатор или тип битмапа характерной точки,
int	bitmapIdMove	идентификатор битмапа характерной точки при прохождении над точкой курсора,
int	bitmapIdSelect	идентификатор битмапа характерной точки при ее селектировании,
double	hotPointAngle	угол смещения для отображения горячих точек,
double	hotPointOffset	величина смещения горячих точек.
int	enableRotate	0 - запретить поворот хот-точки, 1 - разрешить поворот хот-точки, -1 по умолчанию (поворот разрешен только для ksHPTriangleDisplaced)
LPOLESTR	bitmapFont	Шрифт для шрифтовых хот-точек. Шрифт хот-точек может быть расположен в ресурсах библиотеки. Библиотека должна зарегистрировать его в системе. Пример инсталляции шрифта хот точек...
COLORREF	fontSymbolColor	Цвет для шрифтовых хот-точек
double	fontSymbolScale	Масштабирование для шрифтовых хот-точек Параметр влияет на размер хот-точки. По умолчанию fontSymbolScale == 1.0, при этом используется высота шрифта 3.
int	hotPointOffsetType	1- величина смещения задана в абсолютных единицай, 0- в условных. Смещение влияет на положение отрисовки хот-точки относительно координат хот-точки. Смещение выполняется по направлению угла поворота хот-точки. При сдвиге хот-точки приходят координаты без учета смещения хот-точки. Это дает воз можность в одном положении задать две или более хот-точек с разным назначением, как в дуге.
COLORREF	fontSymbolMoveColor	Цвет для шрифтовых хот-точек при перемещении
COLORREF	fontSymbolSelectColor	Цвет для шрифтовых хот-точек при селектировании

Для иконки ksHPSmall в Компас используется коэффициент 0.75.

InsertFragmentParam – Структура параметров вставки фрагмента

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksInsertFragmentParam.

PlacementParam	place	положение вставки фрагмента (PlacementParam),
unsigned char	insertType	тип вставки фрагмента (0 - взять в документ, 1 - внешней ссылкой, 3 - локальный фрагмент),
unsigned char	multiLayer	признак размещения объектов фрагмента по слоям (0 - объекты на одном слое, 1 - объекты на разных слоях,
char	fileName [MAX_TEXT_LENGTH]	имя файла фрагмента или "\0" для локального фрагмента,
char	comment [MAX_TEXT_LENGTH	имя вставки.

Структура параметров положения вставки PlacementParam...

Примечание.

При использовании Unicode следует использовать структуру параметров InsertFragmentParamW.

InsertFragmentParamW – Структура параметров вставки фрагмента (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksInsertFragmentParam.

PlacementParam	place	положение вставки фрагмента (PlacementParam),
unsigned char	insertType	тип вставки фрагмента (0 - взять в документ, 1 - внешней ссылкой, 3 - локальный фрагмент)
unsigned char	multiLayer	признак размещения объектов фрагмента по слоям (0 - объекты на одном слое, 1 - объекты на разных слоях
wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла фрагмента или "\0" для локального фрагмента
wchar_t	comment [MAX_TEXT_LENGTH	имя вставки

Структура параметров положения вставки PlacementParam...

Примечание.

При использовании ANSI следует использовать структуру параметров InsertFragmentParam.

LineParam – Структура параметров вспомогательной прямой

Справка системы КОМПАС...

1396_vspomogatelnye.htm#simple_line

Аналог данных параметров при использовании Automation - интерфейс ksLineParam.

double	x, y	координаты точки на прямой,
double	ang	угол наклона прямой к оси X.

LineSegParam – Структура параметров отрезка

Справка системы КОМПАС...

1418_Postroenie_otrezkov_.htm#otr_dve_tochki

Аналог данных параметров при использовании Automation - интерфейс ksLineSegParam.

double	x1,y1	координаты начальной точки,
double	x2,y2	координаты конечной точки,
unsigned short	style	стиль линии.

Системные стили линий...

MathPointParam – Структура параметров математической точки

Справка системы КОМПАС...

Prjamougolqnik.htm

double	x, y	координаты точки.

NurbsParam – Структура параметров кривой NURBS

Справка системы КОМПАС...

149_17_1_Lomanaja.htmАналог данных параметров при использовании Automation - интерфейс ksNurbsParam.

unsigned char	degree	порядок NURBS (степень полинома + 1), от 3 до 10,
unsigned char	close	признак замыкания сплайна (0 - незамкнутый, 1 - замкнутый),
unsigned short	style	стиль линии,
reference	pPoint	динамический массив точек сплайна (NurbsPointParam),
reference	pKnot	динамический массив узлов сплайна,
unsigned char	periodic	признак периодичности сплайна (0 - сплайн непериодический, 1 - сплайн периодический).

Системные стили линий...

Структура параметров точки NURBS (NurbsPointParam)...

Примечание:

1. Параметр periodic используется только для функции GetObjParam.
2. Если для описания NURBS выбран вариант ALLPARAM или SHEET_ALLPARAM, то способ заполнения массивов pPoint и kPoint определяется значением параметра Close:

- Если Close=0 (NURBS замкнут) - параметры с разжатым узловым вектором,
- Если Close=1 (NURBS разомкнут) - параметры с зажатым узловым вектором.

Если для описания NURBS выбран вариант NURBS_CLAMPED_PARAM или NURBS_CLAMPED_SHEETPARAM, то, вне зависимости от значения параметра Close, возвращаются параметры с зажатым узловым вектором.

NurbsPointParam – Структура параметров точки NURBS

Справка системы КОМПАС...

914_3_2_9_Tablica_Koordinati_vershin.htm#table_pointes_splineАнало г данных параметров при использовании Automation - интерфейс ksNurbsPointParam.

double	x, y	координаты базовой точки,
double	weight	вес точки (должен быть больше нуля).

Phantom – Структура параметров фантома

Аналог данных параметров при использовании Automation - интерфейс ksPhantom.

unsigned short	phType	тип фантома
struct Type1	type1	параметры сдвига группы (phType=1)
struct Type2	type2	параметры фантома-отрезка (phType=2)

struct Type3	type3	параметры фантома-прямоугольника (phType=3)
struct Type3	type4	параметры фантома-отрезка с заданным углом (phType=4)
struct Type5	type5	параметры фантома-половины прямоугольника (phType=5)
struct Type6	type6	параметры пользовательского фантома (phType=6)
struct Type2	type7	параметры фантома-окружности (phType=7)

Типы фантомов...

Структура параметров Type1...

Структура параметров Type2...

Структура параметров Type3...

Структура параметров Type5...

Структура параметров Type6..

Примечания:

1. Какая из структур параметров фантома - Type1, Type2, Type3, Type5 или Type6 используется, зависит от значения параметра phType.
2. Фантом-отрезок и фантом-окружность используют одну структуру параметров - Type2.
3. Фантом-прямоугольник и фантом-отрезок с заданным углом используют одну структуру параметров - Type3.

PointParam – Структура параметров точки (графического объекта)

Справка системы КОМПАС...

1391_Postroenie_tochek.htm#simple_point

Аналог данных параметров при использовании Automation - интерфейс ksPointParam.

double unsigned short	x, y style	координаты точки, стиль отрисовки точки.

Системные стили отрисовки точек

PolylineParam –Структура параметров ломаной линии

Справка системы КОМПАС...

spline_postroenie.htm#POLYLINEАналог данных параметров при использовании Automation - интерфейс ksPolylineParam.

reference	pMathPoint	динамический массив математических точек POINT_ARR,
unsigned short	style	стиль линии.

Системные стили линий...

PolylineParamEx – Структура параметров ломаной линии (расширенная)

Справка системы КОМПАС...

spline_postroenie.htm#POLYLINEАналог данных параметров при использовании Automation - интерфейс ksPolylineParam.

reference	pMathPoint	динамический массив математических точек POINT_ARR,
unsigned short	style	стиль линии,
unsigned char	closed	признак замкнутой кривой (1 - ломаная замкнута, 0 - ломаная разомкнута).

Системные стили линий...

RectParam – Структура параметров прямоугольника по диагональным точкам

Справка системы КОМПАС...

Prjamougolqnik.htm#rectangulerАналог данных параметров при использовании Automation - интерфейс ksRectParam.

MathPointParam	pBot	параметры левой нижней точки прямоугольника,
MathPointParam	pTop	параметры правой верхней точки прямоугольника.

Структура параметров MathPointParam...

RectangleParam – Структура параметров прямоугольника

Справка системы КОМПАС: команда Прямоугольник по центру и вершине...

PrjamougolqnikCENTER_VERTEX

Справка системы КОМПАС: команда Прямоугольник...

PrjamougolqnikrectangulerАналог данных параметров при использовании Automation - интерфейс ksRecangletParam.

double	x, y	координаты базовой точки прямоугольника - одной из его вершин,
double	ang	угол наклона стороны прямоугольника, выходящей из базовой точки,
double	height	высота прямоугольника,
double	wight	"ширина" прямоугольника - длина стороны, характеризующейся углом наклона ang,
unsigned short	style	стиль линии,
reference	pCorner	динамический массив CORNER_ARR структур параметров скругленных (или усеченных) углов CornerParam.

Системные стили линий...

Структура параметров CornerParam...

RegularPolygonParam – Структура параметров правильного многоугольника

Справка системы КОМПАС...

`Mnogougolqnik.htm`

Аналог данных параметров при использовании Automation - интерфейс ksRegularPolygonParam

int	count	количество вершин многоугольника,
double	xc, yc	центр вписанной или описанной окружности,
double	ang	угол радиус-вектора, направленного от центра к первой вершине,
double	radius	радиус вписанной или описанной окружности,
unsigned char	describe	признак описанного или вписанного многоугольника, (0 - вписанный многоугольник, 1 - описанный многоугольник),
unsigned short	style	стиль линии,
reference	pCorner	динамический массив CORNER_ARR структур параметров скругленных (или усеченных) углов CornerParam.

Системные стили линий...

Структура параметров CornerParam..

Type1– Структура параметров сдвига группы

Аналог данных параметров при использовании Automation - интерфейс ksType1.

double	xBase, yBase	координаты начальной точки группы,
double	ang	угол поворота группы,
double	scale	масштаб,

reference	gr	указатель на группу.

Type2 – Структура параметров фантома-отрезка или фантома-окружности

Аналог данных параметров при использовании Automation - интерфейс ksType2.

double	xBase, yBase	координаты начальной точки отрезка или центра окружности.

Type3 – Структура параметров фантома-отрезка с заданным углом и фантома-прямоугольника

Аналог данных параметров при использовании Automation - интерфейс ksType3.

double	xBase, yBase	координаты начала отрезка или угла прямоугольника,
double	ang	угол наклона отрезка или диагонали прямоугольника.

Type5 – Структура параметров фантома-половины прямоугольника с заданным углом диагонали

Аналог данных параметров при использовании Automation - интерфейс ksType5.

double	xBase, yBase	координаты угла прямоугольника,
double	ang	угол наклона диагонали прямоугольника,
unsigned char	horizon	признак, с какой стороны подходить к курсору (1 - по горизонтали, 0 - по вертикали).

Type6 – Структура параметров пользовательского фантома

Аналог данных параметров при использовании Automation - интерфейс ksType6.

reference	gr	указатель на временную группу объектов, которая отображается в виде фантома.

Примечание:

В этой структуре, в отличие от структуры Type1, не устанавливаются параметры поворота и масштабирования группы.

TAN – Структура параметров прямой, касательной к двум кривым

Справка системы КОМПАС...

1396_vspomogatelnye.htm#TANGENT2_LINEАналог данных параметров при использовании Automation - интерфейс ksTAN.

double	x1, y1	координаты первой точки касания,
double	x2, y2	координаты второй точки касания.

Структуры параметров объектов оформления чертежа

BaseParam – Структура параметров обозначения базы

Справка системы КОМПАС...

CM_BASE.htmАналог данных параметров при использовании Automation - интерфейс ksBaseParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию),
double	x1, y1	координаты базовой точки (начало "опоры"),
double	x2, y2	координаты конечной точки "опоры"
unsigned char	type	способ задания надписи в обозначении базы (0 - текст в виде строки, 1 - динамический массив компонентов текста),
char	str [50]	текст в обозначении базы,
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR.

Примечание:

1. Какой из параметров - str или pTextItem используется, определяется значением параметра type.
2. При использовании Unicode следует использовать структуру параметров BaseParamW.

BaseParamW – Структура параметров обозначения базы (Unicode)

Справка системы КОМПАС...

CM_BASE.htmАналог данных параметров при использовании Automation - интерфейс ksBaseParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию),

double	x1, y1	координаты базовой точки (начало "опоры"),
double	x2, y2	координаты конечной точки "опоры",
unsigned char	type	способ задания надписи в обозначении базы (0 - текст в виде строки, 1 - динамический массив компонентов текста),
wchar_t	str [50]	текст в обозначении базы,
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR.

Примечание:

1. Какой из параметров - str или pTextItem используется, определяется значением параметра type.
2. При использовании ANSI следует использовать структуру параметров BaseParam.

BrandLeaderParam – Структура параметров линии-выноски для обозначения клеймения

Справка системы КОМПАС...

CM_BRANDLEADER.htmАналог данных параметров при использовании Automation - интерфейс ksBrandLeaderParam.

int	dirX	направление полки вдоль оси X (1 - вправо, -1 - влево),
double	x, y	координаты базовой точки (начало первой полки, точка выхода из нее "ножки"),
unsigned char	arrowType	тип стрелки,
unsigned short	style1	стиль текста в знаке клеймения (если style = 0, то стиль умолчательный, если style = INDICATIN_TEXT_LINE_ARR, то pText - массив TEXT_LINE_ARR,
unsigned short	style2	стиль текстов у ножки (если style = 0, то стиль умолчательный),
unsigned char	cText0	количество строк текста в знаке клеймения (не более 1 строки),
unsigned char	cText1	количество строк текста над "ножкой" (не более 1 строки),
unsigned char	cText2	количество строк текста под "ножкой" (не более 1 строки),
reference	pText	динамический массив строк: если style1 = INDICATIN_TEXT_LINE_ARR, то TEXT_LINE_ARR, если style1 - другой, то CHAR_STR_ARR или CHAR_STR_ARR_W,

reference	pPolyline	POLYLINE_ARR - массив неопределенной длины "ножек" (ответвлений) линии-выноски.

Примечания:

1. Если cText0 = 0 или cText1 = 0 или cText2 = 0, то соответствующий текст на линии-выноске отсутствует.
2. Строки текста в массиве TEXT_LINE_ARR должны учитывать последовательность расположения текстов на линии-выноске (текст над полкой, текст над "ножкой", текст под "ножкой").
3. В общем случае одна "ножка" в массиве POLYLINE_ARR - это ломаная линия. Ее первый узел - базовая точка линии-выноски. Базовую точку в массив помещать не нужно - она общая для всех ответвлений. Остальные узлы - изломы "ножки" (они могут отсутствовать). Последний узел - конец ответвления (указывает на объект).

ChangeLeaderParam – Структура параметров линии-выноски для обозначения изменения

Справка системы КОМПАС...

CM_CHANGELEADER.htm

Аналог данных параметров при использовании Automation - интерфейс ksChangeLeaderParam.

double	x	координаты базовой точки (начало первой полки),
double	y	
unsigned short	style	стиль текста,
unsigned char	signType	тип значка,
double	signHeight	высота значка,
double	leaderLength	длина выноски (< 0 - на всю длину),
reference	pText	если style = INDICATIN_TEXT_LINE_ARR, то pText = TEXT_LINE_ARR - динамический массив строк текста, если style = 0, то pText = CHAR_STR_ARR или CHAR_STR_ARR_W - динамический массив строк символов текста. Строки текстов лежат в следующей последовательности: - текст над полкой, - текста над ножкой, - текст под ножкой,

reference	pPolyline	POLYLINE_ARR - массив неопределенной длины ответвлений линии выноски. В общем случае одно ответвление - это полилиния. Первый узел - базовая точка (ее в массив помещать не нужно - общая для всех ответвлений). Остальные узлы - изломы могут отсутствовать), последний узел - конец ответвления (указывает на объект).

CutLineParam – Структура параметров линии разреза/сечения

Справка системы КОМПАС...

414_37_6_Liniy_razreza.htmАналог данных параметров при использовании Automation - интерфейс ksCutLineParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию),
unsigned char	right	признак положения стрелок (0 - слева, 1 - справа по направлению ломаной),
double	x1, y1	координаты надписи у первого участка,
double	x2, y2	координаты надписи у второго участка,
unsigned char	type	способ задания надписи на линии (0 - текст в виде строки, 1 - динамический массив компонентов текста),
char	str[50]	текст на линии,
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR,
reference	pMathPoint	динамический массив POINT_ARR точек ломаной линии (начальная точка, точки перегибов, конечная точка).

Примечание:

1. Какой из параметров - str или pTextItem используется, определяется значением параметра type.
2. При использовании Unicode следует использовать структуру параметров CutLineParamW.

CutLineParamW – Структура параметров линии разреза/сечения (Unicode)

Справка системы КОМПАС...

414_37_6_Liniy_razreza.htmАналог данных параметров при использовании Automation - интерфейс ksCutLineParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию),
unsigned char	right	признак положения стрелок (0 - слева, 1 - справа по направлению ломаной),
double	x1, y1	координаты надписи у первого участка,
double	x2, y2	координаты надписи у второго участка,
unsigned char	type	способ задания надписи на линии (0 - текст в виде строки, 1 - динамический массив компонентов текста),
wchar_t	str[50]	текст на линии,
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR,
reference	pMathPoint	динамический массив POINT_ARR точек ломаной линии (начальная точка, точки перегибов, конечная точка).

Примечание:

1. Какой из параметров - str или pTextItem используется, определяется значением параметра type.
2. При использовании ANSI следует использовать структуру параметров CutLineParam.

ksTolerancePar – Структура параметров обозначения допуска формы

Аналог данных параметров при использовании Automation - интерфейс ksToleranceParam.

unsigned char	tBase	число от 1 до 8, определяющее возможные положения базовой точки,
unsigned short	style	стиль текста (0 - стиль по умолчанию),
double	x, y	координаты базовой точки,
unsigned char	type	ориентация таблицы допуска (0 - горизонтально, 1- вертикально),
reference	branchArr	- динамический массив опор (структур ToleranceBranch),

LeaderParam – Структура параметров линии-выноски

Справка системы КОМПАС...

CM_LEADER.htmАналог данных параметров при использовании Automation - интерфейс ksLeaderParam.

double	x, y	координаты базовой точки (начало полки, точка выхода из нее "ножки"),
unsigned char	arrowType	Типы отрисовки стрелок для линии выноски...,
int	dirX	направление полки (0 - нет полки, -1 - полка влево, 1 - полка вправо, 2 - полка вверх, 3 - полка вниз),
unsigned char	signType	тип знака на "ножке" (0 - знак отсутствует, 1 - знак склеивания, 2 - знак пайки, 3 - знак сшивания, 4 - знак соединения внахлестку металлическими скобами, 5 - знак углового соединения металлическими скобами, 6 - знак монтажного шва,
unsigned char	around	простановка знака обработки по контуру (0 - выключена, 1 - включена),
unsigned char	cText0	количество строк текста над полкой,
unsigned char	cText1	количество строк текста под полкой,
unsigned char	cText2	количество строк текста над "ножкой" (не более 1 строки),
unsigned char	cText3	количество строк текста под "ножкой" (не более 1 строки),
reference	pTextline	TEXT_LINE_ARR - динамический массив строк текста,
reference	pPolyline	POLYLINE_ARR - массив неопределенной длины "ножек" (ответвлений) линии-выноски.

Примечания:

1. Если cText0 = 0, или cText1 = 0, или cText2 = 0, или cText3 = 0, то соответствующий текст на линии-выноске отсутствует.
2. Строки текста в массиве TEXT_LINE_ARR должны учитывать последовательность расположения текстов на линии-выноске (текст над полкой, текст под полкой, текст над "ножкой", текст под "ножкой").
3. В общем случае одна "ножка" в массиве POLYLINE_ARR - это ломаная линия. Ее первый узел - базовая точка линии-выноски. Базовую точку в массив помещать не нужно - она общая для всех ответвлений. Остальные узлы - изломы "ножки" (они могут отсутствовать). Последний узел - конец ответвления (указывает на объект).

PosLeaderParam – Структура параметров позиционной линии-выноски

Справка системы КОМПАС...

CM_POSITIONLEADER.htmАналог данных параметров при использовании Automation - интерфейс ksPosLeaderParam.

unsigned short	style	стиль текста, если style = 0, то стиль умолчательный, если style = INDICATIN_TEXT_LINE_ARR, то pText - массив TEXT_LINE_ARR,
double	x, y	координаты базовой точки (начало первой полки, точка выхода из нее "ножки"),
unsigned char	arrowType	Типы отрисовки стрелок для линии выноски...,
int	dirX	направление полки вдоль оси X (1 - вправо, -1 - влево),
int	dirY	направление построения полок вдоль оси Y (1 - вверх, -1 - вниз),
reference	pText	динамический массив строк: если style = INDICATIN_TEXT_LINE_ARR, то TEXT_LINE_ARR, если style - другой, то CHAR_STR_ARR или CHAR_STR_ARR_W,
reference	pPolyline	POLYLINE_ARR - массив неопределенной длины "ножек" (ответвлений) линии-выноски.

Примечания:

1. В каждой строке массива TEXT_LINE_ARR, или CHAR_STR_ARR, или CHAR_STR_ARR_W лежит один номер позиции.
2. В общем случае одна "ножка" в массиве POLYLINE_ARR - это ломаная линия. Ее первый узел - базовая точка линии-выноски. Базовую точку в массив помещать не нужно - она общая для всех ответвлений. Остальные узлы - изломы "ножки" (они могут отсутствовать). Последний узел - конец ответвления (указывает на объект).

MarkerLeaderParam – Структура параметров линии-выноски для обозначения маркировки

Справка системы КОМПАС...

CM_BRANDLEADER.htmАналог данных параметров при использовании Automation - интерфейс ksMarkerLeaderParam.

double	x, y	координаты базовой точки (начало первой полки, точка выхода из нее "ножки"),

unsigned char	arrowType	тип стрелки,
unsigned short	style1	стиль текста в знаке маркировки (если style = 0, то стиль умолчательный, если style = INDICATIN_TEXT_LINE_ARR, то pText - массив TEXT_LINE_ARR,
unsigned short	style2	стиль текстов у ножки (если style = 0, то стиль умолчательный),
unsigned char	cText0	количество строк текста в знаке маркировки (не более 1 строки),
unsigned char	cText1	количество строк текста над "ножкой" (не более 1 строки),
unsigned char	cText2	количество строк текста под "ножкой" (не более 1 строки),
reference	pText	динамический массив строк: если style1 = INDICATIN_TEXT_LINE_ARR, то TEXT_LINE_ARR, если style1 - другой, то CHAR_STR_ARR или CHAR_STR_ARR_W,
reference	pPolyline	POLYLINE_ARR - массив неопределенной длины "ножек" (ответвлений) линии-выноски

Примечания:

1. Если cText0 = 0, или cText1 = 0, или cText2 = 0, то соответствующий текст на линии-выноске отсутствует.
2. Строки текста в массиве TEXT_LINE_ARR должны учитывать последовательность расположения текстов на линии-выноске (текст над полкой, текст над "ножкой", текст под "ножкой").
3. В общем случае одна "ножка" в массиве POLYLINE_ARR - это ломаная линия. Ее первый узел - базовая точка линии-выноски. Базовую точку в массив помещать не нужно - она общая для всех ответвлений. Остальные узлы - изломы "ножки" (они могут отсутствовать). Последний узел - конец ответвления (указывает на объект).

QualityContensParam – Структура параметров квалитета

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksQualityContensParam.

LtQualSystem	systemQuality	система квалитета,
LtQualDir	kindQuality	тип квалитета,
char	name[MAX_TEXT_LENGTH]	поле допуска,
reference	pQualityItems	массив интервалов.

Примечание.

При использовании Unicode следует использовать структуру параметров QualityContensParamW.

QualityContensParamW – Структура параметров квалитета (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksQualityContensParam.

LtQualSystem	systemQuality	система квалитета,
LtQualDir	kindQuality	тип квалитета,
wchar_t	name[MAX_TEXT_LENGTH]	поле допуска,
reference	pQualityItems	массив интервалов.

Примечание.

При использовании ANSI следует использовать структуру параметров QualityContensParam.

QualityItemParam – Структура параметров интервала квалитета

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksQualityItemParam.

short	minLimit	минимальное значение в интервале размеров,
short	maxLimit	максимальное значение в интервале размеров,
double	high	верхнее отклонение,
double	low	нижнее отклонение.

RemoteElementParam – Структура параметров объекта "Выносной элемент"

Справка системы КОМПАС...

CM_REMOTE_ELEMENT.htmАналог данных параметров при использовании Automation - интерфейс ksRemoteElementParam.

unsigned short	style	стиль текста. если style = INDICATIN_TEXT_LINE_ARR, то pText - массив TEXT_LINE_ARR, если style = 0, то pText = TEXT_LINE_ARR,
long	signType	тип значка из перечислителя LtRemoteElmSignType,
double	x	координаты центра выносного элемента,
double	y	
double	width	ширина (для прямоугольника и скругленного прямоугольника),
double	height	высота (для прямоугольника и скругленного прямоугольника),

double	smooth	высота (для прямоугольника и скругленного прямоугольника),
double	radius	радиус окружности (для окружности),
double	shelfX;	координаты начала полки,
double	shelfY	
int	shelfDir	направление полки: 1 - вправо; -1- влево, 2 - вверх, 3 - вниз,
reference	pText	если style = INDICATIN_TEXT_LINE_ARR, то pText = TEXT_LINE_ARR - динамический массив строк текста, если style = 0, то pText = CHAR_STR_ARR или CHAR_STR_ARR_W - динамический массив строк символов текста шероховатости.

RoughPar – Структура параметров обозначения шероховатости

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRoughPar.

unsigned short	style	стиль текста; (если style = 0, то стиль умолчательный, если style = INDICATIN_TEXT_LINE_ARR, то pText - массив TEXT_LINE_ARR,
unsigned char	type	тип шероховатости (0 - вид обработки не устанавливается, 1 - обработка удалением слоя материала, 2 - обработка без удаления слоя материала),
unsigned char	around	признак наличия обозначения контура: 0 - обычный знак шероховатости, 1 - шероховатость "по контуру",
double	x, y	координаты точки привязки,
double	ang	угол наклона оси значка шероховатости к оси X,
unsigned char	cText0	количество строк в тексте над знаком,
unsigned char	cText1	количество строк в тексте над полкой,
unsigned char	cText2	количество строк в первом тексте под полкой (не более 2 строк),
unsigned char	cText3	количество строк во втором тексте под полкой (не более 1 строки),

reference	pText	динамический массив строк: - если style = INDICATIN_TEXT_LINE_ARR, то TEXT_LINE_ARR, - если style - другой, то CHAR_STR_ARR или CHAR_STR_ARR_W.

Примечания:

1. Текст над знаком содержит параметры шероховатости по ГОСТ 2789-73.

   Текст над полкой содержит вид обработки и дополнительные указания по ГОСТ 2789-73.

   Первый текст под полкой содержит базовую длину по ГОСТ 2789-73.

   Второй текст под полкой содержит условное обозначение направления неровностей по ГОСТ 2789-73.

2. Если cText0 = 0, или cText1 = 0, или cText2 = 0, или cText3 = 0, то соответствующий текст в обозначении шероховатости отсутствует.

RoughParam – Структура параметров обозначения шероховатости с выносной полкой

Справка системы КОМПАС...

223_28_3_2_Nastrojka_otrisovki_.htmАналог данных параметров при использовании Automation - интерфейс ksRoughParam.

RoughPar	rPar	параметры знака шероховатости,
ShelfPar	shPar	параметры выносной полки.

Структура параметров RoughPar...

Структура параметров ShelfPar...

SpecRoughParam – Структура параметров знака неуказанной шероховатости

Справка системы КОМПАС...

221_28_3_1_Vvod_nadpisi_oboznac.htmАналог данных параметров при использовании Automation - интерфейс ksSpecRoughParam.

unsigned short	style	номер стиля текста,
unsigned char	sign	тип знака (0 - вид обработки не устанавливается, 1 - обработка удалением слоя материала, 2 - обработка без удаления слоя материала),
unsigned char	t	наличие знака в скобках (0 - нет знака, 1 - есть знак),
char	s[TEXT_LENGTH]	текст.

Примечание.

При использовании Unicode следует использовать структуру параметров SpecRoughParamW.

SpecRoughParamW – Структура параметров знака неуказанной шероховатости (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpecRoughParam.

unsigned short	style	номер стиля текста,
unsigned char	sign	тип знака (0 - вид обработки не устанавливается, 1 - обработка удалением слоя материала, 2 - обработка без удаления слоя материала),
unsigned char	t	наличие знака в скобках (0 - нет знака, 1 - есть знак),
wchar_t	s[TEXT_LENGTH]	текст.

Примечание.

При использовании ANSI следует использовать структуру параметров SpecRoughParam.

TechnicalDemandParam – Структура параметров технических требований

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTechnicalDemandParam.

reference	pGab	динамический массив габаритных прямоугольников RECT_ARR (структуры RectParam),
unsigned short	style	номер стиля текста,
unsigned short	strCount	количество строк в технических требованиях.

ToleranceBranch – Структура параметров "опоры" допуска формы

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksToleranceBranch.

unsigned char	arrowType	тип опоры; (0 - нет опоры, 1 - треугольник, 2 - стрелка),
unsigned char	tCorner	число от 1 до 8, определяющее возможные положения точки выхода "опоры" из таблицы.,
reference	pMathPoint	динамический массив математических точек (POINT_oARR).

Примечания:

В динамический массив математических точек (POINT_ARR) не нужно помещать первый узел "опоры" - точку на таблице (она определена параметром tCorner). Остальные узлы - точки изломов "опоры" (они могут отсутствовать). Последний узел - конец опоры (указывает на опорный объект).

ToleranceParam – Структура параметров обозначения допуска формы

Аналог данных параметров при использовании Automation - интерфейс ksToleranceParam.

Замечание:

Структура устарела. Функции GetObjParam и SetObjParam с ней больше не работают.Рекомендуется использовать ksTolerancePar.

unsigned char	tBase	число от 1 до 8, определяющее возможные положения базовой точки,
unsigned short	style	стиль текста (0 - стиль по умолчанию),
double	x, y	координаты базовой точки,
unsigned char	type	ориентация таблицы допуска (0 - горизонтально, 1- вертикально),
ToleranceBranch	branch1	структура параметров первой "опоры",
ToleranceBranch	branch2	структура параметров второй "опоры".

Структура параметров ToleranceBranch...

Примечания:

Чтобы вторая "опора" отсутствовала, ее параметр arrowType должен быть равен нулю.

ViewPointerParam – Структура параметров стрелки направления взгляда

Аналог данных параметров при использовании Automation - интерфейс ksViewPointerParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию),
double	x1, y1	координаты вершины ("острия") стрелки,
double	x2, y2	координаты конечной точки стрелки,
double	xt, yt	координаты точки привязки текста,
unsigned char	type	способ задания надписи на линии (0 - текст в виде строки, 1 - динамический массив компонентов текста),
char	str[50]	текст на стрелке,
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR.

Примечание.

При использовании Unicode следует использовать структуру параметров ViewPointerParamW.

ViewPointerParamW – Структура параметров стрелки направления взгляда (Unicode)

Аналог данных параметров при использовании Automation - интерфейс ksViewPointerParam.

unsigned short	style	стиль текста (0 - стиль по умолчанию)
double	x1, y1	координаты вершины ("острия") стрелки
double	x2, y2	координаты конечной точки стрелки
double	xt, yt	координаты точки привязки текста
unsigned char	type	способ задания надписи на линии (0 - текст в виде строки, 1 - динамический массив компонентов текста)
wchar_t	str[50]	текст на стрелке
reference	pTextItem	динамический массив компонентов текста TEXT_ITEM_ARR

Примечание.

При использовании ANSI следует использовать структуру параметров ViewPointerParam.

Структуры параметров спецификации

DocAttachedSpcParam – Структура параметров документа, подключенного к объекту спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksDocAttachedSpcParam.

char	fileName [MAX_TEXT_LENGTH]	имя файла подключенного документа,
char	comment [MAX_TEXT_LENGTH]	комментарий к подключенному документу,
unsigned char	transmit	признак передачи в документ изменений в объекте спецификации (0 - не передавать в изменения в документ, 1 - передавать в изменения в документ).

Примечание.

При использовании Unicode следует использовать структуру параметров DocAttachedSpcParamW.

DocAttachedSpcParamW – Структура параметров документа, подключенного к объекту спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksDocAttachedSpcParam.

wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла подключенного документа,
wchar_t	comment [MAX_TEXT_LENGTH]	комментарий к подключенному документу,
unsigned char	transmit	признак передачи в документ изменений в объекте спецификации (0 - не передавать в изменения в документ, 1 - передавать в изменения в документ).

Примечание.

При использовании ANSI следует использовать структуру параметров DocAttachedSpcParam.

NumberTypeAttrParam – Структура числового значения в колонке спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksNumberTypeAttrParam.

double	minValue	минимальное значение в колонке спецификации,
double	maxValue	максимальное значение в колонке спецификации.

Примечание:

Данная структура заполняется для числовых типов значений в колонке спецификации (SPC_DOUBLE и SPC_INT).

RecordTypeAttrParam – Структура записи в колонке спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRecordTypeAttrParam.

char	attrLibName [MAX_TEXT_LENGTH]	имя файла библиотеки типов атрибутов,
int	key1	значения ключей атрибутов, служащих
int	key2	шаблонами заполнения колонки
int	key3	спецификации в данном разделе.
int	key4	

Примечания:

1. Данная структура заполняется, если тип значения в колонке спецификации - запись (SPC_RECORD).
2. Если значение какого-либо ключа 0, то этот ключ не учитывается.
3. При использовании Unicode следует использовать структуру параметров RecordTypeAttrParamW.

RecordTypeAttrParamW – Структура записи в колонке спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksRecordTypeAttrParam.

wchar_t	attrLibName [MAX_TEXT_LENGTH]	имя файла библиотеки типов атрибутов,
int	key1	значения ключей атрибутов,
int	key2	служащих шаблонами
int	key3	заполнения колонки
int	key4	спецификации в данном разделе.

Примечания:

1. Данная структура заполняется, если тип значения в колонке спецификации - запись (SPC_RECORD).
2. Если значение какого-либо ключа 0, то этот ключ не учитывается.
3. При использовании ANSI следует использовать структуру параметров RecordTypeAttrParam.

SpcColumnParam – Структура параметров колонки спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcColumnParam.

unsigned int	columnType	тип колонки,
unsigned int	ispoln	номер исполнения данного типа, начиная с 1,
unsigned int	block	номер блока,
unsigned int	typeVal	тип данных в колонке,
char	name [TEXT_LENGTH]	имя колонки.

Типы колонок спецификации...

Типы данных в колонках...

Примечание:

1. Тип данных в колонках спецификации columnType может принимать значения: SPC_CLM_FORMAT...SPC_CLM_USER.
2. Тип колонки спецификации typeVal может принимать значения: INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.
3. При использовании Unicode следует использовать структуру параметров SpcColumnParamW.

SpcColumnParamW – Структура параметров колонки спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcColumnParam.

unsigned int	columnType	тип колонки,
unsigned int	ispoln	номер исполнения данного типа, начиная с 1,
unsigned int	block	номер блока,
unsigned int	typeVal	тип данных в колонке,
wchar_t	name [TEXT_LENGTH]	имя колонки.

Типы колонок спецификации...

Типы данных в колонках...

Примечание:

1. Тип данных в колонках спецификации columnType может принимать значения: SPC_CLM_FORMAT...SPC_CLM_USER.
2. Тип колонки спецификации typeVal может принимать значения: LINT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.
3. При использовании ANSI следует использовать структуру параметров SpcColumnParam.

SpcDescrParam – Структура параметров описания спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcDescrParam.

char	layoutName [MAX_TEXT_LENGTH]	имя файла библиотеки стилей,
unsigned int	styleId	номер стиля в библиотеке,
char	spcName [MAX_TEXT_LENGTH]	имя подключенного файла спецификации, имеющей данный стиль.

Примечание.

При использовании Unicode следует использовать структуру параметров SpcDescrParamW.

SpcDescrParamW – Структура параметров описания спецификации (Unicode)

Аналог данных параметров при использовании Automation - интерфейс ksSpcDescrParam.

wchar_t	layoutName [MAX_TEXT_LENGTH]	имя файла библиотеки стилей,
unsigned int	styleId	номер стиля в библиотеке,
wchar_t	spcName [MAX_TEXT_LENGTH]	имя подключенного файла спецификации, имеющей данный стиль.

Примечание.

При использовании ANSI следует использовать структуру параметров SpcDescrParam.

SpcObjParam – Структура параметров объекта спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcObjParam.

reference	docArr	документы, прикрепленные к объекту спецификации,
unsigned short	typeObj	тип строки спецификации,
unsigned short	numbSection	номер раздела, которому принадлежит объект,
unsigned short	blockNumber	номер блока исполнений (нумерация блоков начинается с нуля),
unsigned short	numbSubSection	номер подраздела, которому принадлежит объект,
char	subSectionName [MAX_TEXT_LENGTH]	имя подраздела, которому принадлежит объект,
unsigned char	firstOnSheet	признак принудительного разрыва страницы перед объектом (0 - разрыва нет, 1 - объект располагается с начала страницы),
unsigned char	insFrgType	признак объекта спецификации, пришедшего со вставкой фрагмента (0 - самостоятельный объект, 1 - объект из вставки фрагмента, 2 - объект из вставки фрагмента редактировался в документ,

unsigned	char posInc	признак возрастания позиции объекта (0 - позиция совпадает с позицией предыдущего объекта, 1 - позиция возрастает),
unsigned char	first	признак уникальности текстовой части объекта (1 - уникальный объект, 0 - объект имеет такую же текстовую часть, как ближайший расположенный над ним уникальный объект,
unsigned char	draw	признак отображения объекта в таблице (0 - объект не показывать, 1 - объект показывать),
unsigned char	posNotDraw	признак отображения номера позиции объекта (0 - позицию показывать, 1 - не показывать).
unsigned char	ispoln	признак объекта-исполнения (0 - объект не является исполнением 1 - объект является исполнением).

Структура параметров документов, прикрепленных к объекту DocAttachedSpcParamW...

Типы строк спецификации...

Примечания:

1. В качестве параметра docArr используется динамический массив DOC_SPCOBJ_ARR, содержащий структуры параметров документов, прикрепленных к объекту спецификации (DocAttachedSpcParam).
2. Параметры numbSection, subSectionName, typeObj, insFrgType, first используются только при чтении параметров функцией GetObjParam. Функцией SetObjParam установить значения этих параметров невозможно.
3. Параметры first, draw, posNotDraw, ispoln используются только для базовых объектов спецификации.
4. Параметр blockNumber используется только для вспомогательных объектов спецификации.
5. Параметры docArr, numbSubSection, subSectionName, firstOnSheet, insFrgType и posInc используются для базовых и вспомогательных объектов спецификации.
6. При использовании Unicode следует использовать структуру параметров SpcObjParamW.

SpcObjParamW – Структура параметров объекта спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcObjParam.

reference	docArr	документы, прикрепленные к объекту спецификации,
unsigned short	typeObj	тип строки спецификации,
unsigned short	numbSection	номер раздела, которому принадлежит объект,
unsigned short	blockNumber	номер блока исполнений (нумерация блоков начинается с нуля),
unsigned short	numbSubSection	номер подраздела, которому принадлежит объект,
wchar_t	subSectionName [MAX_TEXT_LENGTH]	имя подраздела, которому принадлежит объект,
unsigned char	firstOnSheet	признак принудительного разрыва страницы перед объектом (0 - разрыва нет, 1 - объект располагается с начала страницы),
unsigned char	insFrgType	признак объекта спецификации, пришедшего со вставкой фрагмента (0 - самостоятельный объект, 1 - объект из вставки фрагмента, 2 - объект из вставки фрагмента редактировался в документ,
unsigned	char posInc	признак возрастания позиции объекта (0 - позиция совпадает с позицией предыдущего объекта, 1 - позиция возрастает),
unsigned char	first	признак уникальности текстовой части объекта (1 - уникальный объект, 0 - объект имеет такую же текстовую часть, как ближайший расположенный над ним уникальный объект,

unsigned char	draw	признак отображения объекта в таблице (0 - объект не показывать, 1 - объект показывать),
unsigned char	posNotDraw	признак отображения номера позиции объекта (0 - позицию показывать, 1 - не показывать),
unsigned char	ispoln	признак объекта-исполнения (0 - объект не является исполнением, 1 - объект является исполнением).

Структура параметров документов, прикрепленных к объекту DocAttachedSpcParamW...

Типы строк спецификации...

Примечания:

1. В качестве параметра docArr используется динамический массив DOC_SPCOBJ_ARR, содержащий структуры параметров документов, прикрепленных к объекту спецификации (DocAttachedSpcParam).
2. Параметры numbSection, subSectionName, typeObj, insFrgType, first используются только при чтении параметров функцией GetObjParam. Функцией SetObjParam установить значения этих параметров невозможно.
3. Параметры first, draw, posNotDraw, ispoln используются только для базовых объектов спецификации.
4. Параметр blockNumber используется только для вспомогательных объектов спецификации.
5. Параметры docArr, numbSubSection, subSectionName, firstOnSheet, insFrgType и posInc используются для базовых и вспомогательных объектов спецификации.
6. При использовании ANSI следует использовать структуру параметров SpcObjParam.

SpcStyleParam – Структура параметров стиля спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleParam.

char	layoutName1 [MAX_TEXT_LENGTH]	имя файла библиотеки оформлений для первого листа спецификации,
char	layoutName2 [MAX_TEXT_LENGTH]	имя файла библиотеки оформлений для последующих листов спецификации,
unsigned int	shtType1	номер оформления из библиотеки для первого листа спецификации,

unsigned int	shtType2	номер оформления из библиотеки для последующих листов спецификации,
unsigned char	ispoInVariant	вариант оформления спецификации (0 - простая, 1 - групповая, вариант А, 2 - групповая, вариант Б, 3 - групповая, вариант В, 4 - групповая, вариант Г),
unsigned char	sectionOn	деление на разделы (0 - выключено, 1- включено),
char	rezerv[10]	служебный параметр,
SpcTuningStyleParam	tuning	параметры умолчательных настроек спецификации,
reference	arrColumn	массив структур параметров колонок SpcStyleColumnParam,
reference	arrAdditionalColumn	массив структур параметров дополнительных колонок SpcStyleColumnParam,
reference	arrSection	массив структур параметров разделов спецификации SpcStyleSectionParam,
unsigned char	type	формат листа бумаги (0 - стандартный, 1 - пользовательский),
StandartSheet	stPar	параметры листа бумаги стандартного формата,
SheetSize	usPar	параметры листа бумаги пользовательского формата,

Примечания:

1. Структура stPar заполняется, если параметр type равен 1, структура usPar заполняется, если параметр type равен 0.
2. В настоящее время параметр rezerv не используется.
3. При использовании Unicode следует использовать структуру параметров SpcStyleParamW.

SpcStyleParamW – Структура параметров стиля спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleParam.

wchar_t	layoutName1 [MAX_TEXT_LENGTH]	имя файла библиотеки оформлений для первого листа спецификации

wchar_t	layoutName2 [MAX_TEXT_LENGTH]	имя файла библиотеки оформлений для последующих листов спецификации
unsigned int	shtType1	номер оформления из библиотеки для первого листа спецификации
unsigned int	shtType2	номер оформления из библиотеки для последующих листов спецификации
unsigned char	ispolnVariant	вариант оформления спецификации (0 - простая, 1 - групповая, вариант А, 2 - групповая, вариант Б, 3 - групповая, вариант В, 4 - групповая, вариант Г)
unsigned char	sectionOn	деление на разделы (0 - выключено, 1- включено)
char	rezerv[10]	служебный параметр
SpcTuningStyleParamW	tuning	параметры умолчательных настроек спецификации
reference	arrColumn	массив структур параметров колонок SpcStyleColumnParam
reference	arrAdditionalColumn	массив структур параметров дополнительных колонок SpcStyleColumnParam
reference	arrSection	массив структур параметров разделов спецификации SpcStyleSectionParam
unsigned char	type	формат листа бумаги (0 - стандартный, 1 - пользовательский)
StandartSheet	stPar	параметры листа бумаги стандартного формата
SheetSize	usPar	параметры листа бумаги пользовательского формата

Примечания:

1. Структура stPar заполняется, если параметр type равен 1, структура usPar заполняется, если параметр type равен 0.
2. В настоящее время параметр rezerv не используется.
3. При использовании ANSI следует использовать структуру параметров SpcStyleParam.

SpcStyleColumnParam – Структура параметров стиля колонки спецификаци

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleColumnParam.

char	nameColumn [MAX_TEXT_LENGTH]	имя колонки,
unsigned int	columnType	тип колонки спецификации,
unsigned int	ispoln	номер колонки данного типа,
unsigned int	typeVal	тип значения в колонке,
unsigned char	edit	разрешение редактирования, (1 - редактировать колонку в данном разделе, 0 - не редактировать колонку в данном разделе),
unsigned char	createSum	разрешение расчета суммы (1 - рассчитывать сумму значений в колонке, 0 - не рассчитывать сумму значений в колонке),
unsigned char	multiplyToCount	разрешение умножения при расчете суммы (1 - при расчете суммы умножать значения в колонке на количество, 0 - не умножать),
unsigned char	useForSectionTitle	признак размещения имен разделов (1 - использовать колонку для показа имени раздела, 0 - не использовать колонку для показа имени раздела)
unsigned char	textDn	признак размещения текста в колонке (1 - выравнивать текст в колонке по нижней строке объекта специфИкации, 0 - выравнивать текст в колонке по верхней строке
char	rezerv[10]	служебный параметр
unsigned int	linkId	номер ячейки штампа для связи с данной колонкой

RecordTypeAttrParam	recordType	параметры записи в колонке спецификации
NumberTypeAttrParam	numberType	параметры числового значения в колонке спецификации

Типы колонок спецификации...

Структура параметров записи в колонке спецификации RecordTypeAttrParam...

Структура параметров числового значения в колонке спецификации NumberTypeAttrParam...

Типы данных в колонках...

Примечания:

1. В качестве типа значения в колонке используются INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.
2. В настоящее время параметр rezerv не используется.
3. Структура recordType заполняется, если тип значения в колонке - запись (SPC_RECORD).
4. Структура numberType заполняется для числовых значений в колонке (SPC_DOUBLE и SPC_INT).
5. При использовании Unicode следует использовать структуру параметров SpcStyleColumnParamW.

SpcStyleColumnParamW – Структура параметров стиля колонки спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleColumnParam.

wchar_t	nameColumn [MAX_TEXT_LENGTH]	имя колонки
unsigned int	columnType	тип колонки спецификации
unsigned int	ispoln	номер колонки данного типа
unsigned int	typeVal	тип значения в колонке
unsigned char	edit	разрешение редактирования (1 - редактировать колонку в данном разделе, 0 - не редактировать колонку в данном разделе)
unsigned char	createSum	разрешение расчета суммы (1 - рассчитывать сумму значений в колонке, 0 - не рассчитывать сумму значений в колонке)

unsigned char	multiplyToCount	разрешение умножения при расчете суммы (1 - при расчете суммы умножать значения в колонке на количество, 0 - не умножать)
unsigned char	useForSectionTitle	признак размещения имен разделов (1 - использовать колонку для показа имени раздела, 0 - не использовать колонку для показа имени раздела)
unsigned char	textDn	признак размещения текста в колонке (1 - выравнивать текст в колонке по нижней строке объекта спецификации, 0 - выравнивать текст в колонке по верхней строке
char	rezerv[10]	служебный параметр
unsigned int	linkId	номер ячейки штампа для связи с данной колонкой
RecordTypeAttrParam	recordType	параметры записи в колонке спецификации
NumberTypeAttrParam	numberType	параметры числового значения в колонке спецификации

Типы колонок спецификации...

Структура параметров записи в колонке спецификации RecordTypeAttrParamW...

Структура параметров числового значения в колонке спецификации NumberTypeAttrParam...

Типы данных в колонках...

Примечания:

1. В качестве типа значения в колонке используются INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.
2. В настоящее время параметр rezerv не используется.
3. Структура recordType заполняется, если тип значения в колонке - запись (SPC_RECORD).
4. Структура numberType заполняется для числовых значений в колонке (SPC_DOUBLE и SPC_INT).
5. При использовании ANSI следует использовать структуру параметров SpcStyleColumnParam.

SpcStyleSectionParam – Структура параметров стиля раздела спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleSectionParam.

char	sectionName [MAX_TEXT_LENGTH]	имя раздела
unsigned short	number	номер раздела
unsigned int	sortColumnType	общий тип колонки, по данным в которой производится сортировка
unsigned int	sortIspoln	номер колонки, по данным в которой производится сортировка
unsigned char	dataType	способ ввода данных (0 - ручное заполнение колонок, 1 - ручное заполнение или чтение из основной надписи подключенного документа)
unsigned short	sortType	тип сортировки объектов в разделе
reference	arrColumn	массив структур параметров стиля колонок спецификации
reference	arrAdditionalColumn	массив структур параметров стиля дополнительных колонок спецификации
char	rezerv[10]	служебный параметр

Типы сортировки объектов в разделе спецификации...

Структура параметров стиля колонки спецификации SpcStyleColumnParam...

Примечание:

1. В настоящее время параметр rezerv не используется.
2. При использовании Unicode следует использовать структуру параметров SpcStyleSectionParamW.

SpcStyleSectionParamW – Структура параметров стиля раздела спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcStyleSectionParam.

wchar_t	sectionName [MAX_TEXT_LENGTH]	имя раздела
unsigned short	number	номер раздела
unsigned int	sortColumnType	общий тип колонки, по данным в которой производится сортировка
unsigned int	sortIspoln	номер колонки, по данным в которой производится сортировка
unsigned char	dataType	способ ввода данных (0 - ручное заполнение колонок, 1 - ручное заполнение или чтение из основной надписи подключенного документа)
unsigned short	sortType	тип сортировки объектов в разделе

reference	arrColumn	массив структур параметров стиля колонок спецификации
reference	arrAdditionalColumn	массив структур параметров стиля дополнительных колонок спецификации
char	rezerv[10]	служебный параметр

Типы сортировки объектов в разделе спецификации...

Структура параметров стиля колонки спецификации SpcStyleColumnParamW...

Примечание:

1. В настоящее время параметр rezerv не используется.
2. При использовании ANSI следует использовать структуру параметров SpcStyleSectionParam.

SpcSubSectionParam – Структура параметров подраздела спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcSubSectionParam.

char	name [MAX_TEXT_LENGTH]	имя подраздела,
unsigned short	number	номер подраздела.

Примечание.

При использовании Unicode следует использовать структуру параметров SpcSubSectionParamW.

SpcSubSectionParamW – Структура параметров подраздела спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcSubSectionParam.

wchar_t	name [MAX_TEXT_LENGTH]	имя подраздела,
unsigned short	number	номер подраздела.

Примечание.

При использовании ANSI следует использовать структуру параметров SpcSubSectionParam.

SpcTuningSectionParam – Структура параметров настройки раздела спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcTuningSectionParam.

unsigned char	subsectionOn	признак деления на подразделы (1 - включено, 0 - выключено),
unsigned char	geometryOn	признак подключения геометрии к объектам раздела (1 - разрешено подключение геометрии, 0 - запрещено подключение геометрии),
unsigned char	positionOn	признак простановки позиций в разделе (1 - позиции ставить, 0 - позиции не ставить),
unsigned char	sortOn	признак наличия сортировки объектов в разделе (1 - сортировка включена, 0 - сортировка выключена),
unsigned char	firstOnSheet	признак размещения раздела (1 - начинать раздел с новой страницы, 0 - начинать раздел после предыдущего раздела,
char	rezerv[10]	служебный параметр,
unsigned short	rezervCount	число резервных строк (и позиций) в разделе,
unsigned short	number	номер раздела,
reference	arrSubSection	массив структур параметров подразделов данного раздела.

Структура параметров подраздела SpcSubSectionParam...

Примечания:

1. Параметр number предназначен только для чтения.
2. В настоящее время параметр rezerv не используется.

SpcTuningStyleParam – Структура параметров настройки спецификации

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcTuningStyleParam.

unsigned char	grToSP	связь сборочного чертежа со спецификацией (0 – нет связи, 1 - только вставка объектов спецификации, 2 - связь с расчетом позиций),
unsigned char	zoneCalc	расчет зон (1 - рассчитывать зоны, 0 - не рассчитывать зоны),
unsigned char	showSectionName	показ имен разделов (1 - показывать имена разделов в таблице, 0 - не показывать имена разделов),
unsigned char	positionCalc	расчет номеров позиций (1 - позиции рассчитывать 0 - позиции не рассчитывать),
unsigned char	geometryDel	признак удаления геометрии удаленного объекта спецификации (1 - удалять геометрию при удалении объекта спецификации, 0 - не удалять геометрию),
unsigned char	positionDel	зарезервировано,
unsigned char	massCalc	зарезервировано,
unsigned char	disableEmptyStr	наличие пустых строк вокруг заголовка раздела (1 - пустых строк нет, 0 - пустые строки есть),
unsigned char	insertNull	префикс автоматически формируемого номера исполнения (1 - вставлять нули перед числом, 0 - не вставлять нули перед числом),
unsigned char	insertDash	префикс автоматически формируемого номера исполнения (1 - вставлять тире перед числом, 0 - не вставлять тире перед числом),
unsigned char	disableEmptyBlockStr	наличие пустых строк вокруг заголовка блока (1 - пустых строк нет, 0 - пустые строки есть),
unsigned char	showInfoByDetBlock	порядок представления информации (0 - выдавать информацию блоками 1- выдавать информацию по объектам),
unsigned char	ispoInOn	создание исполнений объектов в спецификации (1 - разрешено, 0 - запрещено),

unsigned char	ispoInMarkFull	отображение номеров исполнений объектов (1 - обозначение исполнения показывать полностью, 0 - показывать только номер исполнения),
unsigned char	blocOnNewPage	признак размещения блоков (1 - начинать блоки с новой страницы, 0 - начинать блоки после предыдущих),
unsigned char	userTextStyle	стиль текста объектов спецификации (1 - пользовательский, 0 - из стиля спецификации),
unsigned char	countIspoIn	количество исполнений,
unsigned char	countBlock	количество блоков,
unsigned short	countIspoInEx	количество исполнений,
unsigned short	countBlockEx	количество блоков,
unsigned char	delSpcObjOnDelGeometry	удалять объекты спецификации при удалении геометрии,
char	rezerv[5]	служебный параметр,
TextStyleParam	sectionTextStyleFirst	параметры стиля текста первой строки заголовков разделов,
TextStyleParam	sectionTextStyleNext	параметры стиля текста последующих строк заголовков разделов,
TextStyleParam	objectTextStyle	параметры стиля текста объектов спецификации,
char	predefinedTextFileName [MAX_TEXT_LENGTH]	имя файла предопределенных текстов, использующегося при заполнении спецификации,
reference	arrSection	массив структур параметров настроек разделов спецификации SpcTuningSectionParam,
	copySpcObjOnCopyGeometry	копировать объекты спецификации при копировании геометрии

Примечания:

1. Значение параметра showInfoByDetBlock учитывается, только если количество исполнений больше, чем количество колонок, предназначенных для записи количества на исполнение.
2. В настоящее время параметр rezerv не используется.
3. Параметры countIspoIn и countBlock для совместимости заполняются для исполнений <= 255 и countIspoInEx == 0.
4. При использовании Unicode следует использовать структуру параметров SpcTuningStyleParamW.

SpcTuningStyleParamW – Структура параметров настройки спецификации (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksSpcTuningStyleParam.

unsigned char	grToSP	связь сборочного чертежа со спецификацией (0 – нет связи, 1 - только вставка объектов спецификации, 2 - связь с расчетом позиций)
unsigned char	zoneCalc	расчет зон (1 - рассчитывать зоны, 0 - не рассчитывать зоны)
unsigned char	showSectionName	показ имен разделов (1 - показывать имена разделов в таблице, 0 - не показывать имена разделов)
unsigned char	positionCalc	расчет номеров позиций (1 - позиции рассчитывать 0 - позиции не рассчитывать)
unsigned char	geometryDel	признак удаления геометрии удаленного объекта спецификации (1 - удалять геометрию при удалении объекта спецификации, 0 - не удалять геометрию)
unsigned char	positionDel	зарезервировано
unsigned char	massCalc	зарезервировано
unsigned char	disableEmptyStr	наличие пустых строк вокруг заголовка раздела (1 - пустых строк нет, 0 - пустые строки есть)
unsigned char	insertNull	префикс автоматически формируемого номера исполнения (1 - вставлять нули перед числом, 0 - не вставлять нули перед числом)
unsigned char	insertDash	префикс автоматически формируемого номера исполнения (1 - вставлять тире перед числом, 0 - не вставлять тире перед числом)
unsigned char	disableEmptyBlockStr	наличие пустых строк вокруг заголовка блока (1 - пустых строк нет, 0 - пустые строки есть)

unsigned char	showInfoByDetBlock	порядок представления информации (0 - выдавать информацию блоками 1- выдавать информацию по объектам)
unsigned char	ispolnOn	создание исполнений объектов в спецификации (1 - разрешено, 0 - запрещено)
unsigned char	ispolnMarkFull	отображение номеров исполнений объектов (1 - обозначение исполнения показывать полностью, 0 - показывать только номер исполнения)
unsigned char	blocOnNewPage	признак размещения блоков (1 - начинать блоки с новой страницы, 0 - начинать блоки после предыдущих)
unsigned char	userTextStyle	стиль текста объектов спецификации (1 - пользовательский, 0 - из стиля спецификации)
unsigned char	countIspoIn	количество исполнений
unsigned char	countBlock	количество блоков
unsigned short	countIspoInEx	количество исполнений
unsigned short	countBlockEx	количество блоков
unsigned char	delSpcObjOnDelGeometry	удалять объекты спецификации при удалении геометрии
char	rezerv[5]	служебный параметр
TextStyleParamW	sectionTextStyleFirst	параметры стиля текста первой строки заголовков разделов
TextStyleParamW	sectionTextStyleNext	параметры стиля текста последующих строк заголовков разделов
TextStyleParamW	objectTextStyle	параметры стиля текста объектов спецификации
wchar_t	predefinedTextFileName [MAX_TEXT_LENGTH]	имя файла предопределенных текстов, использующегося при заполнении спецификации
reference	arrSection	массив структур параметров настроек разделов спецификации SpcTuningSectionParam
	copySpcObjOnCopyGeometry	копировать объекты спецификации при копировании геометрии

Примечания:

1. Значение параметра showInfoByDetBlock учитывается, только если количество исполнений больше, чем количество колонок, предназначенных для записи количества на исполнение.
2. В настоящее время параметр rezerv не используется.
3. Параметры countIspoIn и countBlock для совместимости заполняются для исполнений <= 255 и countIspoInEx == 0.
4. . При использовании ANSI следует использовать структуру параметров SpcTuningStyleParam.

Структуры параметров текста

ParagraphParam – Структура параметров параграфа

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksParagraphParam.

unsigned short	style	номер стиля текста (0 - стиль по умолчанию),
double	x, y	координаты точки привязки текста,
double	ang	угол наклона текста,
unsigned int	hFormat	признак горизонтального форматирования (0 - нет форматирования, 1 - сужение текста, 2 - перенос на другую строку),
unsigned int	vFormat	признак вертикального форматирования (0 - нет форматирования, 1-изменение шага строк),
double	width	ширина блока форматирования,
double	height	высота блока форматирования.

Системные стили текстов...

TextDocumentParam – Структура параметров текстового документа

Справка системы КОМПАС...

1132_133_1_Obshchie_svedenija.htmАналог данных параметров при использовании Automation - интерфейс ksTextDocumentParam.

unsigned char	regim	0 - видимый режим, 1 - невидимый режим,
char	fileName [MAX_TEXT_LENGTH]	имя файла,

char	comment [TEXT_LENGTH	комментарий,
char	author [TEXT_LENGTH]	автор,
LibraryStyleParam	firstSheet	оформление первого листа (имя библиотеки стилей, номер стиля в библиотеке),
LibraryStyleParam	evenSheet	оформление четных листов (имя библиотеки стилей, номер стиля в библиотеке),
LibraryStyleParam	oddSheet	оформление нечетных листов (имя библиотеки стилей, номер стиля в библиотеке),
reference	arrTitleSheet	динамический массив оформлений титульных листов LIBRARY_STYLE_ARR,
reference	arrTailSheet	динамический массив оформлений листов заключительной части LIBRARY_STYLE_ARR,
unsigned char	type	тип документа из DocType (lt_DocTxtStandart или lt_DocTxtUser),
struct StandartSheet	stPar	параметры стандартного формата, структура StandartSheet, не используется,
struct SheetSize	usPar	параметры нестандартного формата, структура SheetSize.

Примечание.

При использовании Unicode следует использовать структуру параметров TextDocumentParamW.

TextDocumentParamW – Структура параметров текстового документа (Unicode)

Справка системы КОМПАС...

1132_133_1_Obshchie_svedenija.htmАналог данных параметров при использовании Automation - интерфейс ksTextDocumentParam.

unsigned char	regim	0 - видимый режим, 1 - невидимый режим,
wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла,
wchar_t	comment [TEXT_LENGTH	комментарий,
wchar_t	author [TEXT_LENGTH]	автор,

LibraryStyleParamW	firstSheet	оформление первого листа (имя библиотеки стилей, номер стиля в библиотеке),
LibraryStyleParamW	evenSheet	оформление четных листов (имя библиотеки стилей, номер стиля в библиотеке),
LibraryStyleParamW	oddSheet	оформление нечетных листов (имя библиотеки стилей, номер стиля в библиотеке),
reference	arrTitleSheet	динамический массив оформлений титульных листов LIBRARY_STYLE_ARR,
reference	arrTailSheet	динамический массив оформлений листов заключительной части LIBRARY_STYLE_ARR,
unsigned char	type	тип документа из DocType (lt_DocTxtStandart или lt_DocTxtUser),
struct StandartSheet	stPar	параметры стандартного формата, структура StandartSheet, multiply не используется,
struct SheetSize	usPar	параметры нестандартного формата, структура SheetSize.

Примечание.

При использовании ANSI следует использовать структуру параметров TextDocumentParam.

TextLineParam – Структура параметров строки текста

Аналог данных параметров при использовании Automation - интерфейс ksTextLineParam.

unsigned short	style	номер стиля строки текста (0 - стиль по умолчанию),
reference	pTextItem	указатель на динамический массив компонент строки текста TEXT_ITEM_ARR.

Системные стили текстов...

TextItemFont – Структура параметров шрифта компоненты строки текста

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextItemFont.

char	fontName [MAX_TEXT_LENGTH]	имя шрифта,
double	height	высота шрифта (в миллиметрах),
double	ksu	коэффициент сужения символов,
unsigned	long color	цвет символов,
unsigned int	bitVector	битовый вектор, определяющий начертание символов (наклон, толщину, подчеркивание, тип составной части - дробь, отклонение и т.д.).

Примечание.

При использовании Unicode следует использовать структуру параметров TextItemFontW.

TextItemFontW – Структура параметров шрифта компоненты строки текста (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextItemFont.

wchar_t	fontName [MAX_TEXT_LENGTH]	имя шрифта,
double	height	высота шрифта (в миллиметрах),
double	ksu	коэффициент сужения символов,
unsigned	long color	цвет символов,
unsigned int	bitVector	битовый вектор, определяющий начертание символов (наклон, толщину, подчеркивание, тип составной части - дробь, отклонение и т.д.).

Примечание.

При использовании ANSI следует использовать структуру параметров TextItemFont.

TextItemParam – Структура параметров компоненты строки текста

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextItemParam.

int	tip	SPECIAL_SYMBOL или FONT_SYMBOL или FRACTION_TYPE или SUM_TYPE или 0,
TextItemFont	font	параметры шрифта компоненты строки текста,
char	s [TEXT_LENGTH]	массив символов компоненты строки текста,

unsigned int	iSNumb	номер спецсимвола, или символа из произвольного шрифта, или тип отрисовки дроби, или выражения типа суммы, или 0.

Структура параметров TextItemFont...**Примечание**.

1. Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
2. При использовании Unicode следует использовать структуру параметров TextItemParamW.

TextItemParamW – Структура параметров компоненты строки текста (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextItemParam.

int	tip	SPECIAL_SYMBOL или FONT_SYMBOL, или FRACTION_TYPE, или SUM_TYPE, или 0.
TextItemFont	font	параметры шрифта компоненты строки текста,
wchar_t	s [TEXT_LENGTH]	массив символов компоненты строки текста,
unsigned int	iSNumb	номер спецсимвола или символа из произвольного шрифта или тип отрисовки дроби, или выражения типа суммы, или 0.

Структура параметров TextItemFontW...**Примечание**.

1. Таблица спецсимволов размещена в файле SDK\NumbSymb.frw.
2. При использовании ANSI следует использовать структуру параметров TextItemParam.

TextParam – Структура параметров текста

Аналог данных параметров при использовании Automation - интерфейс ksTextParam.

ParagraphParam	par	параметры параграфа,
reference	pTextLine	указатель на динамический массив строк текста TEXT_LINE_ARR.

Структура параметров ParagraphParam...

TextStyleParam – Структура параметров стиля текста

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextStyleParam.

char	name [TEXT_LENGTH]	имя стиля,
double	height	высота текста шрифта,
double	ksu	коэффициент сужения символов,
double	step	шаг строк,
char	fontName [TEXT_LENGTH]	имя шрифта,
unsigned long	color	цвет,
unsigned char	align	выравнивание (0 - влево, 1 - по центру, 2 - вправо, 3 - на всю ширину,
unsigned char	bold	начертание символов (0 - не утолщенные, 1 - утолщенные),
unsigned char	italic	начертание символов (0 - без наклона, 1 - с наклоном),
unsigned char	underline	подчеркивание (0 - без подчеркивания, 1 - с подчеркиванием),
double	posKS	отступ красной строки (в миллиметрах)
double	stepParPre	интервал перед текстом,
double	stepParPst	интервал после текста,
double	leftEdge	отступ текста слева,
double	rightEdge	отступ текста справа.

Примечание.

При использовании Unicode следует использовать структуру параметров TextStyleParamW.

TextStyleParamW – Структура параметров стиля текста (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksTextStyleParam.

wchar_t	name [TEXT_LENGTH]	имя стиля,
double	height	высота текста шрифта,
double	ksu	коэффициент сужения символов,
double	step	шаг строк,
wchar_t	fontName [TEXT_LENGTH]	имя шрифта,
unsigned long	color	цвет,
unsigned char	align	выравнивание (0 - влево, 1 - по центру, 2 - вправо, 3 - на всю ширину,
unsigned char	bold	начертание символов (0 - не утолщенные, 1 - утолщенные),
unsigned char	italic	начертание символов (0 - без наклона, 1 - с наклоном),

unsigned char	underline	подчеркивание (0 - без подчеркивания, 1 - с подчеркиванием),
double	posKS	отступ красной строки (в миллиметрах),
double	stepParPre	интервал перед текстом,
double	stepParPst	интервал после текста,
double	leftEdge	отступ текста слева,
double	rightEdge	отступ текста справа.

Примечание.

При использовании ANSI следует использовать структуру параметров TextStyleParam.

Структуры параметров событий

NotifyConnectionParam – Структура параметров для осуществления подписки, отписки событий в COM

reference	pContainer	- указатель на объект КОМПАС, контейнер событий. Для приложения 0. Для документа указатель документа.
long	objType	Дополнительный параметр. Либо тип объекта, либо указатель на объект (для ksObject2DNotify).
int	ifType	Тип интерфейсов событий ksNotifyTypeEnum.
LPUNKNOWN	iContainer	Указатель на объект КОМПАС, контейнер сообщений.
LPUNKNOWN	iObj	Дополнительный параметр. Указатель на объект (для ksObject3DNotify).

Структуры параметров измерений

InertiaParam – Структура параметров для расчета МЦХ плоской фигуры

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksInertiaParam.

double	xc, yc	координаты центра тяжести фигуры
double	f	площадь
double	Iy, Ix	моменты инерции относительно исходных осей x и y
double	Ixy	центробежный момент инерции относительно исходных осей x и y
double	my, mx	моменты инерции относительно осей, параллельных осям x и y и проходящих через центр тяжести

double	mxy	центробежный момент инерции относительно осей, параллельных осям x и y и проходящих через центр тяжести
double	jy, jx	главные центральные моменты инерции
double	a	угол между первой главной осью и осью x

MassInertiaParam – Структура параметров для расчета МЦХ тел вращения и выдавливания

Справка системы КОМПАС...

CM_MEASURE_MIX3D.htm

Аналог данных параметров при использовании Automation - интерфейс ksMassInertiaParam.

double	r	плотность материала,
double	m	масса тела,
double	v	объем тела,
double	xc, yc, zc	координаты центра тяжести,
double	Ixy, Ixz, Iyz	центробежные моменты инерции в глобальной (исходной) системе координат,
double	Ix, Iy, Iz	осевые моменты инерции в глобальной (исходной) системе координат,
double	jy0z, jx0z, jx0y	плоскостные моменты инерции,
double	jxy, jxz, jyz	центробежные моменты инерции в центральной системе координат (с началом в центре масс и осями, параллельными осям глобальной (исходной) системы координат),
double	jx, jy, jz	осевые моменты инерции в центральной системе координат (с началом в центре масс и осями, параллельными осям глобальной (исходной) системы координат).

Структуры параметров библиотек

InsertFragmentParamEx – Структура параметров вставки фрагмента (расширенная)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksInsertFragmentParam.

PlacementParam	place	положение вставки фрагмента (PlacementParam),
unsigned char	insertType	тип вставки фрагмента (0 - взять в документ, 1 - внешней ссылкой, 3 - локальный фрагмент),

unsigned char	multiLayer	признак размещения объектов фрагмента по слоям (0 - объекты на одном слое, 1 - объекты на разных слоях),
char	fileName [MAX_TEXT_LENGTH]	имя файла фрагмента или "\0" для локального фрагмента,
char	comment [MAX_TEXT_LENGTH]	имя вставки,
unsigned char	scaleProjLinesSize	признак масштабирования длины выносных линий размеров: (1 - выносные линии масштабируются, 0 - выносные линии не масштабируются).

Примечание.

При использовании Unicode следует использовать структуру параметров InsertFragmentParamExW.

InsertFragmentParamExW – Структура параметров вставки фрагмента (расширенная), Unicode

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс

-.

PlacementParam	place	положение вставки фрагмента (PlacementParam)
unsigned char	insertType	тип вставки фрагмента (0 - взять в документ, 1 - внешней ссылкой, 3 - локальный фрагмент)
unsigned char	multiLayer	признак размещения объектов фрагмента по слоям (0 - объекты на одном слое, 1 - объекты на разных слоях)
wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла фрагмента или "\0" для локального фрагмента
wchar_t	comment [MAX_TEXT_LENGTH]	имя вставки
unsigned char	scaleProjLinesSize	признак масштабирования длины выносных линий размеров: (1 - выносные линии масштабируются, 0 - выносные линии не масштабируются)

Примечание.

При использовании ANSI следует использовать структуру параметров InsertFragmentParamEx.

LibStyle – Структура параметров для подключения стиля из библиотеки

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibStyle.

char	fileName [MAX_TEXT_LENGTH]	имя файла библиотеки,
int	styleNumber	номер стиля в библиотеке,
unsigned char	typeAllocation	тип размещения стиля в документе (0 - телом; 1 - ссылкой на библиотеку).

Примечание.

При использовании Unicode следует использовать структуру параметров LibStyleW.

LibStyleW – Структура параметров для подключения стиля из библиотеки (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibStyle.

wchar_t	fileName [MAX_TEXT_LENGTH]	имя файла библиотеки,
int	styleNumber	номер стиля в библиотеке,
unsigned char	typeAllocation	тип размещения стиля в документе (0 - телом; 1 - ссылкой на библиотеку)

Примечание.

При использовании ANSI следует использовать структуру параметров LibStyle.

LibToolbarSettings – Структура параметров панели команд библиотеки

unsigned char	defaultVisible	0 погашена, по умолчанию,
VARIANT	docsType	константа из перечислений DocumentTypeEnum и ksSlaveDocumentTypeEnum.

LibraryStyleParam – Структура параметров стиля в библиотеке стилей

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibraryStyleParam.

char	styleName [MAX_TEXT_LENGTH]	имя стиля,
unsigned int	styleId	номер стиля в библиотеке.

Примечание.

При использовании Unicode следует использовать структуру параметров LibraryStyleParamW.

LibraryStyleParamW – Структура параметров стиля в библиотеке стилей (Unicode)

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksLibraryStyleParam.

wchar_t	styleName [MAX_TEXT_LENGTH]	имя стиля,
unsigned int	styleId	номер стиля в библиотеке.

Примечание.

При использовании ANSI следует использовать структуру параметров LibraryStyleParam.

TreeNodeParam – Структура параметров узла дерева библиотеки документов, библиотеки атрибутов

Аналог данных параметров при использовании Automation - интерфейс ksTreeNodeParam.

long	type	- тип узла: корень, папка, файл (из LtNodeType),
char	name [MAX_TEXT_LENGTH]	- имя узла,
reference	comment	- комментарий - динамический массив CHAR_STR_ARR (динамический массив строк символов),
reference	nodes	- динамический массив узлов TreeNodeParam.

Примечание.

При использовании Unicode следует использовать структуру параметров TreeNodeParamW.

TreeNodeParamW – Структура параметров узла дерева библиотеки документов, библиотеки атрибутов (Unicode)

Аналог данных параметров при использовании Automation - интерфейс ksTreeNodeParam.

long	type	- тип узла: корень, папка, файл (из LtNodeType),
wchar_t	name [MAX_TEXT_LENGTH]	- имя узла,
reference	comment	- комментарий - динамический массив CHAR_STR_ARR (динамический массив строк символов),
reference	nodes	- динамический массив узлов.

Примечание.

При использовании ANSI следует использовать структуру параметров TreeNodeParam.

Структуры параметров системы

ConstraintParam – Структура параметрических связей и ограничений

Справка системы КОМПАС...

Аналог данных параметров при использовании Automation - интерфейс ksConstraintParam.

unsigned short	constrType	тип параметрического ограничения,
int	index	индекс точки на объекте (нумерация начинается с 0, у дуги и окружности 0 - центр),
reference	partner	указатель на второй объект, участвующий в параметрической связи,
int	partnerIndex	индекс точки на втором объекте (нумерация начинается с 0, у дуги и окружности 0 - центр).

Типы параметрических ограничений...

LtVariant – Структура параметров для хранения данных некоторого типа.

Аналог данных параметров при использовании Automation - интерфейс ksLtVariant.

LtVariantType	vType	тип хранимых данных,
short	colVisible	для спецификации: 1 - видимая колонка, 0 - невидимая колонка,

short	sortKey	ключ сортировки для колонки спецификации (0 - сортировки нет),		
short	wReserved3	параметр зарезервирован для дальнейшего использования,		
char	chVal	символ,	ltv_Char	1
unsigned char	ucVal	байт,	ltv_Uchar	2
int	iVal	целое число,	ltv_Int	3
unsigned int	uiVal	беззнаковое целое число,	ltv_Uint	4
long	lVal	длинное целое число,	ltv_Long	5
float	fVal	вещественное число,	ltv_Float	6
double	dVal	двойное число,	ltv_Double	7
char	strVal [MAX_TEXT_LENGTH]	строка,	ltv_Str	8
short	shVal	короткое целое число.	ltv_Short	10

Примечание.

При использовании Unicode следует использовать структуру параметров LtVariantEx.

LtVariantEx – Структура параметров для хранения данных некоторого типа (Unicode)

Аналог данных параметров при использовании Automation - интерфейс ksLtVariant.

LtVariantType	vType	тип хранимых данных,		
short	colVisible	для спецификации: 1 - видимая колонка, 0 - невидимая колонка,		
short	sortKey	ключ сортировки для колонки спецификации (0 - сортировки нет),		
short	wReserved3	параметр зарезервирован для дальнейшего использования,		
union				
char	chVal	символ,	ltv_Char	1
unsigned char	ucVal	байт,	ltv_Uchar	2
int	iVal	целое число,	ltv_Int	3
unsigned int	uiVal	беззнаковое целое число,	ltv_Uint	4
long	lVal	длинное целое число,	ltv_Long	5
float	fVal	вещественное число,	ltv_Float	6
double	dVal	двойное число,	ltv_Double	7
wchar_t	strVal [MAX_TEXT_LENGTH]	строка,	ltv_Str	8
short	shVal	короткое целое число.	ltv_Short	10
wchar_t	wstrVal[MAX_TEXT_LENGTH]	строка	ltv_WStr	11

Примечание.

При использовании ANSI следует использовать структуру параметров LtVariant.

RequestInfo – Структура параметров запроса к системе

Аналог данных параметров при использовании Automation - интерфейс ksRequestInfo.

char	*title	строка или идентификатор строки заголовка окна,
char	*commands	строка или идентификатор меню состава команд,
char	*prompt	строка или идентификатор приглашения,
char	*cursor	строка с именем или идентификатор стандартного курсора,
void	*callBack	указатель на функцию обратной связи,
int	dynamic	признак динамического запроса (1 - динамический запрос, 0 - статический),
HINSTANCE	commInstance	идентификатор модуля, в котором размещен состав команд.

Примечание:

1. Эта структура используется в функциях Cursor, Placement, CommandWindow.
2. В случае динамического запроса функция обратной связи вызывается по перемещению курсора. В случае статического запроса функция обратной связи вызывается по щелчку мыши.
3. Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (Idefin2d.h):


OCR_SELECT


OCR_SNAP


OCR_DEFAULT


OCR_CATCH

RequestInfoW – Структура параметров запроса к системе (Unicode)

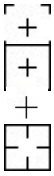
Аналог данных параметров при использовании Automation - интерфейс ksRequestInfo.

wchar_t	*title	строка или идентификатор строки заголовка окна,
wchar_t	*commands	строка или идентификатор меню состава команд,
wchar_t	*prompt	строка или идентификатор приглашения,
wchar_t	*cursor	строка с именем или идентификатор стандартного курсора,
void	*callBack	указатель на функцию обратной связи,

int	dynamic	признак динамического запроса (1 - динамический запрос, 0 - статический),
HINSTANCE	commInstance	идентификатор модуля, в котором размещен состав команд.

Примечание:

1. Эта структура используется в функциях Cursor, Placement, CommandWindow.
2. В случае динамического запроса функция обратной связи вызывается по перемещению курсора. В случае статического запроса функция обратной связи вызывается по щелчку мыши.
3. Кроме стандартных констант из WINUSER.H можно использовать стандартные курсоры системы КОМПАС (ldefin2d.h):



OCR_SELECT

OCR_SNAP

OCR_DEFAULT

OCR_CATCH

VariableParam – Структура параметров параметрической переменной

char	name [MAX_TEXT_LENGTH]	имя переменной,
double	value	значение переменной,
char	note [MAX_TEXT_LENGTH]	комментарий.

Примечание.

При использовании Unicode следует использовать структуру параметров VariableParamW.

VariableParamW – Структура параметров параметрической переменной (Unicode)

wchar_t	name [MAX_TEXT_LENGTH]	имя переменной,
double	value	значение переменной,
wchar_t	note [MAX_TEXT_LENGTH]	комментарий.

Примечание.

При использовании ANSI следует использовать структуру параметров VariableParam.

Структуры параметров окна Свойства

PropertyParam – Структура параметров свойства отображаемого в окне свойств

int	propertyType	Тип свойства из перечисления ksObjectPropertyControlTypeEnum,
int	propertyId;	Идентификатор свойства - используется для загрузки имени и параметров свойства из ресурсов,
HINSTANCE	propertyInstance;	Модуль ресурсов для загрузки параметров свойства (имя, список значений...),
LPOLESTR	displayPropertyName;	Отображаемое имя свойства (необязательное),
VARIANT	propertyValue;	Текущее значение,
double	propertyMinValue;	Минимальное значение,
double	propertyMaxValue;	Максимальное значение,
int	isDefCpyProp;	Будет отмечено для копирования,
int	enable;	Признак доступности для изменения,
int	emptyValue;	Значение не задано,
VARIANT	additionalData;	Дополнительные данные для свойства,
LPOLESTR	additionalText;	Дополнительный текст,
int	summList;	TRUE - объединять списки, FALSE - пересекать списки.

Примечание:

Поля propertyType, propertyId и propertyInstance являются обязательными для заполнения. Остальные поля заполняются в зависимости от типа контрола.

Константы

Перечисления событий

ksDocument2DNotifyEnum – События графического документа

d2BeginRebuild	1	Начало перестроения модели
d2Rebuild	2	Модель перестроена
d2BeginChoiceMaterial	3	Начало выбора материала
d2ChoiceMaterial	4	Закончен выбор материала
d2BeginInsertFragment	5	Начало вставки фрагмента (до диалога выбора имени)
d2LocalFragmentEdit	6	Редактирование локального фрагмента
d2BeginChoiceProperty	7	Начало выбора свойства
d2ChoiceProperty	8	Закончен выбор свойства

ksDocument3DNotifyEnum – События документа-модели

d3BeginRebuild	1	Начало перестроения модели
d3Rebuild	2	Модель перестроена
d3BeginChoiceMaterial	3	Начало выбора материала
d3ChoiceMaterial	4	Закончен выбор материала
d3BeginChoiceMarking	5	Начало выбора обозначения
d3ChoiceMarking	6	Закончен выбор обозначения
d3BeginSetPartFromFile	7	Начало установки компонента в сборку (до диалога выбора имени)
d3BeginCreatePartFromFile	8	Начало создания компонента в сборке (до диалога выбора имени)
d3BeginChoiceProperty	12	Начало выбора свойства
d3ChoiceProperty	13	Закончен выбор свойства
d3BeginRollbackFeatures	14	Начало отката дерева модели
d3RollbackFeatures	15	Завершение отката дерева модели
d3BedinLoadCombinationChange	16	Начало переключения типа загрузки
d3LoadCombinationChange	17	Завершение переключения типа загрузки

ksDocumentFileNotifyEnum – События для документов; работа с файлом

kdBeginCloseDocument	1	- начало закрытия документа
kdCloseDocument	2	- документ закрыт
kdBeginSaveDocument	3	- начало сохранения документа
kdSaveDocument	4	- документ сохранен
kdActiveDocument	5	- документ активизировался
kdDeactiveDocument	6	- документ деактивизировался
kdBeginSaveAsDocument	7	- начало сохранения документа с другим именем (до диалога выбора имени)
kdDocumentFrameOpen	8	- окно документа открылось
kdProcessActivate	9	- процесс активизирован
kdProcessDeactivate	10	- процесс деактивизирован
kdBeginProcess	11	- начало процесса
kdEndProcess	12	- завершение процесса
kdBeginAutoSaveDocument	13	- начало автосохранения документа
kdAutoSaveDocument	14	- Документ автосохранен

ksKompasObjectNotifyEnum – События приложения

koCreateDocument	1	- документ создан
koBeginOpenDocumen	2	- начало открытия документа
koOpenDocumen	3	- документ открыт,
koChangeActiveDocument	4	- переключение на другой активный документ
koApplicatinDestroy	5	- закрытие приложения.
koBeginCreate	6	- начало создания документа (до диалога выбора типа)
koBeginOpenFile	7	- начало открытия документа (до диалога выбора имени)
koBeginCloseAllDocument	8	- начало закрытия всех открытых документов
koKeyDown	9	- событие нажатия клавиатуры. клавиша нажата.
koKeyUp	10	- событие нажатия клавиатуры. клавиша отпущена
koKeyPress	11	- событие нажатия клавиатуры. клавиша нажата
koIsNeedConvertToSavePrevious	15	Начало сохранения документа в предыдущую версию
koBeginConvertToSavePrevious	16	Начало конвертации документа перед записью в предыдущую верию
koEndConvertToSavePrevious	17	Завершение конвертации документа перед записью в предыдущую версию
koChangeTheme	18	Изменение темы

ksLayoutSheetsNotifyEnum – События для листов оформления

ksLayoutAdd	1	Добавлен лист
ksLayoutDelete	2	Удален лист оформления
ksLayoutUpdate	3	Изменены параметры листа оформления

ksLibraryManagerNotifyEnum – События для менеджера библиотек

ksLMBeginAttach	1	Подключить библиотеку
ksLMAttach	2	Библиотека подключена
ksLMBeginDetach	3	Отключить библиотеку
ksLMDetach	4	Библиотека отключена
ksLMBeginExecute	5	Запуск выполнения команды библиотеки

ksLMEndExecute	6	Завершение выполнения команды библиотеки
ksLMSystemControlStop	7	Передача управления библиотеке
ksLMSystemControlStart	8	Передача управления системе
ksLMAddLibraryDescription	9	Добавлено описание библиотеки
ksLMDeleteLibraryDescription	10	Удалено описание библиотеки
ksLMAddInsert	11	Добавлен документ в библиотеку документов
ksLMDeleteInsert	12	Удален документ из библиотеки документов
ksLMEditInsert	13	Редактирование документа из библиотеки документов
ksLMTryExecute	14	Попытка вызвать команду библиотеки
ksLMBeginInsertDocument	15	Запуск вставки документа из библиотеки

ksNotifyTypeEnum – Перечень интерфейсов событий

ntKompasObjectNotify	1	События приложения
ntDocumentFileNotify	2	События документа, работа с файлом
ntStampNotify	3	События основной надписи
ntObject2DNotify	4	События объекта графического документа
ntSelectionMngNotify	5	События менеджера выделенных объектов
ntSpcObjectNotify	6	События объекта спецификации
ntSpcDocumentNotify	7	События документа-спецификации
ntSpecificationNotify	8	События спецификации
ntDocument3DNotify	9	События документа-модели
ntObject3DNotify	10	События объекта документа-модели
ntDocument2DNotify	11	События графического документа
ntPropertyManagerNotify	12	События для Панели свойств
ntPropertyUserControlNotifyEnum	13	События пользовательского элемента управления
ntDocumentFrameNotify	14	События для окна документа
ntViewsAndLayersManagerNotify	15	События для менеджера видов и слоев
ntLibraryManagerNotify	16	События для менеджера библиотек
ntProcess2DNotify	18	События процесса 2D
ntProcess3DNotify	19	События процесса 3D
ntContentDialogNotify	20	События диалога с внешним наполнением
ntFindObjectParametersNotify	21	События функции поиска объектов под курсором
ntProcess3DManipulatorsNotify	22	События манипуляторов процесса 3D
ntPLMObjectNotify	23	События объектов версионирования

ksObject2DNotifyEnum – События объектов графических документов

koChangeActive	1	переключение активности объекта(вид, слой)
koBeginDelete	2	начало удаления объекта
koDelete	3	объект удален

koBeginMove	4	начало сдвига объекта
koMove	5	завершение сдвига объекта,
koBeginRotate	6	начало поворота объекта
koRotate	7	завершение поворота объекта
koBeginScale	8	начало масштабирования объекта
koScale	9	завершение масштабирования объекта
koBeginTransform	10	начало трансформации объекта
koTransform	11	завершение трансформации объекта
koBeginCopy	12	начало копирования объекта,
koCopy	13	завершение копирования объекта
koBeginSymmetry	14	начало симметричного преобразования объекта
koSymmetry	15	завершение симметричного преобразования объекта
koBeginProcess	16	начало редактирования\создания объекта
koEndProcess	17	завершение редактирования\создания объекта
koCreateObject	18	создание объекта
koUpdateObject	19	редактирование объекта
koBeginDestroyObject	20	начало разрушения объекта
koDestroyObject	21	разрушение объекта
koBeginPropertyChanged	22	начало изменения свойств объекта
koPropertyChanged	23	изменены свойства объекта

ksObject3DNotifyEnum – События объектов документа-модели

o3BeginDelete	1	Начало удаления объектов
o3Delete	2	Объекты удалены
o3Excluded	3	Объект исключен/включен в расчет
o3Hidden	4	Объект скрыт/показан
o3BeginPropertyChanged	5	Начало изменения свойств объекта
o3PropertyChanged	6	Изменены свойства объекта
o3BeginPlacementChanged	7	Начало изменения положения объекта
o3PlacementChanged	8	Начало изменения положения объекта
o3BeginProcess	9	Начало
o3EndProcess	10	Конец редактирования\создания объекта
o3CreateObject	11	Создание объекта
o3UpdateObject	12	Редактирование объекта
o3BeginLoadStateChange	13	Начало изменения типа загрузки
o3LoadStateChange	14	Завершение изменения типа загрузки

ksProcess2DTypeEnum – Типы процессов 2D

ksProcess2DCursor	1	- запрос на получение точки
ksProcess2DPlacement	2	- запрос на получение точки и угла

ksProcess3DTypeEnum – Типы процессов 3D

ksProcess3DPlacementAndEntity	1	- запрос на указание местоположения и объекта
ksProcess3DSelectEntity	2	- запрос на выбор объекта

ksProcess2DNotifyEnum – События процесса 2D

ksProcess2DPlacementChanged	1	- изменение положения
ksProcess2DExecuteCommand	2	- выполнить команду меню
ksProcess2DRun	3	- запуск процесса
ksProcess2DStop	4	- остановка процесса
ksProcess2DActivate	5	- активизация процесса
ksProcess2DDeactivate	6	- деактивизация процесса
ksProcess2DEndProcess	7	- окончание процесса
ksProcess2DMouseEnterLeavePara m	8	- запрос параметров точек для визуального определения места применения параметра

ksProcess3DNotifyEnum – События процесса 3D

ksProcess3DPlacementChanged	1	- изменение положения
ksProcess3DExecuteCommand	2	- выполнить команду меню
ksProcess3DRun	3	- запуск процесса
ksProcess3DStop	4	- остановка процесса
ksProcess3DActivate	5	- активизация процесса
ksProcess3DDeactivate	6	- деактивизация процесса
ksProcess3DFilterObjects	7	- фильтрация объектов
ksProcess3DCreateTakeObject	8	- событие создания объекта в - - - подчиненном режиме
ksProcess3DEndProcess	9	- окончание процесса
ksProcess3DProcessingGroupObje cts	10	- обработать объекты, пришедшие при селектировании рамкой

ksTwinSwitcherValueEnum – Значения переключателя

ksTwinSwitcherPos1	1	- положение 1
ksTwinSwitcherPos2	2	- положение 2

ksToleranceArrowType – Тип стрелки ответвления у допуска формы

ksTANone	0	- нет
ksTAArrow	1	- стрелка
ksksTATriangle	2	- треугольник

ksEnterButtonIconTypeEnum – Тип иконки для кнопки Создать

ksEnterCheckIcon	0	Обычная кнопка. Создать в виде галочки
ksEnterFloppyIcon	1	Дискета
ksEnterNewInputIcon	2	Новый ввод
ksEnterApplyIcon	3	Применить

ksShowHideTmpObjTypeEnum – Тип отображения временного объекта в документе

ksTmpObjHide	0	- объект скрыт
ksTmpObjShow	1	- объект отображается как обычный объект
ksTmpObjShowPhantom	2	- объект отображается как фантом

ksSelectionMngNotifyEnum – События менеджера выделенных объектов

ksmSelect	1	- объект выделен
ksmUnselect	2	- снято выделение с объекта
ksmUnselectAll	3	- снято выделение со всех объектов

ksStampNotifyEnum – События основной надписи графических документов (штампа)

kdBeginEditStamp	1	Начало работы со штампом
kdEndEditStamp	2	Завершение работы со штампом
kdStampCellDblClick	3	Двойной щелчок мышью в ячейке штампа
kdStampCellBeginEdit	4	Начало редактирования в ячейке штампа
kdStampBeginClearCells	5	Начало очистки ячеек штампа

ksSpcDocumentNotifyEnum – События документа-спецификации

sdDocumentBeginAdd	1	Начало добавления документа сборочного чертежа
sdDocumentAdd	2	Добавлен документ сборочного чертежа
sdDocumentBeginRemove	3	Начало удаления документа сборочного чертежа
sdDocumentRemove	4	Удален документ сборочного чертеж
sdSpcStyleBeginChange	5	Начало изменения стиля спецификации
sdSpcStyleChange	6	Стиль спецификации изменился

ksSpcObjectNotifyEnum – События объекта спецификации

soBeginDelete	1	Начало удаления объекта
soDelete	2	Объект удален
soCellDblClick	3	Двойной щелчок в ячейке
soCellBeginEdit	4	Начало редактирования в ячейке
soChangeCurrent	5	Текущий объект изменен
soDocumentBeginAdd	6	Начало добавления документа
soDocumentAdd	7	Документ в объекте спецификации добавлен
soDocumentRemove	8	Документ из объекта спецификации удален
soBeginGeomChange	9	Начало изменения геометрии объекта спецификации
soGeomChange	10	Геометрия объекта спецификации изменилась
soBeginProcess	11	Начало редактирования\создания объекта
soEndProcess	12	Конец редактирования\создания объекта
soCreateObject	13	Объект создан
soUpdateObject	14	Объект изменен
soBeginCopy	15	Начало копирования объекта
soCopy	16	Копирование объекта

ksSpecificationNotifyEnum – События для спецификации

ssTuningSpcStyleBeginChange	1	Начало изменения настроек спецификации
ssTuningSpcStyleChange	2	Настройки спецификации изменились
ssChangeCurrentSpcDescription	3	Изменилось текущее описание спецификации
ssSpcDescriptionAdd	4	Добавилось описание спецификации
ssSpcDescriptionRemove	5	Удалилось описание спецификации
ssSpcDescriptionBeginEdit	6	Начало редактирования описания спецификации
ssSpcDescriptionEdit	7	Отредактировали описание спецификации
ssSynchronizationBegin	8	Начало синхронизации
ssSynchronization	9	Синхронизация проведена
ssBeginCalcPositions	10	Начало расчета позиций
ssCalcPositions	11	Проведен расчет позиций
ssBeginCreateObject	12	Начало создания объекта спецификации (до диалога выбора раздела)

Константы спецификации

Варианты оформления спецификации

Параметр variant может иметь следующие значения для выбора варианта оформления спецификации:

0	- простая	
1	- групповая, вариант А	
2	- групповая, вариант Б	
3	- групповая, вариант В	- временно недоступен
4	- групповая, вариант Г	- временно недоступен

Типы колонок спецификации

SPC_CLM_FORMAT	1	- формат
SPC_CLM_ZONE	2	- зона
SPC_CLM_POS	3	- позиция
SPC_CLM_MARK	4	- обозначение
SPC_CLM_NAME	5	- наименование
SPC_CLM_COUNT	6	- количество
SPC_CLM_NOTE	7	- примечание
SPC_CLM_MASSA	8	- масса

SPC_CLM_MATERIAL	9	- материал
SPC_CLM_USER	10	- пользовательская
SPC_CLM_KOD	11	- код
SPC_CLM_FACTORY	12	- предприятие-изготовитель

Типы объектов спецификации

В API7 соответствует перечисление ksSpecificationObjectTypeEnum - Типы объектов спецификации.

SPC_BASE_OBJECT	1	- базовый объект
SPC_COMMENT	2	- вспомогательный объект
SPC_SECTION_NAME	3	- заголовок раздела
SPC_BLOCK_NAME	4	- заголовок блока исполнений
SPC_RESERVE_STR	5	- резервная строка
SPC_EMPTY_STR	6	- пустая строка в конце страницы

Типы сортировки объектов в разделе спецификации

SPC_SORT_OFF	0	- нет сортировки
SPC_SORT_COMPOS	1	- составная сортировка
SPC_SORT_ALPHABET	2	- сортировка по алфавиту
SPC_SORT_UP	3	- сортировка по возрастанию числового значения
SPC_SORT_DOCUMENT	4	- сортировка раздела документация
SPC_SORT_DOWN	5	- сортировка по убыванию числового значения

Типы данных в столбце табличного атрибута и в колонках спецификации

CHAR_ATTR_TYPE	1	- целое (от -128 до 127)
UCHAR_ATTR_TYPE	2	- целое (от 0 до 255)
INT_ATTR_TYPE	3	- целое (от -32768 до 32767)
UINT_ATTR_TYPE	4	- целое (от 0 до 65535)
LINT_ATTR_TYPE	5	- целое
FLOAT_ATTR_TYPE	6	- действительное (от -1E38 до 1E38)
DOUBLE_ATTR_TYPE	7	- действительное (от -1E308 до 1E307)
STRING_ATTR_TYPE	8	- строка фиксированной длины MAX_TEXT_LENGTH
RECORD_ATTR_TYPE	9	- запись

Примечание:

Тип данных в колонках спецификации может принимать значения: INT_ATTR_TYPE, DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE и RECORD_ATTR_TYPE.

ksSpecificationStyleDifferenceTypeEnum – Отличие стиля спецификации от библиотечного

ksSpcStyleEqual	0	Стиль спецификации не отличается от библиотечного
ksSpcStyleDistinguish	1	Стиль спецификации отличается от библиотечного
ksSpcStyleNotFound	-1	Стиль спецификации не найден в библиотеке стилей

ksSheetTypeEnum – Тип листа

ksDocumentSheet	0	Лист документа
ksFrontAdditionalSheet	1	Дополнительный лист в начале документа
ksLastAdditionalSheet	2	Дополнительный лист в конце документа

Константы размеров

Типы отрисовки стрелок в размерах

0	- стрелки нет
1	- стрелка изнутри
2	- стрелка снаружи
3	- засечка
4	- вспомогательная точка
5	- стрелка закрытая изнутри
6	- стрелка закрытая снаружи
7	- стрелка открытая изнутри
8	- стрелка открытая снаружи
9	- стрелка 90 град. изнутри
10	- стрелка 90 град. снаружи
11	- точка
12	- маленькая точка
0xff	- размер с обрывом

Примечание:

1. Диаметр точки равен длине стрелки размера, диаметр маленькой точки равен 0,6 длины стрелки размера.
2. Параметры стрелок для размеров, линий выносок и позиционных линий выносок в системе настраиваются отдельно.

Типы размеров высоты

OD_FRONTVIEW	0x00	- для вида спереди, с полкой и стрелкой, возможна выносная линия
OD_TOPVIEW	0x08	- для вида сверху без линии-выноски - только текст в рамке
OD_TOPVIEWLEADER	0x10	- для вида сверху с линией-выноской

Признаки размерной надписи

_AUTONOMINAL	0x1	- автоматическое определение номинального значения размера
_RECTTEXT	0x2	- текст в рамке
_PREFIX	0x4	- есть текст до номинала
_NOMINALOFF	0x8	- нет номинала
_TOLERANCE	0x10	- проставлять квалитет
_DEVIATION	0x20	- проставлять отклонения
_UNIT	0x40	- единица измерения
_SUFFIX	0x80	- есть текст после номинала
_DEVIATION_INFORM	0x100	- при включенном флаге _DEVIATION отклонения есть в массиве текстов (даже если простановка отклонений - не ручная)
_UNDER_LINE	0x200	размер с подчеркиванием
_BRACKETS	0x400	размер в скобках
_SQUARE_BRACKETS	0x800	размер в квадратных скобках, используется вместе с _BRACKETS

Константы документов

Стандартные форматы листа

0	- формат A0
1	- формат A1
2	- формат A2
3	- формат A3
4	- формат A4

Типы локальной привязки

	0	- нет локальной привязки
SN_NEAREST_POINT	1	- Ближайшая точка
SN_NEAREST_MIDDLE	2	- Середина
SN_CENTRE	3	- Центр
SN_INTERSECT	4	- Пересечение
SN_GRID	5	- По сетке
SN_XY_ALIGN	6	- Выравнивание
SN_ANGLE	7	- Угловая привязка

SN_POINT_CURVE	8	- Точка на кривой

Общие настройки привязок

SN_DYNAMICALLY	0x1	- динамически отслеживать привязки
SN_ASSISTANT	0x2	- отображать название действующей привязки
SN_BACKGROUND_LAYER	0x4	- учитывать фоновые слои и виды
SN_SUSPENDED	0x8	- подавить привязки

Состояния слоев и видов

stACTIVE	0	- активный (видимый фоновый)
stREADONLY	1	- фоновый
stINVISIBLE	2	- невидимый (погашенный)
stCURRENT	3	- текущий

Форматы для сохранения модели

FORMAT_SAT	1	- SAT
FORMAT_XT	2	- XT
FORMAT_STEP	3	- STEP
FORMAT_IGES	4	- IGES
FORMAT_VRML	5	- VRML
FORMAT_STL	6	- STL

Форматы растра

FORMAT_BMP	0	- BMP
FORMAT_GIF	1	- GIF
FORMAT_JPG	2	- JPEG
FORMAT_PNG	3	- PNG
FORMAT_TIF	4	- TIFF
FORMAT_TGA	5	- TGA
FORMAT_PCX	6	- PCX
FORMAT_WMF	16	- WMF (не поддерживается)
FORMAT_EMF	17	- EMF

Цвета вывода объектов

BLACKWHITE	0	- черный
COLORVIEW	1	- цвет, установленный для вида
COLORLAYER	2	- цвет, установленный для слоя
COLOROBJECT	3	- цвет, установленный для объекта

Глубина цвета растра

BPP_COLOR_01	1	- монохромный
BPP_COLOR_02	2	- 4 цвета
BPP_COLOR_04	4	- 16 цветов
BPP_COLOR_08	8	- 256 цветов
BPP_COLOR_16	16	- 16 разрядов
BPP_COLOR_24	24	- 24 разряда
BPP_COLOR_32	32	- 32 разряда

Типы стандартных видов

VIEW_FRONT	0x1	Спереди
VIEW_REAR	0x2	Сзади
VIEW_UP	0x4	Сверху
VIEW_DOWN	0x8	Снизу
VIEW_LEFT	0x10	Слева
VIEW_RIGHT	0x20	Справа
VIEW_ISO	0x40	Изометрия

См. также Типы данных в столбце табличного атрибута

ksDrawInScreenPlaneEnum – Способ преобразования координат внешней триангуляции

ksDrawNone	0	- Не преобразовывать координаты в плоскость экран
ksDrawInScreenPlane	1	- Координаты Ox и Oy совпадают с экранными
ksDrawProjectToScreen	2	- Координаты Oz Совпадает с экранными, Ox И Oy проецируются на экран
ksDrawProjectFromScreen	3	- Координаты Ox остаются неизменными, координаты Oz проецируются с экранными

Константы моделей

ksBooleanType – Типы булевых операций над твердыми телами

ksBooleanUnknown	0	Неизвестность
ksIntersect	1	Пересечение
ksDifference	2	Вычитание
ksUnion	3	Объединение

Примечание:

При типе направления dtMiddlePlane в методах SetSideParam и GetSideParam параметр depth интерпретируется как общая глубина выдавливания и задается следующим образом:

SetSideParam(TRUE, etBlind, depth, ...)

Типы операций выдавливания

etBlind	0	строго на глубину
etThroughAll	1	через всю деталь
etUpToVertexTo	2	на расстояние до вершины
etUpToVertexFrom	3	на расстояние за вершину
etUpToSurfaceTo	4	на расстояние до поверхности
etUpToSurfaceFrom	5	на расстояние за поверхность
etUpToNearSurface	6	до ближайшей поверхности

Примечание:

1. При типах выдавливания etUpToVertexTo, etUpToVertexFrom, etUpToSurfaceTo и etUpToSurfaceFrom в методах SetSideParam и GetSideParam параметр depth интерпретируется как глубина, вычитаемая или добавляемая к расстоянию до указанного объекта. Объект, определяющий глубину, задается с помощью метода SetDepthObject.
2. В API7 соответствуют перечислению ksEndTypeEnum.

Константы графических объектов

ksContourFormEnum – форма контура обозначения узла

ksUFormCircle	0	Окружность
ksUFormRectangle	1	Прямоугольник
ksUFormCRectangle	2	Прямоугольник со скругленными вершинами

Константы типов интерфейсов параметров для COM в 2D

ko_ParametrizeParam	9000	ksParametrizationParam - параметры параметризации группы объектов.

Типы отрисовки стрелок для линии-выноски

0	- стрелка отсутствует
1	- вспомогательная точка
2	- стрелка (по умолчанию)
3	- односторонняя стрелка сверху

4	- односторонняя стрелка снизу
5	- засечка
6	- засечка с наклоном влево
7	- стрелка 90 град.
8	- стрелка закрытая
9	- стрелка открытая
10	- точка
11	- точка маленькая
12	- база 60 град.
13	- база 90 град.

Примечание:

1. Диаметр точки равен длине стрелки линии выноски, диаметр маленькой точки равен 0,6 длины стрелки линии выноски.
2. Параметры стрелок для размеров, линий выносок и позиционных линий выносок в системе настраиваются отдельно.
3. Для обозначения клеймения используются только 0,1,2.
4. Для обозначения знака маркировки используют только 0,1,2.
5. Для обозначения позиционной линии выноски используют только 0,1,2.

Системные стили отрисовки точек

Параметр style может иметь следующие значения для выбора системного стиля отрисовки точки:

0	- точка
1	- крестик
2	- х-точка
3	- квадрат
4	- треугольник
5	- окружность
6	- звезда
7	- перечеркнутый квадрат
8	- утолщенный плюс

Системные стили линий

Параметр style может иметь следующие значения для выбора системного стиля линии:

1	- основная
2	- тонкая
3	- осевая
4	- штриховая
5	- для линии обрыва
6	- вспомогательная
7	- утолщенная
8	- пунктир 2
9	- штриховая осн
10	- осевая осн
11	- тонкая линия, включаемая в штриховку
12	- ISO 02 штриховая линия

13	- ISO 03 штриховая линия (дл. пробел)
14	- ISO 04 штрихпунктирная линия (дл. штрих)
15	- ISO 05 штрихпунктирная линия (дл. штрих 2 пунктира)
16	- ISO 06 штрихпунктирная линия (дл. штрих 3 пунктира)
17	- ISO 07 пунктирная линия
18	- ISO 08 штрихпунктирная линия (дл. и кор. штрихи)
19	- ISO 09 штрихпунктирная линия (дл. и 2 кор. штриха)
20	- ISO 10 штрихпунктирная линия
21	- ISO 11 штрихпунктирная линия (2 штриха)
22	- ISO 12 штрихпунктирная линия (2 пунктира)
23	- ISO 13 штрихпунктирная линия (3 пунктира)
24	- ISO 14 штрихпунктирная линия (2 штриха 2 пунктира)
25	- ISO 15 штрихпунктирная линия (2 штриха 3 пунктира)

Параметры пера пользовательского стиля линии

Второе поле параметра curveType может иметь следующие значения для выбора толщины пера при отрисовке пользовательского стиля линии:

LIKE_BASIC_LINE	0x10	- параметры пера как у системной основной линии
LIKE_THIN_LINE	0x20	- параметры пера как у системной тонкой линии
LIKE_HEAVY_LINE	0x30	- параметры пера как у системной утолщенной линии

Системные стили штриховок

Параметр style может иметь следующие значения для выбора системного стиля штриховки:

0	- металл
1	- неметалл
2	- дерево
3	- камень естественный
4	- керамика
5	- бетон
6	- стекло
7	- жидкость
8	- естественный грунт
9	- насыпной грунт
10	- камень искусственный
11	- железобетон
12	- напряженный железобетон
13	- дерево в продольном сечении
14	- песок

Системные типы значков

0	- без значка
1	- стрелка изнутри

2	- стрелка снаружи
3	- засечка с продолжением кривой (с "хвостиком")
4	- верхняя половина стрелки изнутри
5	- нижняя половина стрелки изнутри
6	- большая (7 мм) стрелка изнутри
7	- стрелка для размера высоты (штрихи длиной 4 мм, расположенные под углом 45 градусов)
8	- треугольник по направлению кривой
9	- окружность радиусом 2 мм тонкой линией
10	- обозначение фиктивного центра в виде большого "креста"
11	- знак склеивания
12	- знак пайки
13	- знак сшивания
14	- знак соединения внахлестку металлическими скобами
15	- знак углового соединения металлическими скобами
16	- знак монтажного шва
17	- засечка без продолжения кривой (без "хвостика")
18	- треугольник по текущей СК - для базы
19	- закрытая стрелка изнутри
20	- закрытая стрелка снаружи
21	- открытая стрелка изнутри
22	- открытая стрелка снаружи
23	- стрелка 90 град. изнутри
24	- стрелка 90 град. снаружи
25	- точка
26	- точка маленькая

Типы фантомов

Параметр phType может иметь следующие значения для выбора типа фантома:

1	- сдвиг группы
2	- отрезок
3	- прямоугольник
4	- отрезок с заданным углом
5	- половина прямоугольника
6	- пользовательский фантом
7	- окружность

Типы фантомов

0		- удаление фантома
1	ksType1	- сдвиг группы
2	ksType2	- отрезок
3	ksType3	- прямоугольник
4	ksType3	- отрезок с заданным углом
5	ksType5	- половина прямоугольника
6	ksType6	- пользовательский фантом (только рисовать группу, матрицу сдвига не устанавливать)
7	ksType2	окружность

Типы параметров объектов

ALLPARAM В зависимости от типа компиляции...	-1	- все параметры объекта
SHEET_ALLPARAM	-2	- все параметры объекта в СК листа
NURBS_CLAMPED_ALLPARAM	-5	- параметры NURBS; преобразовать узловой вектор в зажатый
NURBS_CLAMPED_SHEET_ALLPARAM	-6	- параметры NURBS в СК листа; преобразовать узловой вектор в зажатый
VIEW_ALLPARAM	-7	- все параметры объекта в СК вида
ANGLE_ARC_PARAM	0	- параметры дуги по углам (для дуги и эллиптической дуги)
POINT_ARC_PARAM	1	- параметры дуги по точкам (для дуги и эллиптической дуги)
VIEW_LAYER_STATE	1	- состояние слоя, вида
DOCUMENT_STATE	1	- состояние документа
ANGLE_ARC_SHEET_PARAM	2	- параметры дуги по углам (для дуги и эллиптической дуги) в СК листа
POINT_ARC_SHEET_PARAM	3	- параметры дуги по точкам (для дуги и эллиптической дуги) в СК листа
ANGLE_ARC_VIEW_PARAM	4	- параметры дуги по углам (для дуги и эллиптической дуги) в СК вида,
POINT_ARC_VIEW_PARAM	5	- параметры дуги по точкам (для дуги и эллиптической дуги) в СК вида
DOCUMENT_SIZE	0	- размер листа,
DIM_TEXT_PARAM	0	- параметры текста для размеров
DIM_SOURSE_PARAM	1	- параметры привязки размера
DIM_DRAW_PARAM	2	- параметры отрисовки размера
DIM_VALUE	3	- значение размера
DIM_PARTS	4	- составляющие части для размеров (структура DimensionPartsParam),
SHEET_DIM_PARTS	5	- составляющие части для размеров (структура DimensionPartsParam в СК листа)
TECHNICAL_DEMAND_PAR	-1	- параметры технических требований
TT_FIRST_STR	1000	- начало отсчета для получения или замены текста техтребований по строкам
CONIC_PARAM	2	- параметры конического сечения (для эллипса и эллиптической дуги)

SPC_TUNING_PARAM	0	- параметры настроек для стиля спецификации
ALL_OBJ	0	- все объекты, которые могут входить в вид, кроме вспомогательных
ANGLE_ARC_SHEET_PARAM	2	- параметры дуги по углам (для дуги и эллиптической дуги) в СК листа
POINT_ARC_SHEET_PARAM	3	- параметры дуги по точкам (для дуги и эллиптической дуги) в СК листа
ANGLE_ARC_VIEW_PARAM	4	- параметры дуги по углам (для дуги и эллиптической дуги) в СК вида
POINT_ARC_VIEW_PARAM	5	- параметры дуги по точкам (для дуги и эллиптической дуги) в СК вида
ALLPARAM_W В зависимости от типа компиляции...	-20	- все параметры объекта в СК объекта владельца для структур со строками wchar_t
SHEET_ALLPARAM_W В зависимости от типа компиляции...	-21	- тоже что и ALLPARAM но параметры объекта в СК листа для структур со строками wchar_t
VIEW_ALLPARAM_W В зависимости от типа компиляции...	-22	- тоже, что и ALLPARAM, но параметры объекта в СК вида для структур со строками wchar_t
ASSOCIATION_VIEW_PARAM_W В зависимости от типа компиляции...	-23	- параметры ассоциативного вида для структуры со строками wchar_t
DIM_TEXT_PARAM_W В зависимости от типа компиляции...	6	- параметры текста для размеров для структуры со строками wchar_t
DIM_SOURSE_VIEWPARAM В зависимости от типа компиляции...	7	- параметры привязки размера в системе координат вида
DIM_DRAW_VIEWPARAM В зависимости от типа компиляции...	8	- параметры отрисовки размера в системе координат вида
DIM_SOURSE_SHEETPARAM В зависимости от типа компиляции...	9	- параметры привязки размера в системе координат листа
DIM_DRAW_SHEETPARAM В зависимости от типа компиляции...	10	- параметры отрисовки размера в системе координат листа

ksDrawingObjectParamTypeEnum – тип параметров объекта

ksAllParam	-1	Все параметры
ksSheetAllParam	-2	Все параметры объекта в СК листа
ksViewAllParam	-7	Все параметры объекта в СК вида

Константы объектов оформления

Возможные положения базовой точки на таблице допуска формы

1	- левый нижний угол
2	- середина левой стороны
3	- левый верхний угол
4	- середина верхней стороны
5	- правый верхний угол
6	- середина правой стороны
7	- правый нижний угол
8	- середина нижней стороны

Типы объектов оформления чертежа

0	- основная надпись
1	- технические требования
2	- знак неуказанной шероховатости
3	- текущий вид
4	- спецификация на листе
5	- текущий слой

ksStampEnum – Идентификаторы ячеек штампа

ksStPartNumber	1	Наименование изделия
ksStDescription	2	Обозначение документа
ksStMaterial	3	Обозначение материала
ksStMass	5	Масса изделия
ksStScale	6	Масштаб
ksStSheetNumber	7	Номер листа
ksStNumberOfSheets	8	Количество листов
ksStCompany	9	Индекс предприятия
ksStTypeOfWork	10	Характер работы
ksStDocumentLetter1	40	Литера документа (графа 1)
ksStDocumentLetter2	41	Литера документа (графа 2)
ksStDocumentLetter3	42	Литера документа (графа 3)
ksStFullFileName	43	Имя файла (полное)
ksStShortFileName	44	Имя файла (короткое)
ksStMarkingLine	45	Строка обозначения и дефис
ksStDocumentName	51	Наименование документа
ksStDocumentCode	52	Код документа
ksStOKPCode	53	Код ОКП
ksStAuthor	110	Фамилия разработавшего
ksStCheckedBy	111	Фамилия проверившего
ksStMfgApprovedBy	112	Фамилия тех. контр
ksStDesigner	113	Фамилия вып. работу
ksStRateOfInspection	114	Фамилия норм. контр
ksStApprovedBy	115	Фамилия утверждающего
ksStEndDesignDate	130	Дата окончания разработки

ksStCheckedDate	131	Дата проверки
ksStMfgApprovedDate	132	Дата тех. контр
ksStExecutionDate	133	Дата выполнения
ksStRateOfInspectionDate	134	Дата норм. контр
ksStApprovedDate	135	Дата утверждения

Константы отчетов

ksReportTypeEnum – Тип отчета

ksRTPropertiesReport	0	Отчет по свойствам
ksRTEmbodimentsReport	1	Отчет по исполнениям
ksRTPatternWithVariablesTableReport	2	Отчет по таблице изменяемых переменных в массиве
ksRTAdditionNumbersReport	3	Отчет по таблице дополнительных номеров

ksRowsNumberingTypeEnum – Формат нумерации строк отчета

ksRNTNone	0	Нумерацию не использовать
ksRNTSimple	1	Простая нумерация
ksRNTMultiLevel	2	Многоуровневая нумерация

ksNumberingTypeEnum – Формат нумерации

ksNTArabicNumerals	0	1,2,3,4,5...
ksNTRomanNumerals	1	I, II, III, IV, V...
ksNTUpperRegEnglish	2	A,B,C,D...
ksNTLowerRegEnglish	3	a,b,c,d...
ksNTUpperRegRussian	4	А,Б,В,Г...
ksNTLowerRegRussian	5	а,б,в,г...

ksReportStyleInitEnum – Способ инициализации стиля отчета

ksRSIDefault	0	Стиль по умолчанию

ksGroupeTypeEnum – Тип группировки колонки отчета

ksGTNone	0	Нет группировки
ksGTMatch	1	Совпадение - группируются только строки с одинаковыми значениями
ksGTSum	2	Сумма (Только для числовых колонок)

ksGTRange	3	Диапазон (Только для числовых колонок)
ksGTEnum	4	Перечисление

ksValueFormatEnum – Формат формирования текста для свойства

ksVFVariableValue	0x1	0x8			Значение
ksVFvvfToleranceDiv	0x2	0x4	0x8		Отклонение и допуск
ksVFvvfFullValue	0x1	0x2	0x4	0x8	Значение + допуск + отклонения + спецсимволы
ksVFvvfGridFullValue	0x1	0x2	0x4		Значение + допуск + отклонения без спецсимволов
ksVFBooleanValue	0x1				Значение 1/0
ksVFBooleanYesNo	0x2				Значение Да/Нет
ksVFBooleanPlusMinus	0x3				Значение +/-
ksVFFullMarking	0x1				Полное обозначение
ksVFOnlyAdditionMarking	0x2				Только номер исполнения

ksReportBuildingTypeEnum – Способ выбора объектов для отчета

ksRBAllObjects	0	По всем объектам
ksRBChooseObjects	1	По указанным объектам
ksRBCurrentView	2	По текущему виду
ksRBChoiceToLevel	3	До указанного уровня

ksPageLayoutTypeEnum – Тип компоновки таблиц отчета

ksRPLayoutDefault	0	Вправо, затем вниз
ksRPLayout1	1	Вниз, затем вправо

Константы текста

Признаки начертания текста

INVARIABLE	0	- не менять флаги текста
NUMERATOR	0x1	- числитель
DENOMINATOR	0x2	- знаменатель
END_FRACTION	0x3	- конец дроби
UPPER_DEVIAT	0x4	- верхнее отклонение
LOWER_DEVIAT	0x5	- нижнее отклонение
END_DEVIAT	0x6	- конец отклонений
S_BASE	0x7	- основание выражения с под- или над- строкой
S_UPPER_INDEX	0x8	- верхний индекс выражения с под- или надстрокой
S_LOWER_INDEX	0x9	- нижний индекс выражения с под- или надстрокой
S_END	0x10	- конец выражения с под- или над- строкой
SPECIAL_SYMBOL	0x11	- спецзнак
SPECIAL_SYMBOL_END	0x12	- конец спецзнака для спецзнаков с текстом
RETURN_BEGIN	0x13	- начало для ввода следующих строк в спецзнаке с текстом, дробях, отклонениях
RETURN_DOWN	0x14	- для ввода следующих строк в спецзнаке с текстом, дробях, отклонениях
RETURN_RIGHT	0x15	- для ввода строк справа в спецзнаке с текстом, дробях, отклонениях
TAB	0x16	- табуляция по текущему стилю,
FONT_SYMBOL	0x17	- символ шрифта,
ITALIC_ON	0x40	- включить наклон
ITALIC_OFF	0x80	- выключить наклон,
BOLD_ON	0x100	- включить жирное начертание
BOLD_OFF	0x200	- выключить жирное начертание
UNDERLINE_ON	0x400	- включить подчеркивание
UNDERLINE_OFF	0x800	- выключить подчеркивание
NEW_LINE	0x1000	- новая строка в параграфе
FONT_SYMBOL_W	0x2017	- символ шрифта Unicode

Системные стили текстов

Параметр style может иметь следующие значения для выбора системного стиля текста:

0	- умолчательный стиль для данного типа объекта
1	- обычный текст
2	- текст для технических требований
3	- текст размерной надписи
4	- текст в обозначении шероховатости
5	- текст на позиционной линии-выноске
6	- текст над\под полкой линии-выноски
7	- текст на ответвлении линии-выноски
8	- текст в обозначении допуска формы
9	- текст для заголовка таблицы
10	- текст для ячейки таблицы
11	- текст для линии разреза
12	- текст для стрелки направления взгляда
13	- текст в обозначении неуказанной шероховатости
14	- текст в обозначении изменения
15	- текст для фигурной скобки
16	- текст для номера узла
17	- текст для выносной линии
18	- текст для обозначения узла
19	- текст для простых обозначений
20	- текст для позиционного обозначения
21	- текст для МПО(марка/позиционное обозначение) на линии
22	- текст для МПО без линии выноски
23	- текст для заголовков спецификации

Константы системы

ksKompasCommandEnum – Команды меню и команды панели команд системы КОМПАС

Команды меню и команды панели команд системы КОМПАС

ksCMViewFullScreen	32403	Полный экран.
ksCMSaveAll	32404	Сохранить все
ksCMSaveTechnicalDemand	32405	Сохранить технические требования
ksCMSaveTechnicalDemandToTxt	32406	Сохранить технические требования в текстовый документ
ksCMCloseTechnicalDemand	32407	Закрыть технические требования
ksCMCloseSpcSlave	32408	Закрыть окно объектов спецификаций
ksCMDocumentSetup	32410	Настройки текущего документа
ksCMViewVariables	32498	Скрыть/показать панель Переменные
ksCMTutor3D	32535	Азбука КОМПАС-3D

ksCMTutor2D	32540	Азбука КОМПАС-График

Команды масштабирования и зуммирования

ksCMZoomWindow	32411	Увеличить масштаб окном
ksCMZoomIn	32412	Увеличить масштаб
ksCMZoomOut	32413	Уменьшить масштаб
ksCMScaleView	32414	Текущий масштаб (Комбобокс с масштабом)
ksCMZoomEntireDocument	32415	Показать весь документ
ksCMZoomSelected	32416	Показать полностью выделенные объекты
ksCMRefresh	32417	Обновить изображение
ksCMMoveView	32418	Сдвинуть изображение
ksCMPanoramaView	32419	Приблизить/отдалить изображение
ksCMRotateView	32420	Повернуть изображение (для 3Д-окна)
ksCMZoom1	32545	Масштаб 1,0
ksCMZoomSketch	40872	Показать эскиз полностью

Команды работы со стилями

ksCMSetAttributeTypes	32421	Типы атрибутов
ksCMSetCurveStyles	32422	Стили линий
ksCMSetTextStyles	32423	Стили текстов
ksCMSetStampStyles	32424	Редактирование изображения штампа
ksCMSetTextShape	32425	Редактирование текстовых оформлений
ksCMSetGraphicShape	32426	Редактирование графических оформлений
ksCMSetHatchStyles	32427	Стили штриховок
ksCMSetSpcStyles	32448	Стили спецификаций
ksCMSummaryInfo	32440	Информация о документе

Команды навигации по листам для многолистового документа

ksCMMoveDocumentlists_navigationEX	32522	Перейти к листу (со списком листов)
ksCMMoveDocumentFirst	32523	Перейти к первому листу документа
ksCMMoveDocumentLast	32524	Перейти к первому листу документа
ksCMMoveDocumentPrev	32525	Перейти к предыдущему листу документа

ksCMMoveDocumentNext	32526	Перейти к последующему листу документа
ksCMRetryCommand	32534	Повтор последней команды
ksCMCloseAll	32535	Закрыть все
ksCMStop	33206	Отмена текущей команды (Стоп)
ksCMRepeatFind	33207	Повторение поиска объекта
ksCMCreateObject	33700	Создать объект Создать
ksCMCansel	33701	Отказ от создания объекта (Стоп)
ksCMEscape	33211	Отмена текущей команды (По клавише Esc)
ksPrintSpecialExecute	37289	Отправить документ на специальную полистную печать

Команды работы со спецификацией

ksCMSpcObjectsSort	33796	Автоматическая сортировка объектов текущего раздела СП
ksCMSpcRebuild	33797	Перестроить спецификацию
ksCMFullPageHeight	33800	Изменить масштаб по высоте листа для спецификации
ksCMFullPageWidth	33801	Изменить масштаб по ширине листа для спецификации
ksCMSpcMakePosition	33891	Расставить позиции
ksCMSpcObjectDelete	33892	Удалить элемент спецификации
ksCMSpcObjectInsert	33893	Добавить элемент спецификации
ksCMSpcSynchronize	33896	Синхронизировать данные
ksCMSpcInsertLine	33898	Добавить строку
ksCMSpcShowAll	33900	Показать все объекты
ksCMSpcTuningSetup	33901	Настройка спецификации
ksCMSpcCopyObject	33902	Копировать текущий объект спецификации
ksCMSpcObjectMoveUp	33903	Сдвинуть текущий объект спецификации вверх
ksCMSpcObjectMoveDovn	33904	Сдвинуть текущий объект спецификации вниз
ksCMSpcObjectEdit	33905	Редактировать текст текущего объекта спецификации
ksCMSpcInsertIspoIn	33906	Добавить объекты-исполнения для текущего объекта спецификации
ksCMSpcOpenGeometryDocs	33907	Открыть документы с геометрией объекта
ksCMCpcShowExcludedObjects	33911	Показать исключенные объекты

Команды работы с фрагментами

ksCMFragmentManager	35704	Управление фрагментами
ksCMEditFragment	35705	Редактировать фрагмент
ksCMCreateLocalFragment	35706	Создать локальный фрагмент

Команды селектирования объектов и работы с селектированными объектами

Селектирование объектов и работа с селектированными \объектами		
ksCMSelectGroup	35711	Выделить группу рамкой
ksCMSelectLayer	35712	Выделить слой указанием
ksCMSelectByType	35713	Выделить по типу
ksCMStoreGroup	35715	Запомнить группу
ksCMSelectView	35716	Выделить вид из списка видов
ksCMSelectByAttr	35717	Выделить по атрибутам
ksCMSelectByCurveStyle	35718	Выделить по стилю кривой
ksCMSelectPrevList	35719	Выделить прежний список
ksCMUnSelectSroup	35720	Исключить группу
ksCMUnSelectLayer	35721	Исключить слой
ksCMUnSelectByType	35722	Исключить по типу
ksCMUnSelectView	35723	Исключить вид
ksCMUnSelectByAttr	35724	Исключить по атрибутам
ksCMUnSelectByCurveStyle	35725	Исключить по стилю кривой
ksCMUnSelectPrevList	35726	Исключить прежний список
ksCMMakeMacro	35727	Объединить в макроэлемент
ksCMDestroyMacro	35728	Разрушить макроэлемент
ksCMViewObjAttr	35729	Атрибуты объекта
ksCMChangeSelObjStyles	35730	Изменить стиль селектированных объектов
ksCMChangeSelObjLayer	35731	Изменить слой селектированных объектов
ksCMEditObject	35736	Редактировать выделенный объект
ksCMToggleToObjLayer	35737	Сделать текущим слой объекта
ksCMObjStreamline	35738	Очистить фон у объекта
ksClearAttachedLeaders	37186	Очистить оформление составного объекта

Команды удаления объектов

ksCMDelAuxCurves	35739	Удалить вспомогательные кривые во всех видах
ksCMDelAuxCurvesInCurentView	35740	Удалить вспомогательные кривые в текущем виде
ksCMDelStamp	35741	Удалить штамп

ksCMDelTechnicalDemand	35742	Удалить технические требования
ksCMDelSpecRough	35743	Удалить неуказанную шероховатость
ksCMSheetViewStates	35744	Состояния видов
ksCMViewLayerStates	35745	Слои
ksCMGridOnOf	35746	Включить/выключить отображение сетки
ksCMSnapSuspend	35748	Включить/выключить действие глобальных привязок
ksCMSnapSetup	35749	Настройка глобальных привязок
ksCMOrthoModeOnOff	35750	Включить/выключить режим ортогонального черчения
ksCMDiscreteModeOnOff	35747	Включить/выключить режим 'дискретирования' линейных величин по шагу курсора

Команды управления состояниями видов

ksCMRebuildSheet	35751	Перестроить чертеж
ksCMSheetViewParams	35752	Параметры текущего вида
ksCMTechnicalDemand	35753	Технические требования - ввод
ksCMSpecRough	35754	Неуказанная шероховатость
ksCMSlaveSpc	35755	Просмотр\редактирование объектов спецификации
ksCMAddEditSpcObject	35760	Добавить\редактировать объект спецификации
ksCMAddEditChangeListObject	35761	Добавить\редактировать объект таблицы изменений
ksCMSpcSinhronize	35763	Синхронизировать данные
ksCMSheetSpc	35764	Таблица спецификации на листе
ksCMLayoutManager	35765	Управление списком оформлений документа
ksCMAddPage	35766	Добавить лист в многолистовой документ
ksCMGoto00	36028	Поставить курсор в точку 0.0
ksCMToggleCursor	36029	Переключить размер курсора
ksCMRegulateLeaderLineX	36075	Выравнивать линии выноски по горизонтали
ksCMRegulateLeaderLineY	36076	Выравнивать линии выноски по вертикали
ksCMEditSpcObject	36077	Редактировать объект спецификации по линии-выноске
ksCMEditSpcObjectForGeom	36086	Редактировать объект спецификации по геометрии

ksCMGridSetup	36090	Настроить параметры сетки
ksCMSlaveSpcDelegate	36094	Просмотр\редактирование объектов-делегатов спецификации
ksCMAddSpcDelegate	36095	Добавить\редактировать объект-делегат спецификации
ksCMAutoCreateSpcObj	36097	Сформировать объекты спецификации для модели
ksCMLibraryBarVisible	36104	Скрыть/показать панель Библиотеки
ksCMBuildTreeView	37003	Скрыть/показать дерево построений
ksSheetViewParams	37005	Параметры вида
ksCMaximizeWorkArea	37079	Показать панели
ksCMParameters	37608	Настройки системы и новых документов
ksCMZoomUndo	38530	Предыдущий масштаб
ksCMZoomRedo	38531	Последующий масштаб
ksViewShowBreakups	39344	Вид - Показать разрывы

Команды 3D документа

ksCMHideCPlaces	40360	Скрыть\показать начала координат
ksCMHideCPlanes	40361	Скрыть\показать конструктивные плоскости
ksCMHideCAxies	40362	Скрыть\показать конструктивные оси
ksCMHideSketches	40363	Скрыть\показать эскизы
ksCMHideSurfaces	40364	Скрыть\показать поверхности
ksCMHideThreads	40365	Скрыть\показать изображения резьбы
ksCMHideCurves	40366	Скрыть\показать пространственные кривые
ksCMHidePoints	40367	Скрыть\показать конструктивные точки
ksCMHideAllObjects	40368	Скрыть\показать вспомогательные объекты
ksCMHideDimensions	40369	Скрыть\показать размеры
ksCMHideDesignations	40370	Скрыть\показать условные обозначения
ksCMCreateSheetFromModel	40373	Создать новый чертеж из модели
ksCMDeleteRollbackObjects	40375	Удалить все объекты под указателем
ksCMSelectedObjectProperties	40461	Свойства объекта (оси, плоскости, эскиза, операции, грани)

ksCMSelectedObjectOwnerProperties	40462	Свойства родителя (эскиза, операции)
ksCMSelectedCompanentProperties	40463	Свойства детали или сборки
ksCMSelectedCompanentInstanceProperties	40464	Свойства вставленного компонента
ksCMViewFeatureInTree	40524	Показать в дереве
ksCMLODOn	40610	Упрощенное отображение
ksCM3DArrayDestroy	40615	Разрушить массив компонентов
ksCMEditBilletPart	40621	Редактировать источник для операции деталь-заготовка
ksCMChangeBilletPart	40622	Заменить источник для операции деталь-заготовка
ksCMEmbodimentManager	40710	Редактор исполнений
ksCMAdditionNumberberManager	40711	Редактор дополнительных номеров исполнений
ksCMHideInCompCPlaces	40728	Скрыть\показать во вставках начала координат
ksCMHideInCompCPlanes	40729	Скрыть\показать во вставках конструктивные плоскости
ksCMHideInCompCAxies	40730	Скрыть\показать во вставках конструктивные оси
ksCMHideInCompSketches	40731	Скрыть\показать во вставках эскизы
ksCMHideInCompSurfaces	40732	Скрыть\показать во вставках поверхности
ksCMHideInCompThreads	40733	Скрыть\показать во вставках изображения резьбы
ksCMHideInCompCurves	40734	Скрыть\показать во вставках пространственные кривые
ksCMHideInCompPoints	40735	Скрыть\показать во вставках конструктивные точки
ksCMHideInCompAllObjects	40736	Скрыть\показать во вставках вспомогательные объекты
ksCMHideInCompDimensions	40737	Скрыть\показать во вставках размеры
ksCMHideInCompDesignations	40738	Скрыть\показать во вставках условные обозначения
ksCM3DSavePartAs	40744	Преобразование Компонент деталь – компонент сборка и наоборот
ksCM3DUnitParts	40745	Разрушить массив компонентов
ksCM3DAssemblyDestroy	40746	Разрушить подсборку
ksCMWireframeMode	41882	Каркас
ksCMHiddenRemovedMode	41883	Удаление невидимых линий
ksCMHiddenThinMode	41884	Невидимые линии тонкие
ksCMShadedMode	41885	Полутоновое
ksCMPerspective	41886	Перспективное отображение
ksCMShadedWireframeMode	41893	Полутоновое с каркасом
ksCMRotateCCW	41887	Вращать изображение против часовой стрелки

ksCMRotateCC	41888	Вращать изображение по часовой стрелке
ksCMRotate90CCW	41889	Вращать изображение против часовой стрелки на 90гр
ksCMRotate90CC	41890	Вращать изображение по часовой стрелке на 90гр
ksCMFastLines	41891	Быстрое отображение линий
ksCMTreeStructure	41904	Вариант состава дерева построения
ksCMPropertyEditor	45171	Редактор свойств
ksCMProcessBarVisible	46541	Скрыть/показать панель параметров

Команды работы со свойствами и отчетами

ksCMSetProperties	32541	Дополнительные свойства объектов
ksCMSetReportStyles	32542	Поднять окно редактирования стиля отчёта
ksEditDocumentProperties	37171	Процесс редактирования свойств документа
ksEditInserionFragmentProperties	37172	Процесс редактирования свойств вставки фрагмента
ksEditInserionViewProperties	37173	Процесс редактирования свойств вставки вида
ksEditMacroObjectProperties	37174	Процесс редактирования свойств макроэлемента

Стандартные команды меню из \VC98\MFC\Include\AFXRES.H

Команды меню Файл

ksCMFileNew	ID_FILE_NEW	Создать файл	(0xE100)
ksCMFileOpen	ID_FILE_OPEN	Открыть файл	(0xE101)
ksCMFileClose	ID_FILE_CLOSE	Закрыть файл	(0xE102)
ksCMFileSave	ID_FILE_SAVE	Сохранить файл	(0xE103)
ksCMFileCloseAs	ID_FILE_SAVE_AS	Сохранить файл как	(0xE104)
ksCMFilePrintSetup	ID_FILE_PRINT_SETUP	Настройки принтера	(0xE106)
ksCMFilePrint	ID_FILE_PRINT	Печать	(0xE107)
ksCMFilePrintPreview	ID_FILE_PRINT_PREVIEW	Предварительный просмотр	(0xE109)
ksCMFileSendMail	ID_FILE_SEND_MAIL	Отправить файл по электронной почте	(0xE10C)

Команды редактирования

ksCMEditClear	ID_EDIT_CLEAR	Удалить выделенные объекты	(0xE120)

ksCMEditClearAll	ID_EDIT_CLEAR_ALL	Удалить все объекты документа	(0xE121)
ksCMEditCopy	ID_EDIT_COPY	Копировать в буфер обмена	(0xE122)
ksCMEditCut	ID_EDIT_CUT	Вырезать в буфер обмана	(0xE123)
ksCMEditFind	ID_EDIT_FIND	Найти	(0xE124)
ksCMEditPaste	ID_EDIT_PASTE	Вставить из буфера обмена	(0xE125)
ksCMEditRepeat	ID_EDIT_REPEAT	Повторить поиск	(0xE128)
ksCMEditReplace	ID_EDIT_REPLACE	Заменить	(0xE129)
ksCMEditSelectAll	ID_EDIT_SELECT_ALL	Выделить все	(0xE12B)
ksCMEditUndo	ID_EDIT_UNDO	Отменить	(0xE12C)
ksCMEditRedo	ID_EDIT_REDO	Повторить	(0xE12C)

Команды меню Окно

ksCMWindowNew	ID_WINDOW_NEW	Новое окно	(0xE130)

Команды меню Помощь

ksCMAbout	ID_APP_ABOUT	Информация о приложении	(0xE140)
ksCMHelpIndex	ID_HELP_INDEX	Вызов справки	(0xE142)
ksCMHelpFinder	ID_HELP_FINDER	Вызов справки с поиском	(0xE143)
ksCMContextHelp	ID_CONTEXT_HELP	Вызов контекстной справки (shift-F1)	(0xE145)
ksCMHelp	ID_HELP	Получить справку в текущем контексте	(0xE146)

ksHideMessageEnum – Режим скрытия сообщений и диалогов

ksShowMessage	0	Показывать все сообщения и диалоги
ksHideMessageYes	1	Скрывать сообщения и диалоги с выбором **ОК** или **Да**, если сообщение или диалог предусматривают такой выбор, с перестроением документа.
ksHideMessageNo	2	Скрывать сообщения с **ОК**, если имеется только кнопка **ОК**, сообщения и диалоги с выбором **Нет**, если сообщение или диалог предусматривают такой выбор, без перестроения документа

Типы выхода из режима работы под управлением системы КОМПАС

scsSTOPPED_FOR_MENU_COMMAND	1	- выполнена команда меню "Остановить работу библиотеки"
scsSTOPPED_FOR_SYSTEM_STOP	0	- идет закрытие системы КОМПАС

scsSTOPPED_FOR_ITSELF	-1	- вызов функции SystemControlStop из-под библиотеки
scsSTOPPED_FOR_START_THIS_LIB	-2	- управление системе КОМПАС уже передано той же библиотекой,
scsSTOPPED_FOR_START_ANOTHER_LIB	-3	- управление системе КОМПАС уже передано другой библиотекой.

Типы системных папок

sptSYSTEM_FILES	0	- папка системных файлов
sptLIBS_FILES	1	- папка файлов библиотек
sptAPPS_FILES	1	- выдать путь на каталог файлов приложений
sptTEMP_FILES	2	- папка хранения временных файлов,
sptCONFIG_FILE	3	- папка хранения файлов конфигурации системы
sptINI_FILE	4	- папка хранения INI-файла системы.
sptBIN_FILE	5	- папка исполняемых файлов системы
sptPROJECT_FILES	6	- папка сохранения kompas.prj
sptDESKTOP_FILES	7	- папка сохранения kompas.dsk
sptTEMPLATES_FILES	8	- папка шаблонов КОМПАС-документов
sptPROFILES_FILES	9	- папка сохранения профилей пользователя
sptWORK_FILES	10	- рабочая папка
sptSHEETMETAL_FILES	11	- папка таблиц сгибов
sptPARTLIB_FILES	12	- папка PartLib
sptMULTILINE_FILES	13	- папка шаблонов мультилинии
sptPRINTDEVICE_FILES	14	- папка конфигураций плоттеров/принтеров

Последняя папка, использовавшаяся в диалогах OpenlSave для:

sptCURR_WORK_FILES	15	- открытия/сохранения файлов документов
sptCURR_LIBS_FILES	16	- подключения прикладных библиотек и библиотек документов
sptCURR_APPS_FILES	16	запоминание последних директорий, из которых выполнилось открытие/сохранение файла в диалоге OpenlSave
sptCURR_SYSTEM_FILES	17	- подключения библиотек стилей
sptCURR_PROFILES_FILES	18	- загрузки/сохранения профиля
sptCURR_SHEETMETAL_FILES	19	- загрузки таблиц сгибов
sptMULTY_APPS_FILES	20	выдать список каталогов файлов приложений
sptDOC_LIBS_FILES	21	выдать путь на каталог файлов библиотек документов
sptMULTY_DOC_LIBS_FILES	22	выдать список каталогов файлов библиотек документов
sptCUR_DOC_LIBS_FILES	23	запоминание последних директорий, с которых выполнилось открытие/сохранение файла в диалоге OpenlSave
sptUTILS_FILES	24	- выдать путь на каталог утилит
sptMULTY_UTILS_FILES	25	- выдать список каталогов утилит
sptCURR_UTILS_FILES	26	- запоминание последних директорий, из которых выполнилось открытие/сохранение файла в диалоге OpenlSave

sptPROGRAMDATA_FILES	27	- Выдать путь на каталог данных приложения (C:\ProgramData\ASCON\KOMPAS-3D\18)

Коды типов сортаментов

Черные металлы	1	Фасонный
	2	Листовой
	3	Ленты
	4	Проволока
	5	Профильный прокат
	6	Трубы
	7	Сетки
Цветные металлы	8	Прутки
	9	Трубы
	10	Проволока
	11	Листы
	12	Плиты
	13	Ленты
	14	Фольги
	15	Профили
	16	Сетки
Пластмассы	17	Пленки
	18	Трубы
	19	Плиты
	20	Ленты
	21	Листы
	22	Трубки
	23	Прутки
Резины и кожи	24	Листы
	25	Трубки
	26	Прутки
Стекло и минералы	27	Листы
	28	Трубки
	29	Прутки
Материалы разные	30	Пиломатериалы
	31	Бумаги разные
	32	Стройматериалы
	33	Текстиль

Типы валидатора

1	- char
2	- int
3	- long
4	- float
5	- double

Типы динамических массивов

Для automation	Для API экспортных функций

CHAR_STR_ARR В зависимости от типа компиляции..	1	- динамический массив указателей на интерфейсы ksChar255	- динамический массив указателей на строки символов
POINT_ARR	2	- динамический массив указателей на интерфейсы ksMathPointParam	- динамический массив указателей на математические точки (структура MathPointParam)
CURVE_PATTERN_ARR	2	- динамический массив указателей на интерфейсы ksCurvePattern	- динамический массив указателей на участки штриховой линии (структура CurvePattern)
TEXT_LINE_ARR	3	- динамический массив указателей на интерфейсы ksTextLineParam	- динамический массив строк текста (структура TextLineParam)
TEXT_ITEM_ARR	4	- динамический массив указателей на интерфейсы ksTextItemParam	- динамический массив компонент строк текста (структура TextItemParam)
ATTR_COLUMN_ARR	5	- динамический массив указателей на интерфейсы ksColumnInfoParam	- динамический массив колонок атрибутов (структура ColumnInfo)
USER_ARR	6	- динамический пользовательский массив	- динамический пользовательский массив
POLYLINE_ARR	7	- динамический массив указателей на интерфейсы ksDynamicArray типа POINT_ARR	- динамический массив полилиний (указателей массивов POINT_ARR)
RECT_ARR	8	- динамический массив указателей на интерфейсы ksRectParam	- динамический массив габаритных прямоугольников (структура RectParam)
LIBRARY_STYLE_ARR	9	- динамический массив указателей на интерфейсы ksLibraryStyleParam	- динамический массив структур параметров для стиля в библиотеке стилей(LibraryStyleParam)
VARIABLE_ARR	10	- динамический массив указателей на интерфейсы ksVariable	- динамический массив структур параметров параметрических переменных (VariableParam)

CURVE_PATTERN_ARR_EX	11	- динамический массив указателей на интерфейсы ksCurvePattern	- динамический массив указателей на участки штриховой линии (структура CurvePatternEx)
LIBRARY_ATTR_TYPE_ARR	12	- динамический массив указателей на интерфейсы ksLibraryAttrTypeParam	- динамический массив структур параметров для типа атрибута в библиотеке типов атрибутов (LibraryAttrTypeParam)
NURBS_POINT_ARR	13	- динамический массив указателей на интерфейсы ksNurbsPointParam	- динамический массив структур точек кривой NURBS (NurbsPointParam)
DOUBLE_ARR	14	- динамический массив указателей на интерфейсы ksDoubleValue	- динамический массив значений типа double
CONSTRAINT_ARR	15	- динамический массив указателей на интерфейсы ksConstraintParam	- динамический массив структур параметрических ограничений (ConstraintParam)
CORNER_ARR	16	- динамический массив указателей на интерфейсы ksCornerParam	- динамический массив структур параметров скругленных (или усеченных) углов прямоугольников и многоугольников (CornerParam) Пример работы со структурой CornerParam и массивом CORNER_ARR
DOC_SPCOBJ_ARR	17	- динамический массив указателей на интерфейсы ksDocAttachedSpcParam	- динамический массив структур параметров документов, подключенных к объекту спецификации (DocAttachedSpcParam)
SPCSUBSECTION_ARR	18	- динамический массив указателей на интерфейсы ksSpcSubSectionParam	- динамический массив структур параметров подраздела спецификации (SpcSubSectionParam)

SPCTUNINGSEC_ARR	19	- динамический массив указателей на интерфейсы ksSpcTuningSectionParam	- динамический массив структур параметров настройки раздела спецификации (SpcTuningSectionParam)
SPCSTYLECOLUMN_ARR	20	- динамический массив указателей на интерфейсы ksSpcStyleColumnParam	- динамический массив структур параметров стиля колонки спецификации (SpcStyleColumnParam)
SPCSTYLESEC_ARR	21	- динамический массив указателей на интерфейсы ksSpcStyleSectionParam	- динамический массив структур параметров стиля раздела спецификации (SpcStyleSectionParam)
QUALITYITEM_ARR	22	- динамический массив указателей на интерфейсы ksQualityItemParam	динамический массив структур параметров интервала квалитета (QualityItemParam)
LTVARIANT_ARR	23	- динамический массив указателей на интерфейсы ksLtVariant	- динамический массив структур для хранения данных некоторого типа (LtVariant)
TOLERANCEBRANCH_ARR	24	- динамический массив указателей на интерфейсы ksToleranceBranch	- динамический массив структур параметров "опоры" допуска формы (ToleranceBranch)
HATCHLINE_ARR	25	- динамический массив указателей на интерфейсы ksHatchLineParam	- динамический массив структур (HatchLineParam)
TREENODEPARAM_ARR	26	- динамический массив указателей на интерфейсы ksTreeNodeParam	- динамический массив структур узла дерева TreeNodeParam
CHAR_STR_ARR_W В зависимости от типа компиляции...	27		- динамический массив указателей на строки символов. Используется в Unicode-вских структурах и функциях.

Типы стилей

CURVE_STYLE	1	- стиль кривых	ksCurveStyleParam
HATCH_STYLE	2	- стиль штриховок	ksHatchStyleParam
TEXT_STYLE	3	- стиль текста	ksTextStyleParam

STAMP_STYLE	4	- тип основной надписи	В настоящее время не реализовано
CURVE_STYLE_EX	5	- расширенный стиль кривых	ksCurveStyleParam

Типы настроек

DIMENTION_OPTIONS	1	Настройки размера
SNAP_OPTIONS	1	Настройки привязок
ARROWFILLING_OPTIONS	2	Настройки зачернения стрелок
SHEET_OPTIONS	3	Настройка параметров оформления листа документа для новых документов
SHEET_OPTIONS_EX	4	Настройка параметров листа документа
LENGTHUNITS_OPTIONS	5	Настройки единиц измерений
SNAP_OPTIONS_EX	6	Настройки привязок документа
VIEWCOLOR_OPTIONS	7	Настройки цвета фона рабочего поля 2d - документов
TEXTEDIT_VIEWCOLOR_OPTIONS	8	Настройки цвета фона редактирования текста
MODEL_VIEWCOLOR_OPTIONS	9	Настройки цвета фона для моделей
OVERLAP_OBJECT_OPTIONS	10	Настройки перекрывающихся объектов
DIMENTION_OPTIONS_EX	11	Настройки размера

Типы библиотек стилей

Параметр libraryType может иметь следующие значения для выбора типа библиотеки стиля:

CURVE_STYLE_LIBRARY	1	- библиотека стилей кривых (*.lcs)
HATCH_STYLE_LIBRARY	2	- библиотека стилей штриховок (*.lhs)
TEXT_STYLE_LIBRARY	3	- библиотека стилей текстов (*.lts)
STAMP_LAYOUT_STYLE_LIBRARY	4	- библиотека основных надписей (*.lyt)
GRAPHIC_LAYOUT_STYLE_LIBRARY	5	- библиотека оформлений графических документов (*.lyt)

TEXT_LAYOUT_STYLE_LIBRARY	6	- библиотека оформлений текстовых документов (*.lyt)
SPC_LAYOUT_STYLE_LIBRARY	7	- библиотека стилей спецификаций (*.lyt)

Размерности и типы тел

Параметр bitVector может иметь следующие значения для выбора размерности и типа тела при расчете МЦХ:

ST_MIX_MM	0x1	- миллиметры
ST_MIX_SM	0	- сантиметры
ST_MIX_DM	0x2	- дециметры
ST_MIX_M	0x3	- метры
ST_MIX_GR	0	- граммы
ST_MIX_KG	0x10	- килограммы
ST_MIX_EXT	0	- тело выдавливания
ST_MIX_RV	0x20	- тело вращения

Размерности длины

ST_MIX_MM	0x1	- миллиметры
ST_MIX_SM	0	- сантиметры
ST_MIX_DM	0x2	- дециметры
ST_MIX_M	0x3	- метры

Типы параметрических ограничений

Параметр constrType может иметь следующие значения для выбора типа параметрического ограничения:

CONSTRAINT_FIXED_POINT	1	- фиксация точки
CONSTRAINT_POINT_ON_CURVE	2	- точка на кривой
CONSTRAINT_HORIZONTAL	3	- горизонталь
CONSTRAINT_VERTICAL	4	- вертикаль
CONSTRAINT_PARALLEL	5	- параллельность двух прямых или отрезков
CONSTRAINT_PERPENDICULAR	6	- перпендикулярность двух прямых или отрезков
CONSTRAINT_EQUAL_LENGTH	7	- равенство длин двух отрезков
CONSTRAINT_EQUAL_RADIUS	8	- равенство радиусов двух дуг или окружностей
CONSTRAINT_HOR_ALIGN_POINTS	9	- выравнивание двух точек по горизонтали
CONSTRAINT_VER_ALIGN_POINTS	10	- выравнивание двух точек по вертикали
CONSTRAINT_MERGE_POINTS	11	- совпадение двух точек
CONSTRAINT_TANGENT_TWO_CURVES	15	- коллинеарность отрезков

CONSTRAINT_SYMMETRY_TWO_POINTS	16	- симметрия двух точек относительно отрезка
CONSTRAINT_COLLINEAR	17	коллинеарность двух отрезков
CONSTRAINT_FIXED_ANGLE	18	- фиксированный угол
CONSTRAINT_FIXED_LENGHT	19	- фиксированная длина
CONSTRAINT_POINT_ON_CURVE_MIDDLE	20	- точка на середине кривой
CONSTRAINT_BISECTOR	21	- биссектриса
CONSTRAINT_CONCENTRICITY	22	- совпадение центров окружностей, дуг, эллипсов и точек

Примечание.

При использовании API 7 аналогом данного набора является перечисление ksConstraintTypeEnum

Типы элементов массива LTVariant

ltv_Char	1	- символ
ltv_Uchar	2	- байт
ltv_Int	3	- целое
ltv_Uint	4	- беззнаковое целое
ltv_Long	5	- длинное целое
ltv_Float	6	- вещественное
ltv_Double	7	- двойное вещественное
ltv_Str	8	- строка 255 символов
ltv_Short	10	- короткое целое

Типы пути в библиотеке моделей

0	- отказ от выбора пути
1	- "корень" библиотеки
2	- папка библиотеки
3	- трехмерная модель в библиотеке

Типы объектов графического документа; соответствие интерфейсов API5 и API7

В таблице представлены типы графических объектов и соответствующие им интерфейсы API5 и API7.

Название объекта	Идентификатор объекта	DrawingObjectTypeEnum	Старый тип	Тип параметров	Структура	Интерфейс API5	Интерфейс API7

Неизвестный объект	ksUnknown	-1					
Все объекты	ksAllObj	0	ALL_OBJ				
Отрезок	ksDrLineSeg	1	LINESEG_OBJ	ALL PARAM	LineSegParam	ksLineSegParam	ILineSegment
Окружность	ksDrCircle	2	CIRCLE_OBJ	ALL PARAM	CircleParam	ksCircleParam	ICircle
Дуга	ksDrArc	3	ARC_OBJ	ALL PARAM	ArcParam	ksArcByAngleParam	IArc
				POINT_ARC_PARAM	ArcParam1	ksArcByPointParam	IArc
				ANGLE_ARC_SHEET_PARAM	ArcParam	ksArcByAngleParam	IArc
				ANGLE_ARC_VIEW_PARAM	ArcParam	ksArcByAngleParam	IArc
				POINT_ARC_SHEET_PARAM	ArcParam1	ksArcByPointParam	IArc
				POINT_ARC_VIEW_PARAM	ArcParam1	ksArcByPointParam	IArc

Текст на чертеже	ksDrDrawText	4	TEXT_OBJ	ALL PARAM	TextParam	ksTextParam	IDrawingText IText
				Индекс строки текста (начиная с 0)	TextLineParam	ksTextLineParam	ITextLine
Точка	ksDrPoint	5	POINT_OBJ	ALL PARAM	PointParam	ksPointParam	IPoint
Штриховка	ksDrHatch	7	HATCH_OBJ	ALL PARAM	HatchParam	ksHatchParam	IHatch IHatchParam
				HATCH_PARAM_EX	HatchParam	ksHatchParam	Hatch IHatchParam
Кривая Безье, сплайн	ksDrBezier	8	BEZIER_OBJ	ALL PARAM	BezierParam	ksBezierParam	IBezier
Линейны				Индекс узл кривой (начиная с 0)	BezierPointParam	ksBezierPointParam	IBezier::GetPoint
Линейный размер	ksDrLDimension	9	LDIMENSION_OBJ	ALL PARAM	LdimParam	ksLDimParam	ILineDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText

				DIM_SOURSE_PARAM	LDimSource	ksLDimSourceParam	ILineDimension
				DIM_DRAW_PARAM	DimDrawing	ksDimDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
угловой размер	ksDrADimension	10	ADIMENSION_OBJ	ALL PARAM	AdimParam	ksADimParam	IArcDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	ADimSource	ksADimSourceParam	IArcDimension
				DIM_DRAW_PARAM	DimDrawing	ksDimDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	

				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Диаметральный размер	ksDrDDimension	13	DDIMENSION_OBJ	ALL PARAM	RdimParam	ksRDimParam	IDiametralDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	RDimSource	ksRDimSourceParam	IDiametralDimension
				DIM_DRAW_PARAM	RDimDrawing	ksRDimDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Радиальный размер	ksDrRDimension	14	RDIMENSION_OBJ	ALL PARAM	RdimParam	ksRDimParam	IRadialDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText

				DIM_SOURSE_PARAM	RDimSource	ksRDimSourceParam	IRadialDimension
				DIM_DRAW_PARAM	RDimDrawing	ksRDimDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Радиальный размер с изломом	ksDrRBreakDimension	15	RBREAKDIMENSION_OBJ	ALL PARAM	RbreakDimParam	ksRBreakDimParam	IBreakRadiaIDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	RDimSource	ksRDimSourceParam	IBreakRadiaIDimension
				DIM_DRAW_PARAM	RBreakDrawing	ksRBreakDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	

				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Шероховатость	ksDrRough	16	ROUGH_OBJ	ALL PARAM	RoughParam	ksRoughParam	IRough
База	ksDrBase	17	BASE_OBJ	ALL PARAM	BaseParam	ksBaseParam	IBase
Стрелка направления взгляда	ksDrWPointer	18	WPOINTER_OBJ	ALL PARAM	ViewPointerParam	ksViewPointerParam	IViewPointer
Линия разреза	ksDrCut	19	CUT_OBJ	ALL PARAM	CutLineParam	ksCutLineParam	ICutLine
Простая линия выноска	ksDrLeader	20	LEADER_OBJ	ALL PARAM	LeaderParam	ksLeaderParam	ILeader
Линия выноска для обозначения позиции	ksDrPosLeader	21	POSLEADER_OBJ	ALL PARAMa	PosLeaderParam	ksPosLeaderParam	IPositionLeader
Линия выноска для обозначения клеймения	ksDrBrandLeader	22	BRANDLEADER_OBJ	ALL PARAM	BrandLeaderParam	ksBrandLeaderParam	IBrandLeader
Линия выноска для обозначения маркирования	ksDrMarkerLeader	23	MARKERLEADER_OBJ	ALL PARAM	MarkerLeaderParam	ksMarkerLeaderParam	IMarkLeader
Допуск формы	ksDrTolerance	24	TOLERANCE_OBJ	ALL PARAM	ksTolerancePar	ksToleranceParam	ITolerance
Таблица	ksDrTable	25	TABLE_OBJ	ALL PARAM	нереализовано	нереализовано	IDrawingTable ITable
Контур	ksDrContour	26	CONTOUR_OBJ	ALL PARAM	short (стиль)	ksContourParam	IDrawingContour IContour
Нетипизированный макроэлемент	ksDrMacro	27	MACRO_OBJ	ALL PARAM	нереализовано	нереализовано	IMacroObject

Линия	ksDrLine	28	LINE_OBJ	ALL PARAM	LineParam	ksLineParam	ILine
Слой	ksLayer	29	LAYER_OBJ	ALL PARAM	LayerParam	ksLayerParam	ILayer
				VIEW_LAYER_STATE	int (состояние слоя)	ksLtVariant(состояние слоя)	ILayer
Вставленный фрагмент	ksDrFragment	30	FRAGMENT_OBJ	ALL PARAM	InsertFragmentParamEx	ksInsertFragmentParam	IInsertionFragment
Полилиния	ksDrPolyline	31	POLYLINE_OBJ	ALL PARAM	PolylineParamEx	ksPolylineParam	IPolyline
Эллипс	ksDrEllipse	32	ELLIPSE_OBJ	ALL PARAM	EllipseParam	ksEllipseParam	IEllipse
NURBS-кривая по полюсам	ksDrNurbs	33	NURBS_OBJ	ALL PARAM	NurbsParam	ksNurbsParam	INurbs
				NURBS_CLAMPED_PARAM	NurbsParam	ksNurbsParam	
Дуга эллипса	ksDrEllipseArc	34	ELLIPSE_ARC_OBJ	ALL PARAM	EllipseArcParam	ksEllipseArcParam	IEllipseArc
				POINT_ARC_PARAM	EllipseArcParam1	ksEllipseArcParam1	IEllipseArc
Прямоугольник	ksDrRectangle	35	RECTANGLE_OBJ	ALL PARAM	RectangleParam	ksRectangleParam	IRectangle
Многоугольник	ksDrRegularPolygon	36	REGULARPOLYGON_OBJ	ALL PARAM	RegularPolygonParam	ksRegularPolygonParam	IRegularPolygon
Эквидистанта	ksDrEquid	37	EQUID_OBJ	ALL PARAM	EquidistantParam	ksEquidistantParam	IEquidistant

Линейный размер с обрывом	ksDrLBreakDimension	38	LBREAKDIMENSION_OBJ	ALL_PARAM	LbreakDimParam	ksLBreakDimParam	IBreakLineDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	LBreakDimSource	ksLBreakDimSource	IBreakLineDimension
				DIM_DRAW_PARAM	BreakDimDrawing	ksBreakDimDrawing	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Угловой размер с обрывом	ksDrABreakDimension	39	ABREAKDIMENSION_OBJ	ALL_PARAM	AbreakDimParam	ksABreakDimParam	IBreakAngleDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	ADimSource	ksADimSourceParam	IBreakAngleDimension

				DIM_DRAW_PARAM	BreakDimDrawing	ksBreakDimDrawing	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Размер высоты	ksDrOrdinateDimension	40	ORDINATEDIMENSION_OBJ	ALL PARAM	OrdinatedDimParam	ksOrdinatedDimParam	IHeightDimension
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	OrdinatedSource	ksOrdinatedSourceParam	IHeightDimension
				DIM_DRAW_PARAM	OrdinatedDrawing	ksOrdinatedDrawingParam	IDimensionParams
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue

Фоновая заливка цветом	ksDrColorFill	41	COLORFILL_OBJ	ALL PARAM	long (цвет)	ksLtVariant (цвет)	IColouring
Обозначение центра	ksDrCentreMarker	42	CENTREMARKER_OBJ	ALL PARAM	CentreParam	ksCentreParam	ICentreMarker IAxisLineParam
Размер длины дуги	ksDrArcDimension	43	ARCDIMENSION_OBJ	ALL PARAM	нереализовано	нереализовано	IArcDimensiono
				DIM_TEXT_PARAM	DimText	ksDimTextParam	IDimensionText
				DIM_SOURSE_PARAM	нереализовано	нереализовано	IArcDimensiono
				DIM_DRAW_PARAM	нереализовано	нереализовано	IDimensionParamso
				DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				SHEET_DIM_PARTS	DimensionPartsParam	ksDimensionPartsParam	
				DIM_VALUE	double	ksDoubleValue	IDimensionText::NominalValue
Объект спецификации	не 2D объект	44	SPC_OBJ	ALL PARAM	SpcObjParam	ksSpcObjParam	ISpecificationObject ISpecificationBaseObject ISpecificationCommentObject
Растровый объект	ksDrRaster	45	RASTER_OBJ	ALL PARAM	RasterParam	ksRasterParam	IRaster

Обозначение изменения	ksDrChangeLeader	46	CHANGE_LEADER_OBJ	ALL PARAM	ChangeLeaderParam	ksChangeLeaderParam	IChangeLeader
Выносной элемент	ksDrRemoteElement	47	REMOTE_ELEMENT_OBJ	ALL PARAM	RemoteElementParam	ksRemoteElementParam	IRemoteElement
Осевая линия	ksDrAxisLine	48	AXISLINE_OBJ	ALL PARAM	AxisLineParam	ksAxisLineParam	IAxisLine IAxisLineParam
Вставка OLE объекта	ksDrOLEObject	49	OLEOBJECT_OBJ	ALL PARAM	не реализовано	не реализовано	IOleDrawingObject
Номер узла	ksDrUnitNumber	50	KNOTNUMBER_OBJ	ALL PARAM	не реализовано	не реализовано	IUnitMarking
Фигурная скобка	ksDrBrace	51	BRACE_OBJ	ALL PARAM	не реализовано	не реализовано	IBrace
Марка/Марка/позиционное обозначение с линией-выноской	ksDrMarkOnLeader	52	POSNUM_OBJ	ALL PARAM	не реализовано	не реализовано	IMarkOnLeader
Марка/позиционное обозначение на линии	ksDrMarkOnLine	53	MARKONLDR_OBJ	ALL PARAM	не реализовано	не реализовано	IMarkOnLine
Марка/позиционное обозначение без линии-выноски	ksDrMarkInsideForm	54	MARKWOLDR_OBJ	ALL PARAM	не реализовано	не реализовано	IMarkInsideForm
Волнистая линияВолнистая	ksDrWaveLine	55	WAVELINE_OBJ	ALL PARAM	не реализовано	не реализовано	IWaveLine
Прямая ось	ksDrStraightAxis	56	DIRAXIS_OBJ	ALL PARAM	не реализовано	не реализовано	IStraightAxis
Линия обрыва с изломами	ksDrBrokenLine	57	BROKENLINE_OBJ	ALL PARAM	не реализовано	не реализовано	IBrokenLine
Круговая ось	ksDrCircleAxis	58	CIRCLEAXIS_OBJ	ALL PARAM	не реализовано	не реализовано	ICircleAxis
Дуговая ось	ksDrArcAxis	59	ARCAXIS_OBJ	ALL PARAM	не реализовано	не реализовано	IArcAxis

Обозначение узла в сечении	ksDrCutUnitMarking	60	CUTUNITMARKING	ALLPARAM	не реализовано	не реализовано	ICutUnitMarking
Обозначение узла	ksDrUnitMarking	61	UNITMARKING	ALLPARAM	не реализовано	не реализовано	IUnitMarking
Выносная надпись к многослойным конструкциям	ksDrMultiTextLeade	62	MULTITEXTLEADER	ALLPARAM	не реализовано	не реализовано	IMultiTextLeade
Вставка внешнего вида	ksDrExternalView	63	EXTERNALVIEW_OBJ	ALLPARAM	не реализовано	не реализовано	IInsertionView
Аннотационный отрезок	ksDrAnnLineSeg	64	ANNLINESEG_OBJ	ALLPARAM	LineSegParam	ksLineSegParam	ILineSegment IAnnotativeObject
Аннотационная окружность	ksDrAnnCircle	65	ANNCIRCLE_OBJ	ALLPARAM	CircleParam	ksCircleParam	ICircle IAnnotativeObject
Аннотационный эллипс	ksDrAnnEllipse	66	ANNELLIPSE_OBJ	ALLPARAM	EllipseParam	ksEllipseParam	IEllipse IAnnotativeObject
Аннотационная дуга	ksDrAnnArc	67	ANNARC_OBJ	ALLPARAM	ArcParam	ksArcByAngleParam	IArc IAnnotativeObject
Аннотационная дуга эллипса	ksDrAnnEllipseArc	68	ANNELLIPSE_ARC_OBJ	ALLPARAM	EllipseArcParam	ksEllipseArcParam	IEllipseArc IAnnotativeObject
Аннотационная полилиния	ksDrAnnPolyline	69	ANNPOLYLINE_OBJ	ALLPARAM	PolylineParamEx	ksPolylineParam	IPolyline IAnnotativeObject
Аннотационная точка	ksDrAnnPoint	70	ANNPOINT_OBJ	ALLPARAM	PointParam	ksPointParam	IPoint IAnnotativeObject
Текст с аннатационной точкой привязки	ksDrAnnText	71	ANNTEXT_OBJ	ALLPARAM	TextParam	ksTextParam	IDrawingText IText IAnnotativeObject
Мультилиния	ksDrMultiLine	72	MULTILINE_OBJ	ALLPARAM	не реализовано	не реализовано	IMultiline
Линия разреза/ сечения для СПДС	ksDrBuildingCutLine	73	BUILDINGCUTLINE_OBJ	ALLPARAM	CutLineParam	ksCutLineParam	ICutLine

Присоединенная линия выноски	ksDrAttachedLeader	74	ATTACHED_LEADER_OBJ	ALL PARAM	LeaderParam	ksLeaderParam	ILeader
Условное пересечение	ksDrConditionCrossing	75	CONDITIONCROSSING_OBJ	ALL PARAM	не реализовано	не реализовано	IConditionCrossing
Ассоциативная таблица отчета	ksReportTable	76	REPORTTABLE_OBJ	ALL PARAM	не реализовано	не реализовано	IAssociationTable
Таблица исполнений	ksEmbodimentsTable	77	EMBODIMENTSTABLE_OBJ	ALL PARAM	не реализовано	не реализовано	IAssociationTable
Кривая общего вида (проекционная кривая)		78	SPECIALCURVE_OBJ	ALL PARAM	не реализовано	не реализовано	IDrawingObject IDrawingObject1
Таблица параметров массива	ksArrayParamTable	79	ARRAYPARAMTABLE_OBJ	ALL PARAM	не реализовано	не реализовано	IAssociationTable
NURBS-кривая по точкам	ksDrNurbsByPoints	80	NURBS_BY_POINTS_OBJ	ALL PARAM	не реализовано	не реализовано	INurbsByPoints INurbs
Коническая кривая	ksDrConicCurve	81	CONIC_CURVE_OBJ	ALL PARAM	не реализовано	не реализовано	IConicCurve
Круговая сетка центров	ksDrCircularCentres	84	CIRCULAR_CENTRES_OBJ	ALL PARAM	не реализовано	не реализовано	ICircularCentres
Линейная сетка центров	ksDrLinearCentres	85	LINEAR_CENTRES_OBJ	ALL PARAM	не реализовано	не реализовано	ILinearCentres
Дуговая осевая линия	ksDrEllipseArcAxis	86	ELLIPSE_ARC_AXIS_OBJ	ALL PARAM	не реализовано	не реализовано	не реализовано
Спецификация на листе	не объект 2D	121	SPECIFICATION_OBJ	ALL PARAM	не реализовано	не реализовано	ISpecificationDescription::ShowOnSheet
неуказанная шероховатость	не объект 2D	122	SPECROUGH_OBJ	ALL PARAM	не реализовано	не реализовано	ISpecRough
Вид	ksView	123	VIEW_OBJ	ALL PARAM	ViewParam	ksViewParam	IView

Не объекты 2D

графический документ (чертеж или фрагмент)	124	DOCUMENT_OBJ	ALL PARAM	DocumentParam	ksDocumentParam	IKompasDocument2D IKompasDocument2D1 IFragmentDocument IDrawingDocument
технические требования	125	TECHNICALDEMAND_OBJ	ALL PARAM или TECHNICAL_DEMAND_PAR 0, 1, 2... TT_FIRST_STR reference (мас-сив строк TEXT_LINE_ARR)	TechnicalDemandParam ksDynamicArray (массив строк TEXT_LINE_ARR) **Примечание:** Массив составляют только строки, входящие в страницу (габаритный прямоугольник) технических требований с номером, заданным в TECHNICAL_DEMAND_PAR.	ksTechnicalDemandParam	ITechnicalDemand
Штамп	126	STAMP_OBJ	ALL PARAM	нереализовано	нереализовано? можно использовать ksStamp	IStamp

Группа селектирования	127	SELECT_GROUP_OBJ	ALL PARAM	нереализовано	нереализовано	ISelectionManager
Именованная группа	128	NAME_GROUP_OBJ	ALL PARAM	нереализовано	нереализовано	IDrawingGroup
Рабочая группа	129	WORK_GROUP_OBJ	ALL PARAM	нереализовано	нереализовано	IDrawingGroup
Документ-спецификация	130	SPC_DOCUMENT_OBJ	ALL PARAM	DocumentParam	ksDocumentParam	ISpecificationDocument
			DOCUMENT_SIZE	SheetSize	ksSheetSize	ILayoutSheet ISheetFormat
			DOCUMENT_STATE	int (состояние документа)	ksLtVariant (состояние документа) можно использовать ksSpcDocument	IKompasDocument::Active
Документ-модель (деталь или сборка).	131	D3_DOCUMENT_OBJ	ALL PARAM	DocumentParam	ksDocumentParam	IKompasDocument3D IKompasDocument3D1 IPartDocument IAssemblyDocument
			DOCUMENT_STATE	int (состояние документа)	ksLtVariant (состояние документа), можно использовать IDocument3D ksDocument3D	IKompasDocument::Active
Таблица изменений	132	CHANGE_LIST_OBJ	ALL PARAM	нереализовано	нереализовано	IDrawingDocument::ChangeListDescriptions
Текстовый документ	133	TXT_DOCUMENT_OBJ	ALL PARAM	TextDocumentParam	ksTextDocumentParam	ITextDocument
Все документы	134	ALL_DOCUMENTS	константа используется при создании итератора по документам			IApplication::Documents

Верхняя граница типов поиска	134	MAX_TIP_SEARCH	не используется
Все объекты, которые могут входить в вид в порядке отрисовки	-1000	ALL_OBJ_SHOW_ORDER	константа используется при создании итератора по объектам вида

Примечание:

Для SHEET_ALLPARAM используются те же структуры, что и для ALLPARAM.

ksObjectPropertyControlTypeEnum – Типы контролов для отображения свойств в окне свойств

Константа		Тип контрола	Используемые поля в структуре PropertyParam кроме propertyType, propertyId, propertyInstance, isDefCpyProp, enable, и emptyValue
ksOPControlGroup	0	Группа свойств	-
ksOPControlPointCoord	1	Координаты точки	propertyValue - Координата X VT_R8 additionData - Координата Y VT_R8
ksOPControlPointStyle	2	Стиль точки 2D	propertyValue - Стиль точки из перечисления ksAnnotationSymbolEnum VT_I4
ksOPControlLineStyle	4	Стиль линии 2D	propertyValue - Стиль линии из перечисления ksCurveStyleEnum или пользовательский VT_I4
ksOPControlHatchStyle	5	Стиль штриховки	propertyValue - Стиль штриховки из перечисления ksHatchStyleEnum или пользовательский VT_I4
ksOPControlColor	6	Цвет	propertyValue - Цвет VT_I4
ksOPControlEditDouble	7	Вещественное значение	propertyValue - Значение VT_R8 propertyMinValue - Минимальное значение propertyMaxValue - Максимальное значение
ksOPControlEditInt	10	Целое значение	propertyValue - Значение VT_I4 propertyMinValue - Минимальное значение propertyMaxValue - Максимальное значение

ksOPControlListDouble	11	Список вещественных значений	propertyValue - Значение VT_R8 propertyMinValue - Минимальное значение propertyMaxValue - Максимальное значение additionData - Список значений VT_ARRAY | VT_R8 или VT_EMPTY для фиксированного списка загружаемого из ресурсов
ksOPControlListInt	12	Список целых значений	propertyValue - Значение VT_I4 propertyMinValue - Минимальное значение propertyMaxValue - Максимальное значение additionData - Список значений VT_ARRAY | VT_I4 или VT_EMPTY для фиксированного списка загружаемого из ресурсов
ksOPControlListCheck	15	Элемент который принимает два значения true, false	propertyValue - Значение VT_BOOL
ksOPControlListBmp	16	Список с битмапами	propertyValue - Значение VT_I4 Идентификатор текущего битмапа
ksOPControlListString	17	Список строк	propertyValue - Значение VT_BSTR additionData - Список значений VT_ARRAY | VT_BSTR или VT_EMPTY для фиксированного списка загружаемого из ресурсов
ksOPControlSimpleText	19	Однострочный текст объекта	
ksOPControlText	25	Многострочный текст	
ksOPControlEdit	31	Строка	propertyValue - Значение VT_BSTR
ksOPControlExternalEdit	34	Внешнее свойство с внешним редактированием. Отображает битмап	propertyValue - Значение любой простой тип Variant-а (Не отображается. Используется для сравнения значений) additionData - Значение любой тип (Не отображается. Не используется) propertyValue и additionData библиотека
ksOPControlExternalStringEdit	35	Внешнее свойство с внешним редактированием. Отображает строку	propertyValue - Значение VT_BSTR
ksOPControlFontImageList	38	Список со шрифтовыми иконками	

ksOPControlExternalText	44	Однострочный текст объекта с внешним редактированием
ksOPControlExternalMulty Text	45	Многострочный текст с внешним редактированием
ksOPControlUserLineStyle	47	Пользовательский стиль линии

ksHotPointEnum –Типы горячих точек

ksHPDefault	-1	
ksHPNormal	0	- обычный
ksHPSmall	1	- маленький
ksHPRing	2	- изменение угла поворота
ksHPBiDirArrow	3	- изменение длины
ksHPMiddlepoint	4	- средняя точка
ksHPTriangleDisplaced	5	- смещенный треугольник
ksHPVisibilityOn	6	- глаз - видимый объект
ksHPVisibilityOff	7	перечеркнутый глаз - скрытый объект
ksHPTilt	12	наклон

Прочие

3D

D3FormatConvType – Определения для конвертации в дополнительные форматы jgs, sat, xt, step, stl, VRML,C3D

format_SAT	1	формат SAT
format_XT	2	формат XT
format_STEP	3	формат STEP
format_IGES	4	формат IGES
format_VRML	5	формат VRML
format_STL	6	формат STL
format_JT	8	формат JT
load_format_SAT	-1	формат SAT, для открытия документов
load_format_XT	-2	формат XT, для открытия документов
load_format_STEP	-3	формат STEP, для открытия документов
load_format_IGES	-4	формат IGES, для открытия документов
load_format_STL	-6	формат STL для открытия документов

load_format_C3D	-7	формат C3D, для открытия документов математического ядра
load_format_JT	-8	формат JT, для открытия документов
format_STEP_AP203	203	формат STEP AP203. Прикладной протокол 203 (Проектирование с управляемой конфигурацией)
format_STEP_AP214	214	формат STEP AP214. Прикладной протокол 214 (Проектирование автомобилей)
format_STEP_AP242	242	формат STEP AP242.Прикладной протокол 242 (Проектирование автомобилей)

При использовании константы format_STEP сохранение выполняется в формат STEP AP203

Positioner_Type – Тип перемещения

pnMove	0	сдвиг
pnRotate	1	вращение

PartType – Типы компонентов

pInPlace_Part	-4	компонент, редактируемый на месте
pNew_Part	-3	новый компонент
pEdit_Part	-2	редактируемый компонент
pTop_Part	-1	главный компонент, в составе которого находится новый или редактируемый, или указанный компонент (например, сборка, в составе которой находится редактируемая деталь)

MateConstraintType – Типы сопряжений

mc_Coincidence	0	совпадение объектов
mc_Parallel	1	параллельность
mc_Perpendicular	2	перпендикулярность
mc_Tangency	3	касательность
mc_Concentric	4	концентричность
mc_Distance	5	постоянное расстояние между объектами
mc_Angle	6	постоянный угол между объектами

mc_InPlace	7	создание компонента "на месте" (эквивалентно совпадению системы координат плоскости, на которой создается компонент, и системы координат плоскости первого эскиза этого компонента)
mc_Transmission	9	Механическая передача
mc_CamGear	10	Кулачковый механизм. Кулачек-толкатель
mc_Symmetric	11	Симметрия
mc_Dependent	14	Зависимое положение

ksMateFixedTypeEnum – Фиксация компонентов при создании сопряжения

Неопределено	ksMFixedUnknown = 0
Фиксировать первый компонент	ksMFixedPart1 = 1
Фиксировать второй компонент	ksMFixedPart2 = 2

ksTypeLookStyle – Тип отрисовки визуальной части

tls_VisualStudio_97	0	Microsoft Visual Studio 97
tls_VisualStudio_NET	1	Microsoft Visual Studio.NET 2003
tls_Office_2003	2	Microsoft Office 2003
tls_VisualStudio2005	3	Microsoft Visual Studio 2005
tls_WindowsXP	4	Microsoft Windows XP native look
tls_Office_2007	5	Microsoft Office 2007
tls_Office_2007_LunaBlue	5	Microsoft Office 2007. Luna Blue
tls_Office_2007_ObsidianBlack	6	Microsoft Office 2007. Obsidian Black
tls_Office_2007_Aqua	7	Microsoft Office 2007. Aqua
tls_Office_2007_Silver	8	Microsoft Office 2007. Silver
tls_VisualStudio2008	9	Microsoft Visual Studio 2008
tls_VisualStudio2010	10	Microsoft Visual Studio 2010
tls_Office_2010_Blue	11	Microsoft Office 2010 Blue
tls_Office_2010_Dark	12	Microsoft Office 2010 Dark
tls_Office_2010_White	13	Microsoft Office 2010 White
tls_Carbon	14	Carbon

ViewMode – Способы отображения моделей

vm_Wireframe	0	каркас
vm_HiddenRemoved	1	без невидимых линий
vm_HiddenThin	2	невидимые линии тонкие
vm_Shaded	3	полутоновой

ksLineBuildingType – Способ построения сегмента ломаной

ksLBTByPoint	0	По точкам
ksLBTXDirection	1	По оси X

ksLBTYDirection	2	По оси Y
ksLBTZDirection	3	По оси Z
ksLBTParallel	4	Параллельно объекту
ksLBTPerpendicular	5	Перпендикулярно объекту
ksLineBuildingType	6	Через построение точки

DirectionTypes – Типы направлений выдавливания

dtNormal	0	прямое направление (для тонкой стенки - наружу)
dtReverse	1	обратное направление (для тонкой стенки - внутрь)
dtBoth	2	в обе стороны
dtMiddlePlane	3	от средней плоскости

Примечание:

1. При типе направления dtMiddlePlane в методах SetSideParam и GetSideParam параметр depth интерпретируется как общая глубина выдавливания и задается следующим образом:

   SetSideParam(TRUE, etBlind, depth,...)

2. В API7 соответствует перечислению ksDirectionTypeEnum.

ksContour3DBuildingTypeTypeEnum – Способ построения Контура 3D

ksCBTUnknown	0	Не определен
ksCBTEdges	1	Ребрами
ksCBTEquidistant	2	Эквидистанта

Obj3dType (ksObj3dTypeEnum) – Типы объектов документа-модели; соответствие интерфейсов API 5 и API 7

В таблице представлены идентификаторы объектов для интерфейсов API 5 и API 7.

.

Идентификатор объекта		Название объекта	Интерфейс параметров API 5	Интерфейс параметров API 7
o3d_unknown	0	неизвестный (включает все объекты)		
o3d_planeXOY	1	плоскость XOY	ksDefaultObject	IPlane3D
o3d_planeXOZ	2	плоскость XOZ	ksDefaultObject	IPlane3D
o3d_planeYOZ	3	плоскость YOZ	ksDefaultObject	IPlane3D
o3d_pointCS	4	точка начала системы координат	ksDefaultObject	IModelObject
Элементы детали				
o3d_sketch	5	эскиз	ksSketchDefinition	ISketch

o3d_face	6	поверхность	ksFaceDefinition	IFace
o3d_edge	7	ребро	ksEdgeDefinition	IEdge
o3d_vertex	8	вершина	ksVertexDefinition	IVertex
Конструктивные элементы				
o3d_axis2Planes	9	ось по двум плоскостям	ksAxis2PlanesDefinition	IAxis3DBy2Planes
o3d_axis2Points	10	ось по двум точкам	ksAxis2PointsDefinition	IAxis3DBy2Points
o3d_axisConeFace	11	ось конической грани	ksAxisConefaceDefinition	IAxis3DByConeface
o3d_axisEdge	12	ось, проходящая через ребро	ksAxisEdgeDefinition	IAxis3DByEdge
o3d_axisOperation	13	ось операции	ksAxisOperationsDefinition	IAxis3DByOperation
o3d_planeOffset	14	смещённая плоскость	ksPlaneOffsetDefinition	IPlane3DByOffset
o3d_planeAngle	15	плоскость под углом	ksPlaneAngleDefinition	IPlane3DByAngle
o3d_plane3Points	16	плоскость по 3-м точкам	ksPlane3PointsDefinition	IPlane3DBy3Points
o3d_planeNormal	17	нормальная плоскость	ksPlaneNormalToSurfaceDefinition	IPlane3DNormalToSurface
o3d_planeTangent	18	касательная плоскость	ksPlaneTangentToSurfaceDefinition	IPlane3DTangentToFace
o3d_planeEdgePoint	19	плоскость через ребро и вершину	ksPlaneEdgePointDefinition	IPlane3DByEdgeAndPoint
o3d_planeParallel	20	плоскость через вершину параллельно другой плоскости	ksPlaneParallelDefinition	IPlane3DParallelByPoint
o3d_planePerpendicular	21	плоскость через вершину перпендикулярно ребру	ksPlanePerpendicularDefinition	IPlane3DPerpendicularByEdge
o3d_planeLineToEdge	22	плоскость через ребро параллельно / перпендикулярно другому ребру	ksPlaneLineToEdgeDefinition	IPlane3DBy2Edge
o3d_planeLineToPlane	23	плоскость через ребро параллельно / перпендикулярно грани	ksPlaneLineToPlaneDefinition	IPlane3DByEdgeAndPlane
o3d_baseExtrusion	24	базовая операция выдавливания	ksBaseExtrusionDefinition	IExtrusion
o3d_bossExtrusion	25	приклеивание выдавливанием	ksBossExtrusionDefinition	IExtrusion
o3d_cutExtrusion	26	вырезать выдавливанием	ksCutExtrusionDefinition	ICutExtrusion
o3d_baseRotated	27	базовая операция вращения	ksBaseRotatedDefinition	IRotated
o3d_bossRotated	28	приклеивание вращением	ksBossRotatedDefinition	IRotated
o3d_cutRotated	29	вырезать вращением	ksCutRotatedDefinition	ICutRotated

o3d_baseLoft	30	базовая операция по сечениям	ksBaseLoftDefinition	ILoft
o3d_bossLoft	31	приклеивание по сечениям	ksBossLoftDefinition	ILoft
o3d_cutLoft	32	вырезать по сечениям	ksCutLoftDefinition	ILoft
o3d_chamfer	33	операция "фаска"	ksChamferDefinition	IChamfer
o3d_fillet	34	операция "скругление"	ksFilletDefinition	IFillet
o3d_meshCopy	35	операция копирования по сетке	ksMeshCopyDefinition	ILinearPattern
o3d_circularCopy	36	операция копирования по концентрической сетке	ksCircularCopyDefinition	ICircularPattern
o3d_curveCopy	37	операция копирования по кривой	ksCurveCopyDefinition	IPathPattern
o3d_circPartArray	38	операция массив по концентрической сетке для сборки	ksCircularPartArrayDefinition	ICircularPattern
o3d_meshPartArray	39	операция массив по сетке для сборки	ksMeshPartArrayDefinition	ILinearPattern
o3d_curvePartArray	40	операция массив по кривой для сборки	ksCurvePartArrayDefinition	IPathPattern
o3d_derivPartArray	41	операция массив по образцу для сборки	ksDerivativePartArrayDefinition	IDerivedPattern
o3d_incline	42	операция "уклон"	ksInclineDefinition	IIncline
o3d_shellOperation	43	операция "оболочка"	ksShellDefinition	IShell
o3d_ribOperation	44	операция "ребро жесткости"	ksRibDefinition	IRib
o3d_baseEvolution	45	кинематическая операция	ksBaseEvolutionDefinition	IEvolution
o3d_bossEvolution	46	приклеить кинематически	ksBossEvolutionDefinition	IEvolution
o3d_cutEvolution	47	вырезать кинематически	ksCutEvolutionDefinition	IEvolution
o3d_mirrorOperation	48	операция "зеркальный массив"	ksMirrorCopyDefinition	IMirrorPattern
o3d_mirrorAllOperation	49	операция "зеркально отразить все"	ksMirrorCopyAllDefinition	IMirrorPattern
o3d_cutByPlane	50	операция "сечение поверхностью"	ksCutByPlaneDefinition	ICut
o3d_cutBySketch	51	операция "сечение эскизом"	ksCutBySketchDefinition	ICut
o3d_holeOperation	52	отверстие		
Кривые				
o3d_polyline	53	ломаная	ksPolyLineDefinition	IPolyLine
o3d_conicSpiral	54	коническая спираль	ksConicSpiralDefinition	IConicSpiral3D
o3d_spline	55	сплайн	ksSplineDefinition	ISpline3D
o3d_cylindricSpiral	56	цилиндрическая спираль	ksCylindricSpiralDefinition	ICylindricSpiral3D
o3d_importedSurface	57	импортированная поверхность	ksImportedSurfaceDefinition	IImportedSurface

o3d_thread	58	условное изображение резьбы	ksThreadDefinition	IThread
o3d_EvolutionSurface	59	кинематическая поверхность	ksEvolutionSurfaceDefinition	IEvolution
o3d_ExtrusionSurface	60	поверхность выдавливания	ksExtrusionSurfaceDefinition	IExtrusionSurface
o3d_RotatedSurface	61	поверхность вращения	ksRotatedSurfaceDefinition	IRotatedSurface
o3d_LoftSurface	62	поверхность по сечениям	ksLoftSurfaceDefinition	ILoft
o3d_MacroObject	63	макроэлемент 3D	ksMacro3DDefinition	IMacroObject3D
o3d_UnionComponents	64	операция объединения компонентов	ksUnionComponentsDefinition	IUnionComponents
o3d_MoldCavity	65	операция вычитания компонентов	ksMoldCavityDefinition	IMoldCavity
o3d_planeMiddle	66	средняя плоскость	ksPlaneMiddleDefinition	IPlane3DMiddle
o3d_controlPoint	67	контрольная точка	ksControlPointDefinition	IControlPoint
o3d_conjunctivePoint	68	присоединительная точка	ksConjunctivePointDefinition	IConjunctivePoint
o3d_aggregate	69	Булева операция	ksAggregateDefinition	IBoolean
o3d_point3D	70	Конструктивная 3D точка		IPoint3D
o3d_axisOX	71	Ось OX	ksDefaultObject	IAxis3D
o3d_axisOY	72	Ось OY	ksDefaultObject	IAxis3D
o3d_axisOZ	73	Ось OZ	ksDefaultObject	IAxis3D
o3d_sheetMetalBody	74	Листовое тело		ISheetMetalBody
o3d_sheetMetalBend	75	Сгиб		ISheetMetalBend
o3d_sheetMetalLineBend	76	Сгиб по линии		ISheetMetalLineBend
o3d_sheetMetalHole	77	Элемент листового тела "отверстие"		ISheetMetalHole
o3d_sheetMetalCut	78	Элемент листового тела "вырез"		ISheetMetalCut
o3d_UnHistoried	79	Операция без истории		
o3d_baselineDimension3D	80	Линейный размер 3D (от отрезка до точки)		IBaseLineDimension3D
o3d_lineDimension3D	81	Линейный размер 3D (на плоскости)		ILineDimension3D
o3d_radialDimension3D	82	Радиальный размер 3D		IRadialDimension3D
o3d_diametralDimension3D	83	Диаметральный размер 3D		IDiametralDimension3D
o3d_angleDimension3D	84	Угловой размер 3D		IAngleDimension3D
o3d_localCoordinateSystem	85	Локальная система координат		ILocalCoordinateSystem
o3d_leader3D	86	Линия-выноска 3D		ILeader
o3d_markLeader3D	87	Знак маркировки 3D		IMarkLeader
o3d_rough3D	88	Обозначение 3D шероховатости		IRough3D
o3d_positionLeader3D	89	Обозначение позиции 3D		IPositionLeader

o3d_brandLeader3D	90	Знак клеймения 3D		IBrandLeader
o3d_base3D	91	Обозначение 3D базы		IBase3D
o3d_tolerance3D	92	Допуск формы 3D		ITolerance3D
o3d_SplitLine	93	Линия разъема		ISplitLine
o3d_SurfacePatch	94	Заплатка		ISurfacePatch
o3d_FaceRemover	95	Операция удаления граней		IFaceRemover
o3d_SurfaceSewer	96	Операция сшивки поверхностей		ISurfaceSewer
o3d_NurbsSurface	97	NURBS-поверхность		INurbsSurface
o3d_SurfacesIntersectionCurve	98	Кривая пересечения поверхностей		ISurfacesIntersectionCurve
o3d_lastEntityElement	99	Всегда последний из Entity!!!		
Элементы, не являющиеся Entity				
o3d_variable	100	параметрическая переменная	ksVariable	IVariable7
o3d_placement	101	местоположение	ksPlacement	IPlacement3D
o3d_entityCollection	102	Массив трехмерных объектов	ksEntityCollection	
o3d_document	103	Документ-модель	ksDocument3D	IKompasDocument3D
o3d_part	104	Деталь	ksPart	IPart7
o3d_entity	105	Объект	ksEntity	IModelObject
o3d_mateConstraint	106	сопряжение	ksMateConstraint	IMateConstraint3D
o3d_mateConstraintCollection	107	Массив сопряжений	ksMateConstraintCollection	IMateConstraints3D
o3d_partCollection	108	Массив элементов сборки	ksPartCollection	IParts7
Объединенные типы объектов для создания EntityCollection				
o3d_constrElement	109	конструктивные элементы-плоскости и оси (конструктивные от o3d_axis2Planes до o3d_plane3Points)		
o3d_operationElement	110	Операции (от o3d_baseExtrusion до o3d_cylindricSpiral)		
o3d_curveElement	111	Кривые (пространственные и ребра)		
o3d_rasterFormat	112	интерфейс параметров для конвертации в растровый формат	ksRasterFormatParam	
o3d_additionFormat	113	интерфейс параметров для конвертации в дополнительные форматы: jgs, sat, xt, x_b, step, stl, VRML	ksAdditionFormatParam	

o3d_bodyCollection	114	интерфейс массива трехмерных тел	ksBodyCollection	
o3d_body	115	интерфейс трехмерного тела	ksBody	IBody7
o3d_faceCollection	116	интерфейс массива граней	ksFaceCollection	
o3d_tessellation	117	интерфейс триангуляции	ksTessellation	ITesselllation7
o3d_facet	118	интерфейс триангуляционной пластины	ksFacet	
o3d_featureCollection	119	интерфейс массива объектов дерева	ksFeatureCollection	
o3d_feature	120	интерфейс объекта дерева	ksFeature	IFeature7
o3d_edgeCollection	121	интерфейс массива ребер	ksEdgeCollection	
o3d_orientedEdge	122	интерфейс ориентированного ребра	ksOrientedEdge	IOrientedEdge7
o3d_orientedEdgeCollection	123	интерфейс массива ориентированных ребер	ksOrientedEdgeCollection	
o3d_loop	124	интерфейс цикла	ksLoop	ILoop7
o3d_loopCollection	125	интерфейс массива циклов	ksLoopCollection	
o3d_curve3D	126	интерфейс математической кривой в трехмерном пространстве	ksCurve3D	IMathCurve3D
o3d_surface	127	интерфейс математической поверхности в трехмерном пространстве	ksSurface	IMathSurface3D
o3d_massInertiaParam	128	Интерфейс параметров для расчета массово-центровочных характеристик	ksMassInertiaParam	IMassInertiaParam7
o3d_line3dParam	129	Интерфейс параметров 3dLine	ksLineSeg3dParam	
o3d_circle3dParam	130	Интерфейс параметров 3dCircle	ksCircle3dParam	
o3d_ellipse3dParam	131	Интерфейс параметров 3dEllipce	ksEllipse3dParam	
o3d_nurbsPoint3dParam	132	Интерфейс параметров точки для трехмерной NURBS	ksNurbsPoint3dParam	

o3d_nurbsPoint3dCollection	133	Интерфейс массива точек для трехмерной NURBS	ksNurbsPoint3dCollection	
o3d_nurbsPoint3dCollCollection	134	Интерфейс массива массивов точек для трехмерной NURBS поверхности	ksNurbsPoint3dCollCollection	
o3d_nurbsKnotCollection	135	Интерфейс массива узлов для трехмерного NURBS	ksNurbsKnotCollection	
o3d_nurbs3dParam	136	Интерфейс параметров трехмерного NURBS	ksNurbs3dParam	
o3d_planeParam	137	Интерфейс параметров плоскости	ksPlaneParam	
o3d_coneParam	138	Интерфейс параметров конической поверхности	ksConeParam	
o3d_cylinderParam	139	Интерфейс параметров цилиндрической поверхности	ksCylinderParam	
o3d_sphereParam	140	Интерфейс параметров сферы	ksSphereParam	
o3d_torusParam	141	Интерфейс параметров тора	ksTorusParam	
o3d_nurbsSurfaceParam	142	Интерфейс параметров NURBS-поверхности	ksNurbsSurfaceParam	
o3d_mateConstraintGroup	143	Объект дерева: группа сопряжений	IModelObject	IFeature7
o3d_measurer	144	Интерфейс для измерений расстояния и угла между двумя примитивами (гранями, ребрами, вершинами)	ksMeasurer	
o3d_selectionMng	145	Интерфейс менеджера выделенных объектов	ksSelectionMng	ISelectionManager
o3d_chooseMng	146	Интерфейс менеджера выбора (подсветки) объектов	ksChooseMng	IChooseManager
o3d_arc3dParam	147	Интерфейс параметров трехмерной дуги	ksArc3dParam	

o3d_deletedCopyCollection	148	Интерфейс массива удаленных индексов для операций копирования и массивов компонент	ksDeletedCopyCollection	
o3d_viewProjection	149	Интерфейс проекции отображения модели в окне	ksViewProjection	
o3d_viewProjectionCollection	150	Интерфейс массива проекций отображения модели в окне	ksViewProjectionCollection	
o3d_attribute	151	Интерфейс атрибута объекта модели	ksAttribute3D	IAttribute
o3d_attributeCollection	152	Интерфейс массива атрибутов объекта модели	ksAttribute3DCollection	
o3d_componentPositioner	153	Интерфейс управления положением компонентов в сборке	ksComponentPositioner	
o3d_modelLibrary	154	Интерфейс библиотеки моделей	ksModelLibrary	IInsertsLibrary
o3d_ObjectsFilter3D	155	Не используется		
o3d_coordinate3dCollection	156	Интерфейс коллекции координат	ksCoordinate3dCollection	
o3d_intersectionResult	157	Интерфейс результатов пересечений двух тел	ksIntersectionResult	
o3d_PolygonalLineVertexParam	158	Параметры вершины полилинии	ksPolyLineVertexParam	ICurveVertexParam
o3d_variableCollection	159	Массив параметрических переменных	ksvariableCollection	
o3d_sTrackingPointsMeasurer	160	Интерфейс для расчета координат точек при S- образном соединении прямолинейных рёбер	IsTrackingPointsMeasurer	
o3d_surfaceElement	161	Поверхности		
o3d_designationElemen	162	Размеры и условные обозначения		
o3d_copyleftObject	163	Объекты доступные для копирования		
o3d_Embodiment	164	Исполнение		IEmbodiment
o3d_userMateConstraint	165	Пользовательское сопряжение		IMateConstraint + IUserObject3D
o3d_firstEntityElement2	500	Первый из Entity2!!!		
o3d_Equidistant3D	501	Эквидистанта 3D		IEquidistant3D

o3d_TrimmedCurve	502	Операция усечения кривой	ITrimmedCurve
o3d_TrimmedCurveObject	503	Усеченная кривая	ITrimmedCurve
o3d_AuxMeshCopy	504	Массив вспомогательной геометрии по сетке	ILinearPattern
o3d_AuxCircularCopy	505	Массив вспомогательной геометрии по концентрической сетке	ICircularPattern
o3d_AuxCurveCopy	506	Массив вспомогательной геометрии вдоль кривой	IPathPattern
o3d_PointDrivenPattern	507	Массив операций по точкам	IPointDrivenPattern
o3d_PartsPointDrivenPattern	508	Массив компонентов по точкам	IPointDrivenPattern
o3d_AuxMirrorOperation	509	Зеркальный массив вспомогательной геометрии	IMirrorPattern
o3d_ConnectCurve	510	Операция соединения кривых	IConnectCurve
o3d_ConnectCurveObject	511	Кривая соединения	IModelObject
o3d_FilletCurve	512	Операция скругления кривых	IFilletCurve
o3d_FilletCurveObject	513	Скругленная кривая	IModelObject
o3d_EquidistantSurface	514	Операция построения эквидистанты поверхности	IEquidistantSurface
o3d_RuledSurface	515	Линейчатая поверхность	IRuledSurface
o3d_TrimmedSurface	516	Операция усечения поверхности	ITrimmedSurface
o3d_ExtensionSurface	517	Операция продления поверхности	IExtensionSurface
o3d_SurfaceThickening	518	Операция придания толщины поверхности	ISurfaceThickening
o3d_Arc3D	519	3D дуга	IArc3D
o3d_AuxPointDrivenPattern	520	Массив вспомогательной геометрии по точкам	IPointDrivenPattern
o3d_BodiesPointDrivenPatter n	521	Массив тел по точкам	IPointDrivenPattern
o3d_TablePattern	522	Массив операций по таблице из файла	ITablePattern
o3d_PartsTablePattern	523	Массив компонентов по таблице из файла	ITablePattern

o3d_AuxTablePattern	524	Массив вспомогательной геометрии по таблице из файла	ITablePattern
o3d_BodiesTablePattern	525	Массив тел по таблице из файла	ITablePattern
o3d_MeshPointsSurface	526	Поверхность по сети точек	IMeshPointsSurface
o3d_CloudPointsSurface	527	Поверхность по пласту (облаку) точек	ICloudPointsSurface
o3d_BodiesMeshCopy	528	Массив тел по сетке	ILinearPattern
o3d_BodiesCircularCopy	529	Массив тел по концентрической сетке	ICircularPattern
o3d_BodiesCurveCopy	530	Массив тел вдоль кривой	IPathPattern
o3d_Scaling3D	531	Масштабирование	IScaling3D
o3d_MirrorPart	532	Зеркальная деталь, с внешней ссылкой на источник (зеркальная вставка детали заготовки)	IBilletObsolete
o3d_sheetMetalUndercut	533	Листовой металл, операция подсечка	ISheetMetalUndercut
o3d_sheetMetalPlate	534	Листовой металл, операция пластина	ISheetMetalPlate
o3d_sheetMetalCombinedBend	535	Листовой металл, операция комбинированный сгиб, - сгиб по эскизу	ISheetMetalSketchBend
o3d_sheetMetalBendStraighten	536	Листовой металл, операция разогнуть	ISheetMetalBendedStraighten
o3d_sheetMetalBendBended	537	Листовой металл, операция согнуть	ISheetMetalBendedStraighten
o3d_sheetMetalBendUnfold	538	Листовой металл, операция развертка	ISheetMetalBendUnfoldParameters
o3d_sheetMetalClosedCorner	539	Листовой металл, операция 'замыкание углов'	ISheetMetalClosedCorner
o3d_sheetMetalBendObject	540	Листовой металл, сгибы листовых операций	IModelObject
o3d_sheetMetalDimpleCutout	541	Листовой металл, закрытая штамповка	ISheetMetalPressForming
o3d_sheetMetalDrawnCutout	542	Листовой металл, открытая штамповка	ISheetMetalPressForming
o3d_sheetMetalBeat	543	Листовой металл, буртик	ISheetMetalShoulder
o3d_sheetMetalLouver	544	Листовой металл, жалюзи	ISheetMetalJalousie

o3d_sheetMetalCowling	545	Обечайка	ISheetMetalRuledShell
o3d_PointsArrOnCurve	546	Группа точек по кривой	IPointsArrOnCurve
o3d_PointsArrFromFile	547	Группа точек из файла	IPointsArrFromFile
o3d_PointsArrOnSurface	548	Группа точек на поверхности	IPointsArrOnSurface
o3d_ArrayExemplar	549	Экземпляр массива	IModelObject
o3d_AuxGeomArrayExemplar	550	Экземпляр массива вспомогательной геометрии	IModelObject
o3d_BodyArrayExemplar	551	Экземпляр массива копирования тел	IModelObject
o3d_NurbsSurfaceByCurvesMesh	552	Сплайновая поверхность по сетке кривых	INurbsSurfacesByCurvesMeshs
o3d_PlaneByPointAndTangentToFace	553	Конструктивная касательная плоскость к грани в точке	IPlane3DTangentToFaceInPoint
o3d_PlaneByPlaneCurve	554	Конструктивная касательная плоскость через плоскую кривую	IPlane3DByPlaneCurve
o3d_JointSurface	555	Поверхность соединения	IJointSurface
o3d_DistanceAngleMeasure	556	Объект 'Измерение расстояния и угла'	IDistanceAngleMeasurement3D
o3d_EdgeLengthMeasure	557	Объект 'Измерение длины ребра'	IEdgeLengthMeasurement3D
o3d_AreaMeasure	558	Объект 'Измерение площади'	IAreaMeasurement3D
o3d_AxisFromPointByDirection	559	Ось через вершину по направлению	IAxis3DByPointAndObject
o3d_Curve3DWithoutHistory	560	Кривая без истории	IUnhistoredCurve3D
o3d_CurveBy2Projections	561	Кривая по двум проекциям	ICurveBy2Projections
o3d_CurveByLaw	562	Кривая по закону	ICurveByLaw
o3d_IsoparametricCurve	563	Изопараметрическая кривая	IIsoparametricCurve
o3d_CurveOutLine	564	Линия очерка	ICurveOutLine
o3d_SplineOnSurface	565	Сплайн на поверхности	ISplineOnSurface
o3d_IsoparametricCurvesSet	566	Группа изопараметрических кривых	IIsoparametricCurvesSet
o3d_ProjectionCurve	567	Проекционная кривая	IProjectionCurve
o3d_Contour3D	568	Контур 3D	IContour3D
o3d_BodyReposition	569	Перепозиционирование тела, поверхности	IBodyReposition
o3d_LineSegment3D	570	Отрезок 3D	ILineSegment3D
o3d_Billet	571	Операция 'деталь заготовка'	IBilletObsolete

o3d_PolyLine3DPoint	572	Точка ломаной и сплайна	IModelObject
o3d_OperationLinearDimension	573	Управляющий линейный размер операции 3D	IBaseLineDimension3D
o3d_OperationAngularDimension	574	Управляющий угловой размер операции 3D	IAngleDimension3D
o3d_OperationRadialDimension	575	Управляющий радиальный размер операции 3D	IRadialDimension3D
o3d_OperationDiametralDimension	576	Управляющий диаметральный размер операции 3D	IDiametralDimension3D
o3d_SketchLinearDimension	577	Управляющий линейный размер эскиза 3D	IBaseLineDimension3D
o3d_SketchAngularDimension	578	Управляющий угловой размер эскиза 3D	IAngleDimension3D
o3d_SketchBreakAngularDimension	579	Управляющий угловой размер эскиза 3D с обрывом	IAngleDimension3D
o3d_SketchRadialDimension	580	Управляющий радиальный размер эскиза 3D	IRadialDimension3D
o3d_SketchBreakRadialDimension	581	Управляющий радиальный размер эскиза 3D с обрывом	IRadialDimension3D
o3d_SketchDiametralDimension	582	Управляющий диаметральный размер эскиза 3D	IDiametralDimension3D
o3d_Hole3D	583	Отверстие 3D	IHole3D
o3d_UserObjectOperation	584	Пользовательская многотельная операция	IUserObject3D
o3d_Zone3D	585	Зона 3D	IZone
o3d_Zone3DDivision	586	Разбиение зон	IZoneDivision
o3d_Zones3D	587	Группа Зоны 3D	IZonesManager
o3d_WireFrame3D	588	Трехмерный каркас	
o3d_CopyGeometry	589	Копия геометрии	ICopyGeometry
o3d_CollectionGeometry	590	Коллекция геометрии	ICollectionGeometry
o3d_MeshPatternAnyCopy	591	Копирование произвольных объектов по сетке	ILinearPattern
o3d_CircularPatternAnyCopy	592	Копирование произвольных объектов по окружности	ICircularPattern
o3d_CurvePatternAnyCopy	593	Копирование произвольных объектов по кривой	IPathPattern

o3d_PointDrivenPatternAnyCopy	595	Копирование произвольных объектов по точкам	IPointDrivenPattern
o3d_TablePatternAnyCopy	596	Копирование произвольных объектов по таблице	ITablePattern
o3d_LinearUnhistoriedDimension	597	Импортированный линейный размер	IBaseLineDimension3D
o3d_AngularUnhistoriedDimension	598	Импортированный угловой размер	IAngleDimension3D
o3d_RadialUnhistoriedDimension	599	Импортированный радиальный размер	IRadialDimension3D
o3d_DiametralUnhistoriedDimension	600	Импортированный диаметральный размер	IDiametralDimension3D
o3d_FaceLift	601	Операция подтягивания граней	IModelObject
o3d_UserWireFrame3D	602	Трехмерный каркас - пользовательский объект	IUserObject3D
o3d_UndefPartObject	603	Модельный объект неопределенного типа	IModelObject
o3d_SpecRough3D	604	Неуказанная шероховатость 3D	ISpecRough3D
o3d_SketchBreakLinearDimension	605	Управляющий линейный размер эскиза 3D с обрывом	IBaseLineDimension3D
o3d_sheetMetalRuledCowling	606	Линейчатая обечайка	ISheetMetalLinearRuledShell, ISheetMetalRuledShell
o3d_UserDesignationObject3D	607	Пользовательский объект обозначение 3D	IUserObject3D
o3d_SplineFormOperation	608	Операция прямого редактирования	
o3d_UnhistoriedBase3D	609	База без истории	IBase3D
o3d_UnhistoriedThread	610	Резьба без истории	IModelObject
o3d_UserDesignationCompObj	611	Составной объект для пользовательских объектов обозначений 3D	IUserObject3D
o3d_UserFolder	612	Пользовательская директория 3D	IUserObject3D
o3d_MeshObject3D	613	Полигональный объект 3D	IMeshObject3D
o3d_sheetMetalRib	614	Ребро усиления	ISheetMetalRib
o3d_axis3D	615	Ось 3D	IAxis3D
o3d_SubFoldLine	616	Подобъект Линия сгиба	IModelObject

o3d_OperationLeaderDimension	617	Управляющий размер операции в виде линии выноски	IBaseLeader3D
o3d_FullFillet	618	Полное скругление	IFullFillet
o3d_DynamicCrossSection	619	Динамическое сечение	IDynamicCrossSection
o3d_RestoredSurface	620	Восстановленная поверхность	IRestoredSurface
o3d_CurvatureGraph	621	График кривизны	
o3d_CollisionObject	622	Информация о коллизии	
o3d_CurvatureCheckObject	623	Проверка кривизны	
o3d_ContinuityCheck	624	Проверка непрерывности	
o3d_SketchFace	625	Контур эскиза	
o3d_lastEntityElement2	1500	Всегда последний из Entity2!!!	

ErrorType3d – Коды ошибок документа модели

Фатальные ошибки - работа прекращается		
et3dNo3dDocument	-7	Документ не активизирован или не является деталью/сборкой
et3dAbort	-1	Аварийное завершение.
Нефатальные ошибки - выполнение продолжается		
et3dError123	123	Объекты пересекаются
et3dError124	124	Неподходящий тип кривой
et3dError125	125	Преобразование текста выполнить невозможно
et3dError126	126	Контур не разбивает ни одну из граней или совпадает с кромкой грани
et3dError127	127	Для данной операции в эскизе должна быть только одна точка
et3dError128	128	Эскиз с одной точкой может использоваться только для крайнего сечения
et3dError129	129	Построение тонкой стенки невозможно, если одно из сечений представляет собой точку
et3dError130	130	Ошибочная топология
et3dError131	131	Нет объекта в прямом направлении
et3dError132	132	Нет объекта в обратном направлении
et3dError133	133	Тела изменены булевой операцией
et3dError134	134	Кривая содержит сегменты нулевой длины
et3dError135	135	Не найдены одна или несколько исходных операций
et3dError136	136	Контур состоит из 2 отрезков, проходящих друг по другу
et3dError137	137	Избыточное количество переменных
et3dError138	138	Область определения не соответствует заданным значениям
et3dError139	139	Неизвестная ошибка при работе транслятора

et3dError140	140	Неизвестная ошибка при работе синтаксического анализатора
et3dError141	141	В строке присутствует неизвестный символ
et3dError142	142	Не хватает закрывающей скобки
et3dError143	143	Не хватает открывающей скобки
et3dError144	144	Невозможная операция
et3dError145	145	Операция потеряла опорные объекты
et3dError146	146	Не все сгибы согнуты/разогнуты
et3dError147	147	Масштабирование с заданным коэффициентом невозможно
et3dError148	148	Масштабирование с заданным соотношением коэффициентов невозможно
et3dError149	149	Массив объектов для копирования пуст
et3dError175	175	Слишком сложный набор элементов для обработки
et3dError176	176	Оси не пересекаются
et3dError177	177	Объекты слишком далеко
et3dError179	179	Контур невозможно использовать для заданного перемещения
et3dError180	180	Один из контуров невозможно использовать для заданного построения
et3dError181	181	Линии разъема созданы не на всех выбранных гранях
et3dError182	182	В некоторых точках образующая параллельна направляющей
et3dError183	183	Недостаточно памяти
et3dError184	184	Направление боковой границы параллельно касательной на конце образующей кривой
et3dError185	185	Объект не найден
et3dError186	186	В сплайновой поверхности часть точек из полюса передвинута, часть осталась совпадающей
et3dError187	187	Аппроксимация не выполнена
et3dError188	188	Не выполнены условия по точности построения
et3dError189	189	Ошибочное значение переменной
et3dError190	190	Невозможно построить фаску на указанных ребрах
et3dError191	191	Остановка скругления невозможна, скругление выполнено без остановки
et3dError192	192	Не согласованы сопряжения на смежных границах
et3dError193	193	Доступ к объекту запрещен
et3dError194	194	Количество сегментов слишком велико
et3dError195	195	Недопустимое положение зазора обечайки
et3dError196	196	Силуэтная грани линия не разрезает грань
et3dError197	197	Вырожденная проекция опорного объекта

```
const long  MAXERROR3d = 192;
```

Intersection_Type – Типы пересечений

itTangentPoint	1	Пересечение точкой
itTangentCurve	2	Пересечение вдоль касательной линии
itTangentSurface	3	Пересечение касательной областью поверхности
itBody	4	Пересечение образует тело

ksMateType – Типы математических объектов, участвующих в сопряжении

ksMateUnknown	0	Неизвестный объект
ksMatePoint	1	Точка
ksMateLine	2	Линия
ksMatePlane	3	Плоскость
ksMateCylinder	4	Цилиндр
ksMateCone	5	Конус
ksMateSphere	6	Сфера
ksMateTorus	7	Тор

Представление математических объектов, участвующих в сопряжении...

ksPatternOrientationTypeEnum – Способ ориентации экземпляров массива

ksOrientationSave	0	Сохранять исходную ориентацию
ksOrientationByNormal	1	Доворачивать до нормали
ksOrientationByObject	2	Ориентировать по объекту

ksProductObjectTypeEnum – Тип объектов дерева СЧИ

ksPOTAllObjects	-1	Не задан. Выдавать все объекты
ksPOTDocumentObject	1	Документ
ksPOTEmbodimentsOb ject	2	Исполнение
ksPOTPartObject	4	Компонент.
ksPOTBodyObject	8	Тело
ksPOTProductObject	16	Изделие
ksPOTInformObject	32	Информационный объект
ksPOTSurfaceObject	64	Поверхность

ksSaveDocumentVersionEnum – Версия сохранения файла

ksSDV_Prev	-1	В предыдущую версию
ksSDV_Current	0	В текущую версию
ksSDV_Kompas_5_11_R03	1	В версию Компас 5.11.R03
ksSDV_Kompas_6_0	2	В версию Компас 6.0.
ksSDV_Kompas_6_Plus	3	В версию Компас 6 Plus
ksSDV_Kompas_7_0	4	В версию Компас 7.0.
ksSDV_Kompas_7_Plus	5	В версию Компас 7 Plus
ksSDV_Kompas_8_0	6	В версию Компас 8.0
ksSDV_Kompas_8_Plus	7	В версию Компас 8 Plus
ksSDV_Kompas_9_0	8	В версию Компас 9.0
ksSDV_Kompas_10_0	9	В версию Компас 10.0
ksSDV_Kompas_11_0	10	В версию Компас 11.0
ksSDV_Kompas_12_0	11	В версию Компас 12.0
ksSDV_Kompas_13_0	12	В версию Компас 13.0
ksSDV_Kompas_14_0	13	В версию Компас 14.0
ksSDV_Kompas_14_Sp1	14	В версию Компас 14 Sp1
ksSDV_Kompas_14_Sp2	15	В версию Компас 14 Sp2
ksSDV_Kompas_15_0	16	В версию Компас 15.0
ksSDV_Kompas_15_Sp1	17	В версию Компас 15 Sp1
ksSDV_Kompas_15_Sp2	18	В версию Компас 15 Sp2
ksSDV_Kompas_16	19	В версию Компас 16
ksSDV_Kompas_16_Sp1	20	В версию Компас 16 Sp1
ksSDV_Kompas_17	21	В версию Компас 17
ksSDV_Kompas_17_Sp1	22	В версию Компас 17_Sp1
ksSDV_Kompas_18	23	В версию Компас 18
ksSDV_Kompas_18_Sp1	24	В версию Компас 18_Sp1
ksSDV_Kompas_19	25	В версию Компас 19

UseColor – Типы используемого цвета

useColorUnknown	-1	тип не определен
useColorOur	0	собственный цвет
useColorOwner	1	цвет хозяина
useColorSource	2	цвет источника
useColorLayer	3	Цвет слоя

ksTreeTypeEnum – Типы Дерева построения 3D документа

ksOperTree	0	Операционное дерево
ksMultiTree	1	Многотельное дерево

ksVariantMarkingTypeEnum – Параметры формирования обозначения

ksVMFullMarking	-1	Полное обозначение
ksVMBaseMarking	0x1	Базовая часть обозначения
ksVMEmbodimentNumber	0x2	Исполнение
ksVMAdditionalNumber	0x4	Дополнительный номер
ksVMCode	0x8	Код документа

ksPrinterTypeEnum – Параметры формирования обозначения

ksPTPrintPreviewPrinter	0	Принтер для печати через предварительный просмотр
ksPTSpecialPrinter	1	Принтер для специальной печати

ProjectionType – Типы проекций

vp_None	-1	Не определена
vp_NormalTo	0	Нормально к текущему планару
vp_Front	1	Спереди - Фронтальная плоскость
vp_Rear	2	Сзади
vp_Up	3	Сверху - Горизонтальная плоскость
vp_Down	4	Снизу
vp_Left	5	Слева - Профильная плоскость
vp_Right	6	Справа
vp_IsoXYZ	7	Изометрия XYZ
vp_IsoYZX	8	Изометрия YZX
vp_IsoZXY	9	Изометрия ZXY
vp_Dio	10	Диметрия

EndType – Типы действий с библиотекой моделей или фрагментов

-1	- закрыть без сохранения
0	- закрыть с сохранением
1	- открыть
2	- редактировать модель или фрагмент
3	- удалить модель или фрагмент из библиотеки
4	- минимизировать окно библиотекаря (в КОМПАС V6 не поддерживается)
5	- создать файл библиотеки фрагментов
6	- создать файл библиотеки моделей
7	- создать фрагмент в библиотеке фрагментов
8	- создать деталь в библиотеке фрагментов
9	- создать сборку в библиотеке фрагментов

2D

LtQualSystem – Система квалитета

lt_qsShaft	1	вала
lt_qsHole	2	отверстия

LtQualDir – Квалитеты

lt_qdPreferable	1	предпочтительные
lt_qdBasic	2	основные
lt_qdAdditional	3	дополнительные

LtRemoteElmSignType – Типы значка объекта "Выносной элемент"

re_Circle	0	окружность
re_Rectangle	1	прямоугольник
re_Ballon	2	скругленный прямоугольник

Типы операций копирования

Тип	Значение	Интерфейс	Операция
o3d_meshCopy	35	ILinearPattern	Операция копирования по сетке
o3d_circularCopy	36	ICircularPattern	Операция копирования по концентрической сетке
o3d_curveCopy	37	IPathPattern	Операция копирования по кривой
o3d_circPartArray	38	ICircularPattern	Операция массив по концентрической сетке для сборки
o3d_meshPartArray	39	ILinearPattern	Операция массив по сетке для сборки
o3d_curvePartArray	40	IPathPattern	Операция массив по кривой для сборки
o3d_derivPartArray	41	IDerivedPattern	Операция массив по образцу для сборки
o3d_mirrorOperation	48	IMirrorPattern	Операция «зеркальный массив»

o3d_mirrorAllOperation	49	IMirrorPattern	Операция «зеркально отразить все» Дополнительно имеет интерфейс выбора тел IChooseBodies7
o3d_AuxMeshCopy	504	ILinearPattern	Массив вспомогательной геометрии по сетке
o3d_AuxCircularCopy	505	ICircularPattern	Массив вспомогательной геометрии по концентрической сетке
o3d_AuxCurveCopy	506	IPathPattern	Массив вспомогательной геометрии вдоль кривой
o3d_PointDrivenPattern	507	IPointDrivenPattern	Массив операций по точкам
o3d_PartsPointDrivenPattern	508	IPointDrivenPattern	Массив компонентов по точкам
o3d_AuxMirrorOperation	509	IMirrorPattern	Зеркальный массив вспомогательной геометрии
o3d_AuxPointDrivenPattern	520	IPointDrivenPattern	Массив вспомогательной геометрии по точкам
o3d_BodiesPointDrivenPatter n	521	IPointDrivenPattern	Массив тел по точкам
o3d_TablePattern	522	ITablePattern	Массив операций по таблице из файла
o3d_PartsTablePattern	523	ITablePattern	Массив компонентов по таблице из файла
o3d_AuxTablePattern	524	ITablePattern	Массив вспомогательной геометрии по таблице из файла
o3d_BodiesTablePattern	525	ITablePattern	Массив тел по таблице из файла
o3d_BodiesMeshCopy	528	ILinearPattern	Массив тел по сетке
o3d_BodiesCircularCopy	529	ICircularPattern	Массив тел по концентрической сетке
o3d_BodiesCurveCopy	530	IPathPattern	Массив тел по кривой

ChangeOrderType – Типы изменения порядка объектов

co_Top	1	Выше всех
co_Bottom	2	Ниже всех
co_BeforeObject	3	Перед объектом
co_AfterObject	4	За объектом
co_UpLevel	5	На уровень вперед
co_DownLevel	6	На уровень назад

DocType – Типы документов системы КОМПАС

lt_DocUnknown	0	- нет активного документа
lt_DocSheetStandart	1	- чертеж стандартного формата
lt_DocSheetUser	2	- чертеж нестандартного формата

lt_DocFragment	3	- фрагмент
lt_DocSpc	4	- спецификация
lt_DocPart3D	5	- деталь
lt_DocAssemble3D	6	- сборка
lt_DocTxtStandart	7	- текстовый документ стандартный
lt_DocTxtUser	8	- текстовый документ нестандартный
lt_DocSpcUser	9	- спецификация - нестандартный формат
lt_DocTechnologyAssemble3D	10	- 3d-документ технологическая сборка

LtNodeType – Типы узла дерева библиотеки документов

tn_root	0	- корень дерева
tn_dir	1	- папка
tn_file	2	- документ (файл)

LtVariantType – Типы данных для LtVariant

ltv_Char	1	символ
ltv_UChar	2	байт
ltv_Int	3	целое
ltv_UInt	4	беззнаковое целое
ltv_Long	5	длинное целое
ltv_Float	6	вещественное
ltv_Double	7	двойное вещественное
ltv_Str	8	строка 255 символов char[255]
ltv_NoUsed	9	пока не используется
ltv_Short	10	короткое целое
ltv_WStr	11	Строка 255 символов whar_t[255]

StructType2DEnum – Типы интерфейсов параметров объектов графического документа, получаемых методом KompasObject::GetParamStruct

ko_Type1	1	ksType1
ko_Type2	2	ksType2
ko_Type3	3	ksType3
ko_Type5	4	ksType5
ko_Type6	5	ksType6
ko_Phantom	6	ksPhantom
ko_PlacementParam	7	ksPlacementParam
ko_ViewParam	8	ksViewParam
ko_LayerParam	9	ksLayerParam
ko_RequestInfo	10	ksRequestInfo
ko_LineSegParam	11	ksLineSegParam
ko_ArcByAngleParam	12	ksArcByAngleParam
ko_ArcByPointParam	13	ksArcByPointParam
ko_MathPointParam	14	ksMathPointParam
ko_RectParam	15	ksRectParam
ko_PointParam	16	ksPointParam
ko_BezierPointParam	17	ksBezierPointParam
ko_NurbsPointParam	18	ksNurbsPointParam

ko_BezierParam	19	ksBezierParam
ko_CircleParam	20	ksCircleParam
ko_LineParam	21	ksLineParam
ko_EllipseParam	22	ksEllipseParam
ko_EllipsArcParam	23	ksEllipseArcParam
ko_EllipsArcParam1	24	ksEllipseArcParam1
ko_EquidParam	25	ksEquidistantParam
ko_HatchParam	26	ksHatchParam
ko_ParagraphParam	27	ksParagraphParam
ko_TextParam	28	ksTextParam
ko_TextLineParam	29	ksTextLineParam
ko_TextItemFont	30	ksTextItemFont
ko_TextItemParam	31	ksTextItemParam
ko_StandartSheet	32	ksStandartSheet
ko_SheetSize	33	ksSheetSize
ko_SheetPar	34	ksSheetPar
ko_DocumentParam	35	ksDocumentParam
ko_ColumnInfoParam	36	ksColumnInfoParam
ko_AttributeType	37	ksAttributeTypeParam
ko_Attribute	38	ksAttributeParam
ko_LibraryAttrTypeParam	39	ksLibraryAttrTypeParam
ko_TAN	40	ksTAN
ko_CON	41	ksCON
ko_DimText	42	ksDimTextParam
ko_LDimSource	43	ksLDimSourceParam
ko_DimDrawing	44	ksDimDrawingParam
ko_LDimParam	45	ksLDimParam
ko_LBreakDimSource	46	ksLBreakDimSource
ko_BreakDimDrawing	47	ksBreakDimDrawing
ko_LBreakDimParam	48	ksLBreakDimParam
ko_ADimSourceParam	49	ksADimSourceParam
ko_ADimParam	50	ksADimParam
ko_ABreakDimParam	51	ksABreakDimParam
ko_RDimSource	52	ksRDimSourceParam
ko_RDimDrawing	53	ksRDimDrawingParam
ko_RDimParam	54	ksRDimParam
ko_RBreakDrawing	55	ksRBreakDrawingParam
ko_RBreakDimParam	56	ksRBreakDimParam
ko_RoughPar	57	ksRoughPar
ko_ShelfPar	58	ksShelfPar
ko_RoughParam	59	ksRoughParam
ko_LeaderParam	60	ksLeaderParam
ko_PosLeaderParam	61	ksPosLeaderParam
ko_BrandLeaderParam	62	ksBrandLeaderParam
ko_MarkerLeaderParam	63	ksMarkerLeaderParam
ko_BaseParam	64	ksBaseParam
ko_CutLineParam	65	ksCutLineParam
ko_ViewPointerParam	66	ksViewPointerParam
ko_ToleranceBranch	67	ksToleranceBranch
ko_ToleranceParam	68	ksToleranceParam
ko_CurvePattern	69	ksCurvePattern
ko_CurvePicture	70	ksCurvePicture
ko_CurvePatternEx	71	ksCurvePatternEx
ko_CurveStyleParam	72	ksCurveStyleParam
ko_DimensionPartsParam	73	ksDimensionPartsParam
ko_TextStyleParam	74	ksTextStyleParam

ko_ConicArcParam	75	ksConicArcParam
ko_PolylineParam	76	ksPolylineParam
ko_LibStyle	77	ksLibStyle
ko_TechnicalDemandParam	78	ksTechnicalDemandParam
ko_SpecRoughParam	79	ksSpecRoughParam
ko_DimensionOptions	80	ksDimensionsOptions
ko_SpcColumnParam	81	ksSpcColumnParam
ko_LibraryStyleParam	82	ksLibraryStyleParam
ko_InertiaParam	83	ksInertiaParam
ko_MassInertiaParam	84	ksMassInertiaParam
ko_VariableParam	85	ksVariable
ko_SnapOptions	86	ksSnapOptions
ko_NurbsParam	87	ksNurbsParam
ko_InsertFragmentParam	88	ksInsertFragmentParam
ko_ConstraintParam	89	ksConstraintParam
ko_CornerParam	90	ksCornerParam
ko_RectangleParam	91	ksRectangleParam
ko_RegularPolygonParam	92	ksRegularPolygonParam
ko_CentreParam	93	ksCentreParam
ko_DocAttachSpcParam	94	ksDocAttachedSpcParam
ko_SpcObjParam	95	ksSpcObjParam
ko_RasterParam	96	ksRasterParam
ko_RecordTypeAttrParam	97	ksRecordTypeAttrParam
ko_NumberTypeAttrParam	98	ksNumberTypeAttrParam
ko_SpcStyleColumnParam	99	ksSpcStyleColumnParam
ko_SpcStyleSectionParam	100	ksSpcStyleSectionParam
ko_SpcSubSectionParam	101	ksSpcSubSectionParam
ko_SpcTuningSectionParam	102	ksSpcTuningSectionParam
ko_SpcTuningStyleParam	103	ksSpcTuningStyleParam
ko_SpcStyleParam	104	ksSpcStyleParam
ko_SpcDescrParam	105	ksSpcDescrParam
ko_QualityItemParam	106	ksQualityItemParam
ko_QualityContensParam	107	ksQualityContensParam
ko_LtVariant	108	ksLtVariant
ko_ContourParam	109	ksContourParam
ko_DoubleValue	110	ksDoubleValue
ko_Char255	111	ksChar255
ko_UserParam	112	ksUserParam
ko_HatchLineParam	113	ksHatchLineParam
ko_HatchStyleParam	114	ksHatchStyleParam
ko_OrdinatedSourceParam	115	ksOrdinatedSourceParam
ko_OrdinatedDrawingParam	116	ksOrdinatedDrawingParam
ko_OrdinatedDimParam	117	ksOrdinatedDimParam
ko_SheetOptions	118	ksSheetOptions
ko_InsertFragmentParamEx	119	ksInsertFragmentParamEx
ko_TreeNodeParam	120	ksTreeNodeParam
ko_AssociationViewParam	121	ksAssociationViewParam
ko_HatchLineParam	122	ksHatchLineParam
ko_AxisLineParam	123	ksAxisLineParam
ko_TextDocumentParam	124	ksTextDocumentParam
ko_CopyObjectParam	126	ksCopyObjectParam
ko_OverlapObjectOptions	127	ksOverlapObjectOptions
ko_ChangeLeaderParam	128	ksChangeLeaderParam
ko_ParametrizationParam	9000	ksParametrizationParam

ErrorType – Ошибки API, кроме 3D

Фатальные ошибки - работа прекращается		
etNoTXTDocument	-8	Документ не активизирован или не является текстовым документом.
etNo3dDocument	-7	Документ не активизирован или не является деталью/сборкой.
etNoAllDocument	-6	Документ не активизирован.
etNoSPCDocument	-5	Документ не активизирован или не является спецификацией.
etLibraryClose	-4	Принудительное завершение выполнения библиотеки.
etNoPreView	-3	В режиме Preview нельзя создавать или открывать видимые документы.
etNoDocument	-2	Документ не активизирован или не является листом/фрагментом.
etAbort	-1	Аварийное завершение.
etSuccess	0	Успешное завершение.
Нефатальные ошибки - выполнение продолжается		
etError1	1	Попытка выполнить EndObj при неоткрытом составном элементе.
etError2	2	Попытка поставить в сплайн недопустимый объект.
etError3	3	Попытка поставить в штриховку недопустимый объект.
etError4	4	Попытка выполнить delete_mtr при невведенной локальной системе координат.
etError5	5	Ошибка при введении локальной системы координат.
etError6	6	Группа должна быть постоянной.
etError7	7	Объект не существует.
etError8	8	В текущем документе объект не найден.
etError9	9	Нет памяти!
etError10	10	Вырожденный объект.
etError11	11	Неверный указатель группы.
etError12	12	Объект не принадлежит группе.
etError13	13	Объект нельзя поставить в группу.
etError14	14	Группа должна быть временной.
etError15	15	Первый объект не существует или не является кривой.
etError16	16	Второй объект не существует или не является кривой.
etError17	17	Кривые расположены в разных видах.
etError18	18	Не совпадают СК определения кривых (геометрическая и аннотационная).
etError19	19	Первый объект не является кривой.
etError20	20	Второй объект не является кривой.
etError21	21	Объект уже в группе

etError22	22	Временный объект не может быть в постоянной группе.
etError23	23	В документе не предусмотрена работа с видами.
etError24	24	Вид с заданным номером уже существует.
etError25	25	Недопустимое значение номера вида.
etError26	26	В текущем документе вид не найден.
etError27	27	Неверный указатель вида.
etError28	28	Вид не редактируется.
etError29	29	Состояние вида задано неверно.
etError30	30	Параметры текущего вида не меняются.
etError31	31	Неверный указатель макроэлемента.
etError32	32	Должен быть режим редактирования макроэлемента.
etError33	33	Неверный тип параметров редактирования макроэлемента.
etError34	34	В виде остались не закрытые составные элементы.
etError35	35	Неверный указатель слоя.
etError36	36	Недопустимое значение номера слоя.
etError37	37	В текущем виде слой не найден.
etError38	38	У объекта параметров нет.
etError39	39	Не соответствует размер структуры параметров.
etError40	40	Состояние слоя задано неверно.
etError41	41	Параметры текущего слоя не меняются.
etError42	42	В указанном виде редактирование запрещено.
etError43	43	В указанном слое редактирование запрещено.
etError44	44	Параметры системного вида не меняются.
etError45	45	Попытка поставить в текст недопустимый объект.
etError46	46	Неверный ввод текста.
etError47	47	Неверный тип массива.
etError48	48	Неверный указатель массива.
etError49	49	Указатель на структуру параметров должен быть не NULL.
etError50	50	Неверный индекс массива.
etError51	51	Неверное редактирование текста.
etError52	52	Bezier-точка используется неверно.
etError53	53	Неправильный индекс.
etError54	54	Режим работы документа задан неверно.
etError55	55	Режим обработки документов задан неверно.
etError56	56	Неверный указатель документа.
etError57	57	Попытка сохранить документ без имени.
etError58	58	Документ закрыт без сохранения.
etError59	59	Имя файла документа задано неверно.
etError60	60	Объект не соответствует типу поиска.
etError61	61	Не могу создать документ. Документ с таким именем уже открыт.
etError62	62	Поиск объектов задан неверно.
etError63	63	В текущем документе итератор не найден.

etError64	64	Документ не найден или неверная структура файла.
etError65	65	Документ открыт в видимом режиме.
etError66	66	Документ открыт в невидимом режиме.
etError67	67	Нельзя менять тип документа.
etError68	68	Стиль спецификации не найден.
etError69	69	У фрагмента нет размеров листа.
etError70	70	Режим работы документа не меняется.
etError71	71	Вид должен быть активным или текущим.
etError72	72	Тип атрибута задан неверно.
etError73	73	Тип атрибута не найден.
etError74	74	Неверный пароль.
etError75	75	Не найдено определение локального фрагмента.
etError76	76	Атрибут в документе не найден.
etError77	77	Атрибут не принадлежит объекту.
etError78	78	Неправильный номер колонки атрибута.
etError79	79	Неправильный номер строки атрибута.
etError80	80	Родительское окно не найдено.
etError81	81	Библиотека атрибутов не найдена или ошибка в библиотеке.
etError82	82	Текст размера задан неверно.
etError83	83	Параметры привязки размера заданы неверно.
etError84	84	Текст шероховатости задан неверно.
etError85	85	Неверный указатель линейного размера.
etError86	86	Текст линии выноски задан неверно.
etError87	87	Параметры линии выноски заданы неверно.
etError88	88	Попытка поставить в контур недопустимый объект.
etError89	89	У звеньев контура не совпадают узлы.
etError90	90	В документе не предусмотрена работа с техническими требованиями.
etError91	91	В документе не предусмотрена работа с неуказанной шероховатостью.
etError92	92	Попытка поставить в таблицу недопустимый объект.
etError93	93	В составном объекте не предусмотрена работа с техническими требованиями.
etError94	94	В документе не предусмотрена работа с основной надписью.
etError95	95	В составном объекте не предусмотрена работа с основной надписью.
etError96	96	Попытка поставить в допуск формы недопустимый объект.
etError97	97	Nurbs-точка используется неверно.
etError98	98	Попытка поставить в полилинию недопустимый объект.
etError99	99	Объект должен быть геометрическим.
etError100	100	Эквидистанту в контур включать нельзя.
etError101	101	Неправильная работа с указателем на определение вставного фрагмента.
etError102	102	Рекурсивная вставка фрагмента.

etError103	103	Ошибка чтения файла фрагмента.
etError104	104	Аналогичное определение вставки фрагмента уже есть. Новый комментарий не принимается.
etError105	105	Неправильная работа с указателем на основную надпись.
etError106	106	Неправильная работа с указателем на неуказанную шероховатость.
etError107	107	Неправильная работа с указателем на технические требования
etError108	108	Документ должен быть активным.
etError109	109	Стиль кривой не найден.
etError110	110	Объект должен быть кривой.
etError111	111	Стиль текста не найден.
etError112	112	Неверно заданы параметры для расчета длины текста.
etError113	113	Новый слой должен существовать и быть доступным для редактирования.
etError114	114	Номер раздела задан неверно.
etError115	115	Стиль спецификации не найден.
etError116	116	В текущем документе объект спецификации не найден.
etError117	117	Попытка подключить к объекту СП недопустимый объект.
etError118	118	Тип объекта задан неверно.
etError119	119	Объект заданного типа не редактируется.
etError120	120	Нужно завершить редактирование составного объекта.
etError121	121	Объект должен быть таблицей.
etError122	122	Объект должен быть допуском формы.
etError123	123	Не найден файл для отрисовки слайда.
etError124	124	Неверная структура файла.
etError125	125	Нужно завершить редактирование объекта спецификации.
etError126	126	Объект должен быть макроэлементом.
etError127	127	Попытка поставить в макро недопустимый объект.
etError128	128	Библиотека фрагментов уже закрыта или не открывалась.
etError129	129	Библиотека фрагментов уже открыта.
etError130	130	Файл библиотеки фрагментов не найден.
etError131	131	Ошибка в структуре файла библиотеки фрагментов.
etError132	132	Ошибка в имени файла библиотеки фрагментов.
etError133	133	Ошибка в имени фрагмента для библиотеки фрагментов.
etError134	134	Доступ к фрагменту в библиотеке фрагментов невозможен.
etError135	135	В документе не предусмотрена работа со спецификацией на листе.
etError136	136	Некорректный вид типа атрибута.
etError137	137	Номер листа спецификации задан неверно.

etError138	138	Документ должен быть открыт в видимом режиме.
etError139	139	Ошибка при обработке картинки для стиля кривой.
etError140	140	Объект должен быть штриховкой.
etError141	141	Объект должен быть текстом.
etError142	142	В документе не предусмотрена работа со спецификацией на листе.
etError143	143	В документе не предусмотрена работа с зонами.
etError144	144	Объект спецификации не редактируется.
etError145	145	Файл с растровым объектом не найден.
etError146	146	Ошибка в определении параметров описания спецификации.
etError147	147	Описание спецификации не найдено.
etError148	148	Имя файла спецификации уже используется в листе.
etError149	149	Описание спецификации данного типа уже есть в листе.
etError150	150	Необходимо завершить текущую операцию.
etError151	151	Редактируемый макроэлемент удалять нельзя.
etError152	152	Служебный файл допусков graphic.tol не найден.
etError153	153	Попытка поставить в макро собственный объект.
etError154	154	Неправильно задано оформление первого листа спецификации: не указана таблица, предназначенная для спецификации.
etError155	155	Не найдено оформление первого листа спецификации.
etError156	156	Неправильно задано оформление первого листа спецификации: не найдены ячейки типа Для спецификации.
etError157	157	Неправильно задано оформление второго и последующих листов спецификации: не указана таблица, предназначенная для спецификации.
etError158	158	Не найдено оформление второго и последующих листов спецификации.
etError159	159	Неправильно задано оформление второго и последующих листов спецификации: не найдены ячейки типа Для спецификации.
etError160	160	Попытка изменить параметры объекта только для чтения.
etError161	161	Попытка изменить параметры объекта только для чтения.
etError162	162	Ошибка создания файла библиотеки.
etError163	163	Текущий документ пустой. Сохранение в выбранном формате производиться не будет.

etError164	164	Не найдена библиотека стилей спецификаций.
etError165	165	Размер растра превышает допустимый максимальный размер (65536 x 65536).
etError166	166	Недостаточно памяти для создания растра указанного размера.
etError167	167	Выбранный диапазон страниц в документе не существует.
etError168	168	Неверный указатель линии-выноски.
etError169	169	Имя документа изменить нельзя. Документ с таким именем уже открыт.
etError170	170	Базовый вид должен быть ассоциативным.
etError171	171	Базовый объект должен быть стрелкой вида.
etError172	172	Базовый объект должен быть выносным элементом.
etError173	173	Неправильно задан цвет.
etError174	174	Неправильно заданы единицы измерения.
etError175	175	Нельзя создать пользовательскую панель свойств.
etError176	176	Метод не используется для данного интерфейса.
etError177	177	Документ, открытый только для чтения, не может быть сохранён
etError178	178	Нельзя отключить работающую библиотеку
etError179	179	Нельзя отключить AddIn-библиотеку
etError184	184	Значение выходит за границы диапазона.
etError185	185	Значение размера выходит за границы диапазона 30'' - 359°59'30''.
etError186	186	Значение размера выходит за границы диапазона 0.5'' - 359°59'59.5''.
etError187	187	Значение размера выходит за границы диапазона 30' - 359°30'.
etError188	188	Ограничение создать нельзя. Данный тип размера не параметризуется.
etError189	189	Ограничение создать нельзя. Нет информации о привязке.
etError190	190	Ограничение создать нельзя.
etError191	191	Ограничение такого типа уже существует.
etError192	192	Ограничение создать нельзя.Недопустимое имя переменной.
etError193	193	Ограничение создать нельзя.Недопустимое значение размера.
etError194	194	Ограничение создать нельзя.Размер может быть только информационным.
etError195	195	Аннотационный объект может быть создан только в составе макро.
etError201	201	Система не имеет решения
etError202	202	Переменная с таким именем уже существует
etError203	203	Недопустимое имя переменной
etError204	204	Невозможно добавить постоянный объект во временный макроэлемент

etError205	205	Недопустимая команда в режиме редактирования макрообъекта
etError205	205	Недопустимая команда в режиме редактирования макрообъекта
etError206	206	Неправильно задан пароль
etError210	210	Некорректное имя
etError211	211	Хранилище с указанным именем уже зарегистрировано
etError212	212	Запрашиваемое хранилище не зарегистрировано
etError213	213	Запрашиваемого файла нет в хранилище
etError214	214	Ошибка при работе с файловой системой
etError215	215	Ошибка при перезаписи файла
etError216	216	Метод не может быть вызван: отсутствует лицензия
etError217	217	Документ создан более поздней версией
etError218	218	Файл защищен с помощью приложения КОМПАС-Защита

MAXERROR 216

TextAlign –Типы привязки текста

txta_Left	0	- точка привязки слева
txta_Center	1	- точка привязки в центре
txta_Right	2	- точка привязки справа

LtViewType – Типы видов чертежа

vtUnknown	-1	Неизвестный тип
vt_System	0	Системный вид с номером 0; создается автоматически, существует всегда
vt_Normal	1	Обычный вид
vt_Arbitrary	2	Произвольный вид
vt_Standart	3	Стандартный вид
vt_Projected	4	Проекционный вид
vt_Arrow	5	Вид по стрелке
vt_Remote	6	Выносной вид
vt_Section	7	Вид разрез\сечение
vt_Remote2D	100	Выносной вид не связанный с 3D моделью

ErrorType – Коды ошибок графического документа

etNo3dDocument	-7	Документ не активизирован или не является трехмерной моделью
etNoAllDocument	-6	Документ не активизирован

etNoSPCDocument	-5	Документ не активизирован или не является спецификацией
etLibraryClose	-4	Принудительное завершение выполнения библиотеки
etNoPreView	-3	В режиме предварительного просмотра нельзя создавать или открывать выводимые документы
etNoDocument	-2	Документ не активизирован или не является листом/фрагментом
etAbort	-1	Аварийное завершение
etSuccess	0	Успешное завершение
etError1	1	Попытка выполнить EndObj при неоткрытом составном элементе
etError2	2	Попытка поставить в сплайн недопустимый объект
etError3	3	Попытка поставить в штриховку недопустимый объект
etError4	4	Попытка выполнить delete_mtr при невведенной локальной системе координат
etError5	5	Ошибка при введении локальной системы координат
etError6	6	Группа должна быть постоянной
etError7	7	Объект не существует
etError8	8	В текущем документе объект не найден
etError9	9	Нет памяти
etError10	10	Вырожденный объект
etError11	11	Неверный указатель группы
etError12	12	Объект не принадлежит группе
etError13	13	Объект нельзя поставить в группу
etError14	14	Группа должна быть временной
etError15	15	Первый объект не существует или не является кривой
etError16	16	Второй объект не существует или не является кривой
etError17	17	Кривые расположены в разных видах
etError18	18	Один (или оба) из указанных объектов - не геометрический
etError19	19	Первый объект не является кривой
etError20	20	Второй объект не является кривой
etError21	21	Объект уже находится в группе
etError22	22	Временный объект не может быть в постоянной группе
etError23	23	В документе не предусмотрена работа с видами
etError24	24	Вид с заданным номером уже существует
etError25	25	Недопустимое значение номера вида
etError26	26	В текущем документе вид не найден
etError27	27	Неверный указатель вида
etError28	28	Вид не редактируется
etError29	29	Состояние вида задано неверно
etError30	30	Параметры текущего вида не меняются
etError31	31	Неверный указатель макроэлемента
etError32	32	Должен быть режим редактирования макроэлемента
etError33	33	Неверный тип параметров редактирования макроэлемента
etError34	34	В виде остались не закрытые составные элементы
etError35	35	Неверный указатель слоя
etError36	36	Недопустимое значение номера слоя
etError37	37	В текущем виде слой не найден
etError38	38	У объекта нет параметров

etError39	39	Размер структуры параметров не соответствует указанному
etError40	40	Состояние слоя задано неверно
etError41	41	Параметры текущего слоя не меняются
etError42	42	В указанном виде редактирование запрещено
etError43	43	В указанном слое редактирование запрещено
etError44	44	Параметры системного вида не меняется
etError45	45	Попытка поставить в текст недопустимый объект
etError46	46	Неверный ввод текста
etError47	47	Неверный тип массива
etError48	48	Неверный указатель массива
etError49	49	Указатель на структуру параметров должен быть не NULL
etError50	50	Неверный индекс массива
etError51	51	Неверное редактирование текста
etError52	52	Точка кривой Безье используется неверно
etError53	53	Неправильный индекс
etError54	54	Режим работы документа задан неверно
etError55	55	Режим обработки документов задан неверно
etError56	56	Неверный указатель на документ
etError57	57	Попытка сохранить документ без имени
etError58	58	Документ закрыт без сохранения
etError59	59	Имя файла документа задано неверно
etError60	60	Объект не соответствует типу поиска
etError61	61	Невозможно создать документ. Документ с таким именем уже открыт
etError62	62	Поиск объектов задан неверно
etError63	63	В текущем документе итератор не найден
etError64	64	Документ не найден или неверная структура файла
etError65	65	Документ открыт в видимом режиме
etError66	66	Документ открыт в невидимом режиме
etError67	67	Нельзя менять тип документа
etError68	68	Стиль спецификации не найден
etError69	69	У фрагмента нет размеров листа
etError70	70	Режим работы документа не меняется
etError71	71	Вид должен быть активным или текущим
etError72	72	Тип атрибута задан неверно
etError73	73	Тип атрибута не найден
etError74	74	Неверный пароль
etError75	75	Не найдено определение локального фрагмента
etError76	76	Атрибут в документе не найден
etError77	77	Атрибут не принадлежит объекту
etError78	78	Неправильный номер колонки атрибута
etError79	79	Неправильный номер строки атрибута
etError80	80	Родительское окно не найдено
etError81	81	Библиотека атрибутов не найдена или ошибка в библиотеке
etError82	82	Текст размера задан неверно
etError83	83	Параметры привязки размера заданы неверно
etError84	84	Текст шероховатости задан неверно
etError85	85	Неверный указатель линейного размера
etError86	86	Текст линии выноски задан неверно
etError87	87	Параметры линии выноски заданы неверно
etError88	88	Попытка поставить в контур недопустимый объект
etError89	89	У звеньев контура не совпадают узлы

etError90	90	В документе не предусмотрена работа с техническими требованиями
etError91	91	В документе не предусмотрена работа с неуказанной шероховатостью
etError92	92	Попытка поставить в таблицу недопустимый объект
etError93	93	В составном объекте не предусмотрена работа с техническими требованиями
etError94	94	В документе не предусмотрена работа с основной надписью
etError95	95	В составном объекте не предусмотрена работа с основной надписью
etError96	96	Попытка поставить в допуск формы недопустимый объект
etError97	97	Точка кривой NURBS используется неверно
etError98	98	Попытка поставить в ломаную линию недопустимый объект
etError99	99	Объект должен быть геометрическим
etError100	100	Эквидистанту в контур включать нельзя
etError101	101	Неправильная работа с указателем на определение вставного фрагмента
etError102	102	Рекурсивная вставка фрагмента
etError103	103	Ошибка чтения файла фрагмента
etError104	104	Аналогичное определение вставки фрагмента уже есть. Новый комментарий не принимается
etError105	105	Неправильная работа с указателем на основную надпись
etError106	106	Неправильная работа с указателем на неуказанную шероховатость
etError107	107	Неправильная работа с указателем на технические требования
etError108	108	Документ должен быть активным
etError109	109	Стиль кривой не найден
etError110	110	Объект должен быть кривой
etError111	111	Стиль текста не найден
etError112	112	Неверно заданы параметры для расчета длины текста
etError113	113	Новый слой должен существовать и быть доступным для редактирования
etError114	114	Номер раздела задан неверно
etError115	115	Стиль спецификации не найден
etError116	116	В текущем документе объект спецификации не найден
etError117	117	Попытка подключить к объекту спецификации недопустимый объект
etError118	118	Тип объекта задан неверно
etError119	119	Объект заданного типа не редактируется
etError120	120	Нужно завершить редактирование составного объекта
etError121	121	Объект должен быть таблицей
etError122	122	Объект должен быть допуском формы
etError123	123	Не найден файл для отрисовки слайда
etError124	124	Неверная структура файла
etError125	125	Нужно завершить редактирование объекта спецификации
etError126	126	Объект должен быть макроэлементом

etError127	127	Попытка поставить в макроэлемент недопустимый объект
etError128	128	Библиотека фрагментов уже закрыта или не открывалась
etError129	129	Библиотека фрагментов уже открыта
etError130	130	Файл библиотеки фрагментов не найден
etError131	131	Ошибка в структуре файла библиотеки фрагментов
etError132	132	Ошибка в имени файла библиотеки фрагментов
etError133	133	Ошибка в имени фрагмента для библиотеки фрагментов
etError134	134	Доступ к фрагменту в библиотеке фрагментов невозможен"
etError135	135	В документе не предусмотрена работа со спецификацией на листе
etError136	136	Некорректный вид типа атрибута
etError137	137	Номер листа спецификации задан неверно
etError138	138	Документ должен быть открыт в видимом режиме
etError139	139	Ошибка при обработке картинки для стиля кривой
etError140	140	Объект должен быть штриховкой
etError141	141	Объект должен быть текстом
etError142	142	В документе не предусмотрена работа со спецификацией на листе
etError143	143	В документе не предусмотрена работа с зонами
etError144	144	Объект спецификации не редактируется
etError145	145	Файл с растровым объектом не найден
etError146	146	Ошибка в определении параметров описания спецификации
etError147	147	Описание спецификации не найдено
etError148	148	Имя файла спецификации уже используется в листе
etError149	149	Описание спецификации данного типа уже есть в листе
etError150	150	Необходимо завершить текущую операцию
etError151	151	Редактируемый макроэлемент удалять нельзя
etError152	152	Служебный файл допусков graphic.tol не найден
etError153	153	Попытка поставить в макроэлемент собственный объект
etError154	154	Неправильно задано оформление первого листа спецификации: не указана таблица, предназначенная для спецификации
etError155	155	Не найдено оформление первого листа спецификации
etError156	156	Неправильно задано оформление первого листа спецификации: не найдены ячейки типа "Для спецификации".
etError157	157	Неправильно задано оформление второго и последующих листов спецификации: не указана таблица, предназначенная для спецификации
etError158	158	Не найдено оформление второго и последующих листов спецификации.
etError159	159	Неправильно задано оформление второго и последующих листов спецификации: не найдены ячейки типа "Для спецификации".

etError160	160	Попытка изменить параметры объекта только для чтения.
etError161	161	Попытка изменить параметры объекта только для чтения.
etError162	162	Ошибка создания файла библиотекаря фрагментов.
etError163	163	Текущий документ пустой. Сохранение в выбранном формате производиться не будет.
etError164	164	Не найдена библиотека стилей спецификаций.
etError165	165	Размер растра превышает допустимый максимальный размер (65536 x 65536).
etError166	166	Не хватает памяти для создания растра указанного размера.
etError167	167	Выбранный диапазон страниц в документе не существует.
etError168	168	Неверный указатель линии выноски.
etError169	169	Имя документа изменить нельзя. Документ с таким именем уже открыт.
etError170	170	Базовый вид должен быть ассоциативным.
etError171	171	Базовый объект должен быть стрелкой вида.
etError172	172	Базовый объект должен быть выносным элементом.
etError173	173	Неправильно задан цвет.
etError174	174	Неправильно заданы единицы измерения.

KGAX - ActiveX компонент

Константы

KGAX - ActiveX компонент предназначен для использования во внешнем приложении и позволяет открывать и работать с документами системы КОМПАС в диалоге пользователя. При работе с документами, открытыми в ActiveX компоненте, доступно использование API интерфейсов системы КОМПАС. Кроме этого, для работы с KGAX разработаны интерфейсы этого элемента управления.

Компонент KGAX.ocx регистрируется при установке системы КОМПАС.

KDocumentType – Предопределенные типы документов

vt_SheetStandart	1	Чертеж. Стандартный формат,
vt_SheetUser	2	Чертеж. Пользовательский формат,
vt_Fragment	3	Фрагмент,
vt_Spc	4	Спецификация,
vt_3DPart	5	Модель,
vt_3DAssembly	6	Сборка,
vt_TextStandart	7	Текстовый документ. Стандартный формат,
vt_TextUser	8	Текстовый документ. Пользовательский формат,
vt_SpcUser	9	Спецификация. Пользовательский формат.

KZoomType – Способы масштабирования окна документа

vt_ZoomWindow	1	Масштабировать окном. (Запускается процесс масштабирования окном).
vt_ZoomIn	2	Увеличить масштаб.
vt_ZoomOut	3	Уменьшить масштаб.
vt_ZoomSelected	4	Показать полностью выделенные объекты.
vt_ZoomEntireDocument	5	Показать весь документ.
vt_Refresh	6	Обновить изображение.

KDocument3DDrawMode – Режимы отображения документа-модели

vt_WireframeMode	0	Каркас.
vt_HiddenRemovedMode	1	Без невидимых линий.
vt_HiddenThinMode	2	Невидимые линии тонкие.
vt_ShadedMode	3	Полутоновое.

LibManagerMode – Режимы установки текущего менеджера библиотек

vt_MainLibManager	-1	Менеджер библиотек главного окна.
vt_Default	0	Отключить использование текущего менеджера библиотек.
vt_CurrentLibManager	1	Менеджер библиотек текущей OLE вставки.

Интерфейсы

Интерфейс _DKGAX

Dispatch интерфейс для элемента KGAX.

Интерфейс _DKGAX – свойства

Caption – Заголовок

Пример...

Интерфейс...

Тип данных:BSTR.

Синтаксис:

BSTR Caption;

Text – Текст

Пример..

Интерфейс...

Тип данных:BSTR.

Синтаксис:

BSTR Text;

DocumentType – Тип документа для отображения

Пример...

Интерфейс...

Тип данных:KDocumentType.

Синтаксис:

KDocumentType DocumentType;

Примечание:

Свойство доступно на этапе разработки диалога. Если оно установлено, то при открытии диалога с KGAX в нем будет создан новый документ данного типа. Список возможных значений определен в KDocumentType.

DocumenFileName – Имя файла документа для отображения

Пример...

Интерфейс...

Тип данных:BSTR.

Синтаксис:

BSTR DocumenFileName;

Примечание:

Свойство доступно на этапе разработки диалога. Если оно установлено, то при открытии диалога с KGAX в нем будет открыт документ с заданным именем.

Document3DDrawMode – Режим отображения документа-модели

Пример...

Интерфейс...

Тип данных:KDocument3DDrawMode.

Синтаксис:

KDocument3DDrawMode Document3DDrawMode;

Примечание:

1. Свойство изменяет тип отображения текущего документа-модели, загруженного в KGAX.
2. Список возможных значений определен в KDocument3DDrawMode.
3. Свойство работает совместно с Document3DWireframeShadedMode.

Document3DWireframeShadedMode – Признак отображения 3D документа (Полутоновое с каркасом)

Пример...

Интерфейс...

Тип данных:VARIANT_BOOL.

Синтаксис:

VARIANT_BOOL Document3DWireframeShadedMode;

Примечание:

1. Свойство изменяет тип отображения текущего документа-модели, загруженного в KGAX.
2. Свойство работает совместно с Document3DDrawMode.
3. Если установлен режим Document3DDrawMode = vt_ShadedMode и Document3DWireframeShadedMode = TRUE, то документ-модель будет отображаться в режиме Полутоновое с каркасом.

Интерфейс _DKGAX – методы

GetKompasObject – Получить интерфейс на API КОМПАС-3D

Интерфейс...

Синтаксис:

KompasObject* GetKompasObject();

Примечание:

Метод позволяет получит доступ к API КОМПАС-3D, что позволяет работать как с документами, загруженными в KGAX, так и с документами, загруженными в КОМПАС.

Смотрите также: PaintObject.

ZoomEntireDocument – Показать весь документ

Интерфейс...

Синтаксис:

void ZoomEntireDocument();

MoveViewDocument – Сдвинуть изображение

Интерфейс...

Синтаксис:

void MoveViewDocument();

Примечание:

Метод запускает процесс сдвига изображения текущего документа, открытого в KGAX.

PanoramaViewDocument – Приблизить/отдалить изображение

Интерфейс...

Синтаксис:

void PanoramaViewDocument();

Примечание:

Метод запускает процесс - Приблизить/отдалить изображение.

RotateViewDocument – Повернуть изображение

Интерфейс...

Синтаксис:

void RotateViewDocument();

Примечание:

Метод запускает процесс - Повернуть изображение.

OrientationDocument – Ориентация изображения

Интерфейс...

Синтаксис:

void OrientationDocument();

Примечание:

Метод вызывает диалог выбора ориентации текущего документа.

StopCurrentProcess – Завершить текущий процесс

Интерфейс...

Синтаксис:

void StopCurrentProcess ([in][defaultvalue(FALSE)] VARIANT_BOOL cancel);

Входные параметры:

cancel	- отмена.

Значения параметра:

FALSE	(по умолчанию) - принять изменения внесенные в процессе,
TRUE	- отменить изменения, внесенные в процессе.

Примечание:

Метод позволяет прервать текущий процесс.

AddNewDocument – Добавить новый документ

Интерфейс...

Синтаксис:

long AddNewDocument ([in]BSTR fileName);

Входные параметры:

fileName	- полное имя файла документа КОМПАС.

Примечание:

Метод позволяет открыть документ с заданным именем в KGAX. Возвращает индекс в массиве документов, открытых в KGAX.

AddNewEmptyDocument – Добавить новый документ

Интерфейс...

Синтаксис:

long AddNewEmptyDocument ([in]KDocumentType docType);

Входные параметры:

docType - тип нового документа из KDocumentType.

Примечание:

Метод создает новый документ заданного типа и открывает его в KGAX.

RemoveDocumentByIndex – Закрыть документ с указанным индексом

Интерфейс...

Синтаксис:

VARIANT_BOOL RemoveDocumentByIndex ([in]long index);

Входные параметры:

index - индекс документа.

Примечание:

1. Перед закрытием документа должны быть освобождены все интерфейсы, захваченные на данном документе.
2. После применения метода индексы документов, открытых после данного документа, уменьшаются на 1.

ActivateDocumentByIndex – Активизировать документ с указанным индексом

Интерфейс...

Синтаксис:

VARIANT_BOOL ActivateDocumentByIndex ([in]long index);

Входные параметры:

index - индекс документа.

GetActiveDocumentIndex – Получить индекс активного документа

Интерфейс...

Синтаксис:

long GetActiveDocumentIndex();

Возвращаемое значение:

- индекс активного документа.

InvalidateActiveDocument – Перерисовать окно активного документа

Интерфейс...

Синтаксис:

VARIANT_BOOL InvalidateActiveDocument ([in]VARIANT_BOOL erase);

Входные параметры:

erase	- Очистка фона

Значения параметра:

TRUE	- очищать фон,
FALSE	- не очищать фон.

Смотрите также:CWnd::Invalidate.

DrawToDC – Отрисовать документ на заданном HDC

Интерфейс...

Синтаксис:

VARIANT_BOOL DrawToDC ([in]OLE_HANDLE dc, [in]long left, [in]long top, [in]long width, [in]long height);

Входные параметры:

dc	- OLE_HANDLE на котором необходимо отрисовать документ, например HDC принтера или окна,
left, top, width, height	- задают квадрат участка документа.

ZoomWindow – Масштабировать окно документа

Интерфейс...

Синтаксис:

void ZoomWindow ([in] [defaultvalue(vt_ZoomWindow)] KZoomType type);

Входные параметры:

type	- тип масштабирования из KZoomType. По умолчанию используется vt_ZoomWindow - Масштабировать окном.

GetDocumentByIndex – Получить указатель на документ по индексу

Интерфейс...

Синтаксис:

LPDISPATCH GetDocumentByIndex([in]long index);

Входные параметры:

index	- индекс документа.

Возвращаемое значение:

index	- Интерфейс активного документа открытого в KGAX: ksDocument2D, ksDocument3D, ksSpcDocument, ksDocumentTxt.

SetCurrentLibManager – Установить текущий менеджер библиотек

Интерфейс...

Синтаксис:

void SetCurrentLibManager ([in]KLibManagerMode mode);

Входные параметры:

mode	- режим установки текущего менеджера библиотек из KLibManagerMode:

Интерфейс _DKGAX – События

OnKgMouseDown – Кнопка мыши нажата

Синтаксис Automation:

void OnKgMouseDown ([in]short nButton,
[in]short nShiftState,
[in]long x,
[in]long y,
[out]VARIANT_BOOL* proceed);

Входные параметры:

nButton	- номер кнопки мыши,
nShiftState	- признак нажатия кнопки Shift,
x, y	- координаты мыши.

Выходные параметры:

proceed - использовать ли стандартную обработку нажатия кнопки мыши.

Примечание:

Приложение может запретить обработку события системой КОМПАС. Для этого переменной proceed нужно присвоить значение FALSE.

OnKgMouseUp – Кнопка мыши отпущена

Синтаксис Automation:

void OnKgMouseUp ([in]short nButton,
[in]short nShiftState,
[in]long x, [in]long y,
[out]VARIANT_BOOL* proceed);

Входные параметры:

nButton - номер кнопки мыши,
nShiftState - признак нажатия кнопки Shift,
x, y - координаты мыши.

Выходные параметры:

proceed - использовать ли стандартную обработку нажатия кнопки мыши.

Примечание:

Приложение может запретить обработку события системой КОМПАС. Для этого переменной proceed нужно присвоить значение FALSE.

OnKgMouseDblClick – Двойной щелчок мышью

Синтаксис Automation:

void OnKgMouseDblClick ([in]short nButton,
[in]short nShiftState,
[in]long x, [in]long y,
[out]VARIANT_BOOL* proceed);

Входные параметры:

nButton - номер кнопки мыши,
nShiftState - признак нажатия кнопки Shift,
x, y - координаты мыши.

Выходные параметры:

proceed - использовать ли стандартную обработку нажатия кнопки мыши.

Примечание:

Приложение может запретить обработку события системой КОМПАС. Для этого переменной proceed нужно присвоить значение FALSE.

OnKgStopCurrentProcess – Завершен процесс панорамирования, поворота, сдвига и т.д.

Синтаксис Automation:

void OnKgStopCurrentProcess();

Примечание:

Данное событие генерируется при завершении запущенного процесса.

OnKgCreate – Окончание создания окна

Синтаксис Automation:

void OnKgCreate ([in]long index);

OnKgPaint – Окончание отрисовки в окне

Синтаксис Automation:

void OnKgPaint ([in]PaintObject* paintObj);

Входные параметры:

paintObj - интерфейс для создания графических объектов.

Примечание:

После завершения отрисовки документа можно отрисовать свои объекты.

Смотрите также:

KompasObject.

OnKgCreateGLList – Окончание создания листа в контексте OpenGL

Синтаксис Automation:

void OnKgCreateGLList ([in]GL

Object* glObj,
[in]KDocument3DDrawMode drawMode);

Входные параметры:

glObj - интерфейс функций OpenGL системы КОМПАС GLObject,
drawMode - тип отображения в документе.

Примечание:

После завершения отрисовки документа можно отрисовать свои объекты.

_____

Смотрите также:

KompasObject.

OnKgAddGabarit – Габариты документа определены

Синтаксис Automation:

void OnKgAddGabarit ([in]GabaritObject* gabObj);

Входные параметры:

glObj - интерфейс функций OpenGL системы КОМПАС GLObject,

Примечание:

После завершения отрисовки документа можно отрисовать свои объекты.

_____

Смотрите также:

KompasObject.

Интерфейс PaintObject

Интерфейс вспомогательных данных для события OnKgPaint.

PaintObject – методы

GetHWND – Получить дескриптор окна

Синтаксис:

OLE_HANDLE GetHWND();

Возвращаемое значение:

- дескриптор окна.

GetDC – Получить дескриптор контекста отображения

Синтаксис:

OLE_HANDLE GetDC();

Возвращаемое значение:

- Дескриптор контекста отображения.

ReleaseDC – Закрыть дескриптор контекста отображения

Синтаксис:

VARIANT_BOOL ReleaseDC ([in]OLE_HANDLE dc);

Входные параметры:

dc DC - дескриптор закрываемого контекста отображения.

Возвращаемое значение:

TRUE - в случае успешного завершения,
FALSE - в случае ошибки.

GetTransformMatrix – Получить коэффициенты для матрицы преобразования координат

Синтаксис:

VARIANT_BOOL GetTransformMatrix ([out]double* a11,
[out]double* a12,
[out]double* a13,
[out]double* a14,
[out]double* a21,
[out]double* a22,
[out]double* a23,
[out]double* a24);

Выходные параметры:

a11, a12, a13, a14 - первая строчка матрицы.

Интерфейс GabaritObject

Интерфейс вспомогательных данных для события OnKgAddGabatit.

Интерфейс GabaritObject – методы

AddGabarit – Добавить дополнительный габарит к габариту документа

Интерфейс...

Синтаксис:

VARIANT_BOOL AddGabarit ([in]double p1X,
[in]double p1Y,
[in]double p1Z,
[in]double p2X,
[in]double p2Y,
[in]double p2Z);

Входные параметры:

p1X, p1Y, p1Z - координаты первой точки габарита,
p2X, p2Y, p2Z - координаты второй точки габарита.

Интерфейс GLObject

Интерфейс вспомогательных данных для события OnKgCreateGLList.

Примечание:

Данный интерфейс нужно использовать при отрисовке с помощью OpenGL. Более подробное описание методов смотри в описании библиотеки OpenGL.

Интерфейс GLObject – методы

VARIANT_BOOL glBegin ([in]long mode);

VARIANT_BOOL glEnd();

VARIANT_BOOL glEnable ([in]long cap);

VARIANT_BOOL glDisable ([in]long cap);

VARIANT_BOOL glColor3d ([in]double r, [in]double g, [in]double b);

VARIANT_BOOL glLineWidth ([in]double w);

VARIANT_BOOL glLineStipple ([in]long factor, [in]short pattern);

VARIANT_BOOL glPointSize ([in]double w);

VARIANT_BOOL glPolygonMode ([in]long face, [in]long mode);

VARIANT_BOOL glVertex2d ([in]double x, [in]double y);

VARIANT_BOOL glVertex2dv ([in]double* pData, [in]long countDouble);

VARIANT_BOOL glVertex3d ([in]double x, [in]double y, [in]double z);

VARIANT_BOOL glVertex3dv ([in]double* pData, [in]long countDouble);

VARIANT_BOOL glVertex4d ([in]double x, [in]double y, [in]double z, [in]double w);

VARIANT_BOOL glVertex4dv ([in]double* pData, [in]long countDouble);

События

Интерфейс событий ActiveX объекта

OnKgAddGabarit

OnKgCreate

OnKgCreateGLList

OnKgMouseDblClick

OnKgMouseDown

OnKgMouseUp

OnKgPaint

OnKgStopCurrentProcess

Event_common

Связь объектов системы с интерфейсами событий

Объекты системы	Интерфейс событий объекта
Объект приложение KompasObject	Интерфейс событий приложенияИнтерфейс событий документаИнтерфейс событий объектов документа-моделиИнтерфейс событий менеджера выделенных объектовИнтерфейс событий основной надписи графического документаИнтерфейс событий объектов графического документаИнтерфейс событий документа-моделиИнтерфейс событий графического документаИнтерфейс событий объекта спецификацииИнтерфейс событий спецификацииИнтерфейс событий документа-спецификации
Графический документ ksDocument2D	Интерфейс событий документаИнтерфейс событий менеджера выделенных объектовИнтерфейс событий основной надписи графического документа Интерфейс событий графического документаИнтерфейс событий объектов графического документаИнтерфейс событий объекта спецификацииИнтерфейс событий спецификации
Спецификация ksSpcDocument	Интерфейс событий документаИнтерфейс событий основной надписи графического документаИнтерфейс событий объекта спецификацииИнтерфейс событий спецификации Интерфейс событий документа-спецификации
Текстовый документ ksDocumentTxt	Интерфейс событий документаИнтерфейс событий основной надписи графического документа
Документ-модель ksDocument3D	Интерфейс событий документаИнтерфейс событий менеджера выделенных объектовИнтерфейс событий объектов документа-моделиИнтерфейс событий объекта спецификацииИнтерфейс событий спецификацииИнтерфейс событий документа-модели

События системы; перечисления

События приложения...

События документа...

События документа-модели...События графического документа...События менеджера выделенных объектов...

События объектов графического документа...

События объектов документа-модели...

События основной надписи графического документа...

События объекта спецификации...События для спецификации...События документа-спецификации...

IsNotIifyProcess - Ограничение обрабатываемых событий

Синтаксис Automation:

Не используется.

Синтаксис COM:

BOOL IsNotifyProcess (int notifyType);

Входной параметр:

notifyType	- номер события в перечислении событий.

Возвращаемое значение:

TRUE	- событие будет обрабатываться,
FALSE	- событие обрабатываться не будет.

Примечание:

В COM перед любым вызовом события система сделает запрос о возможности его обработки с использованием данного метода. Библиотека сама может ограничивать обрабатываемые события.

SpcObjectNotify

Источник событий объекта спецификации.

Указатель на источник событий SpcObjectNotify можно получить можно получить при помощи метода ksSpecification::GetSpcObjectNotify.

В COM источником для подписки является документ-спецификация, графический документ, документ-модель.

Интерфейс событий объекта спецификации...

SpcDocumentNotify

Источник событий для документа-спецификации.

Указатель на источник событий SpcDocumentNotify можно получить при помощи метода ksSpcDocument::GetSpcDocumentNotify.

В COM источником для подписки является документ-спецификация.

Интерфейс событий для документа-спецификации...

Document3DNotify

Интерфейс событий документа-модели...

Источник событий документа-модели.

Интерфейс позволяет контролировать редактирование документа-модели.

Указатель на источник событий Document3DNotify можно получить при помощи метода ksDocument3D::GetDocument3DNotify.

В COM источником для подписки является документ-модель.

По координатам

Требуется задать следующие параметры:

координаты центра (XC; YC),

координаты начальной и конечной точек (X1; Y1), (X2; Y2),

радиус Radius,

направление Direction.

По трем точкам

Требуется задать следующие параметры:

координаты начальной (X1; Y1) и конечной (X2; Y2) точек,

координаты точки, лежащей на дуге (X3; Y3),

По углам

Требуется задать следующие параметры:

координаты центра (XC; YC),

угол между осью X и направлением на первую точку Angle1,

угол между осью X и направлением на вторую точку Angle2,

радиус Radius,

направление Direction.

По точке и углу

Требуется задать следующие параметры:

координаты центра (XC; YC),

координаты начальной (X1; Y1) или конечной (X2; Y2) точек,

угол между осью X и направлением на первую точку Angle1, если заданы координаты точки 2

угол между осью X и направлением на вторую точку Angle2, если заданы координаты точки 1

направление Direction.

По углу, точке и радиусу

Требуется задать следующие параметры:

координаты центра (XC; YC),

угол между осью X и направлением на первую Angle1, или вторую Angle2 точку,

координаты начальной (X1; Y1), если задан угол на точку 2 или конечной (X2; Y2), если задан угол на точку 1 точек,

радиус Radius,

направление Direction.

По конечным точкам и углу

Требуется задать следующие параметры:

координаты начальной (X1; Y1) и конечной (X2; Y2) точек,

угол между осью X и направлением на первую Angle1 или вторую точку Angle2,

направление Direction.

Плавающий центр, вариант 1

Требуется задать следующие параметры:

угол между осью X и направлением на первую Angle1 и вторую точку Angle2,

координаты начальной (X1; Y1) или конечной (X2; Y2)) точек,

направление Direction.

Плавающий центр, вариант 2

Требуется задать следующие параметры:

координаты начальной (X1; Y1) или конечной (X2; Y2) точек,

угол между осью X и направлением на первую точку Angle1, если заданы координаты точки 2

угол между осью X и направлением на вторую точку Angle2,, если заданы координаты точки 1

радиус Radius,

направление Direction.

NewConstraint - пример использования при создании ассоциативного размера

…

IParametriticConstraintPtr association = dimObj1->NewConstraint();

```
if ( association )
{
 _variant_t partner = CreatePartner( obj1, obj2 ); // obj1, obj2 - объекты с которыми ac-
социируем размер
// Создаем ограничение Ассоциативность
      association->ConstraintType = ksCAssociation;
      association->Partner = partner;
      association->Create();
}
...
//---------------------------------------------------------------------------
// Создать массив SafeArray типа LPDISPATCH
// ---
_variant_t CreatePartner( LPDISPATCH obj1, LPDISPATCH obj2 )
{
  _variant_t res;
  if ( obj2 )
  {
    SAFEARRAY * pSANew = NULL;
    SAFEARRAYBOUND rgsabound;
     rgsabound.lLbound = 0;
     rgsabound.cElements = 2;  // количество элементов в массиве
     pSANew = ::SafeArrayCreate(VT_DISPATCH, 1, &rgsabound);
     if ( pSANew )
     {
     long i1 = 0;
     long i2 = 1;
     ::SafeArrayPutElement( pSANew, &i1, obj1 );  // положить первый объект
     ::SafeArrayPutElement( pSANew, &i2, obj2 );  // положить второй объект

      res.vt = VT_ARRAY | VT_DISPATCH;
      V_ARRAY(&res) = pSANew;
     }
 }
 else
 {
    if ( obj1 )
      res = obj1;
```

```
 }
 return res;
}
```

Главная центральная система координат

Ось OX называется главной осью инерции тела, если центробежные моменты инерции Jxy и Jxz одновременно равны нулю. Через каждую точку тела можно провести три главные оси инерции. Эти оси взаимно перпендикулярны друг другу. Моменты инерции тела относительно трёх главных осей инерции, проведённых в произвольной точке O тела, называются главными моментами инерции тела.

Главные оси инерции, проходящие через центр масс тела, называются главными центральными осями инерции тела, а моменты инерции относительно этих осей — его главными центральными моментами инерции.

Центральная система координат

Ортогональная система координат с началом в центре масс тела и осями, параллельными осям глобальной (исходной) системы координат.

Пример

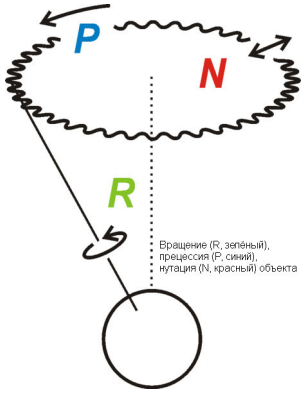
Для подсчета количества компонентов определенного типа на заданном уровне сборки нужно у интерфейса компонентов IPart7 получить список уникальных компонентов с помощью свойства PartsEx[(long)ksUniqueParts]. В списке найденных следует найти нужный компонент и у IPart7 получить количество вставок InstanceCount найденного компонента.

Пример:

```
if (  newKompasAPI ) // newKompasAPI - интерфейс IApplicationPtr
{
IKompasDocument3DPtr doc3D(  newKompasAPI->ActiveDocument );
if (  doc3D )
{
IPart7Ptr topPart (  doc3D->TopPart );
if (  topPart )
{
        long count = topPart->GetInstanceCount(  NULL ); // Общее количество компонентов
Message(  _bstr_t( count) );
_variant_t uniqueParts(  topPart->PartsEx[( long)ksUniqueParts] );
         if (  uniqueParts.vt == (  VT_ARRAY | VT_DISPATCH ) )
{
```

```
                int count = uniqueParts.parray->rgsabound[0].cElements - uniqueParts.parray-
>rgsabound[0].lLbound;
          if (  uniqueParts.parray->cDims == 1 )
{
            HRESULT hr;
            LPDISPATCH HUGEP *pvar;
            hr = ::SafeArrayAccessData( uniqueParts.parray, ( void HUGEP* FAR*)&pvar);
            if (  !FAILED( hr) && pvar )
{
              for (  int i = 0; i < count; i++ )
{
                IPart7Ptr obj(  pvar[i] );
                if (  obj )
{
             long count = topPart->GetInstanceCount(  obj ); // Количество вставок компонента
Obj
       Message(  _bstr_t( count) );
}
}
hr = ::SafeArrayUnaccessData( uniqueParts.parray);
}
}
}
}
}
}
```

Вращение, прецессия и нутация



Прецессия

Прецессия — явление, при котором ось вращающегося объекта поворачивается, например, под действием внешних моментов.

См. Рисунок...

Нутация

Нутация (от лат. nutatio — колебание), происходящее одновременно с прецессией движение твёрдого тела, при котором изменяется угол между осью собственного вращения тела и осью, вокруг которой происходит прецессия; этот угол называется углом нутации.

См.Рисунок...

ksZoom - пример использования

```
//открыть документ "1.cdw"
reference d = OpenDocument ("1.cdw", 0);
//в текущем виде нарисовать отрезок
LineSeg(20, 20, 40, 40, 1);
//увеличить изображение окна, чтобы отрезок оказался в центре и в окне
if(!ksZoom (15, 15, 45, 45))
 Error ("Ошибка при задании окна ");
```

ksZoomScale - пример использования

```
//открыть документ "1.cdw"
reference d = OpenDocument ("1.cdw", 0);
//в текущем виде нарисовать отрезок
```

```
LineSeg (20, 20, 40, 40, 1);
//увеличить изображение окна, чтобы отрезок оказался в центре и в окне
if(!ksZoomScale (30, 30, 2))
 Error("Ошибка при задании окна ");
```

ksZoomPrevNextOrAll - пример использования

```
//открыть документ "1.cdw"
reference d =OpenDocument ("1.cdw", 0);
//показать в окне весь документ
ksZoomPrevNextOrAll (2);
```

ksGetZoomScale - пример использования

```
double x, y, scale;
 if (ksGetZoomScale (&x, &y, &scale)) {
  ksZoomScale (100, 100, 2);
  Message("Изменен масштаб в окне");
  ksZoomScale (x, y, scale);
  Message("Предыдущий масштаб");
 }
 else
  MessageBoxResult();
```

ksRefreshActiveWindow - пример использования

```
{
ksRefreshActiveWindow ();
}
```

Пример использования тригонометрических функций

```
void Trig_Example (void) {

double h, b, Ang =60;

   /* Построение прямоугольного треугольника с гипотенузой L и углом Ang */

h = L * SinD (Ang); /* высота треугольника */
b = L * CosD (Ang); /* основание треугольника */
```

```
LineSeg ( 0, 0, 0, h, 1);
LineSeg ( 0, h, b, 0, 1);
LineSeg ( b,  0, 0, 0, 1);

   /* Связь катетов */
h = b * TanD (Ang);

   /* Обратное вычисление угла */
Ang = AtanD (h / b);

}; /* Trig_Example */
```

Symmetry - пример использования

```
//-------------------------------------------------------------------------------
// Симметрия точки
// ---
void SymmetryPoint()
{
  double px=10, py=10;
  double
    os_x1 = 50, os_y1 = 50,
    os_x2 = 60, os_y2 = 50;

  // Отрисовка точки
  Point( px, py, 0 );

  // Отрисовка отрезка
  LineSeg( os_x1, os_y1, os_x2, os_y2, 1 );

  double s_px, s_py; // Результат симметрии точки

  // Получить координаты точки, симметричной относительно заданной оси
  Symmetry( px, py,   // Координаты точки
          os_x1, os_y1,   // Начальная точка оси
          os_x2, os_y2,   // Конечная точка оси
          &s_px, &s_py ); // Результат симметрии
```

```
  // Отрисовка результирующей точки
  Point( s_px, s_py, 1 );

  // Результат симметрии
  TCHAR buf[255];
  _stprintf_s( buf, _T("x = %4.2f, y = %4.2f"), s_px, s_py );
  MessageT( buf );

  //---- работа с в режиме преобразования координат

  ksMtr(100,100,0,1,1);

  // Отрисовка точки
  Point( px, py, 0 );

  // Отрисовка отрезка
  LineSeg( os_x1, os_y1, os_x2, os_y2, 1 );

  double lcs_px, lcs_py;

  ksPointFromMtr( px, py, &lcs_px, &lcs_py );

  Symmetry( lcs_px, lcs_py,   // Координаты точки
    os_x1, os_y1,   // Начальная точка оси
    os_x2, os_y2,   // Конечная точка оси
    &s_px, &s_py ); // Результат симметрии

  // Отрисовка результирующей точки
  ksPointIntoMtr(s_px, s_py, &lcs_px, &lcs_py );
  Point( lcs_px, lcs_py, 1 );

  DeleteMtr();

  _stprintf_s( buf, _T("x = %4.2f, y = %4.2f"), lcs_px, lcs_py );
  MessageT( buf );
}
```

Rotate - пример использования

```
void Rotate_Example (void) {

double x[2], y[2];


LineSeg ( 10, 60, 60, 90, 1);

Rotate (10, 60, 0, 0, 90, x[0], y[0]);
Rotate (60, 90, 0, 0, 90, x[1], y[1]);

LineSeg (x[0], y[0], x[1], y[1], 1);

}; /* Rotate_Example */
```

MovePoint - пример использования

```
void MovePointExample (void) {

double x1, y1, x2, y2 ;

x1 = 10;
y1 = 60;
x2 = 60;
y2 = 90;

LineSeg ( x1, y1, x2, y2, 1);

MovePoint (&x1, &y1, 90, 100);
MovePoint (&x2, &y2, 90, 100);

LineSeg ( x1, y1, x2, y2, 1);

}; /* MovePointExample */
```

ksPointIntoMtr - пример использования

```
{
double x, y, xn, yn;

x = 10;
y = 20;
Point (x, y, 1);
Mtr(100, 100, 45, 2);

ksPointIntoMtr (x, y, &xn, &yn);
}
```

ksPointFromMtr - пример использования

```
{
double x, y, xn, yn;

Mtr(100, 100, 45, 2);
x = 10;
y = 20;
Point (x, y, 1);
ksPointFromMtr (x, y, &xn, &yn);
}
```

ksLengthIntoMtr - пример использования

```
{
double Len;

Len = 100;
Mtr(100, 100, 45, 2);
ksLenghtIntoMtr (&Len);
}
```

ksLengthFromMtr - пример использования

```
{
```

```
double Len;

Mtr(100, 100, 45, 2);
Len = 100;
ksLenghtFromMtr (&Len);
}
```

ksEqualPoints - пример использования

```
double x1 = 10, y1= 10;
 double x2 = 20, y2= 20;

 if(ksEqualPoints(x1, y1, x2, y2))
  Message("Точки эквивалентны");
 else
  Message("Точки не эквивалентны");
```

ksGetCurvePerpendicular, ksPointsOnCurve - пример использования

```
//равномерное размещение точек и перпендикуляров к ним на кривой
reference pObj;
int count = 10;
RequestInfo info;
double x, y;
memset (&info, 0, sizeof (info));
info.prompt = "Укажите кривую";
int j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj(x, y, 1e6)))
{
//узнаем тип объекта
int type = GetObjParam (pObj, 0, 0, 0);//указатель на графический объект
if (type == CIRCLE_OBJ||//окружность
type == ARC_OBJ||//дуга
type == NURBS_OBJ||//nurbs
type == LINESEG_OBJ||//отрезок
```

```
type == BEZIER_OBJ|| //bezier
type == CONTOUR_OBJ|| //контур
type == POLYLINE_OBJ|| //полилиния
type == ELLIPSE_OBJ|| //эллипс
type == ELLIPSE_ARC_OBJ|| //дуга эллипса
type == RECTANGLE_OBJ|| //прямоугольник
type == REGULARPOLYGON_OBJ)//многоугольник
{
//получим массив точек
reference pointArr = ksPointsOnCurve (pObj,//указатель на кривую
count);//количество точек
if (pointArr)
{
for (uint i = 0; i < count; i)
{
//в цикле получим все точки на кривой
MathPointParam par;
if (GetArrayItem (pointArr,// указатель на массив
i,// индекс в массиве
// (нумерация начинается с 0)
&par,// указатель на структуру элемента
sizeof (MathPointParam)))
// размер структуры элемента
{
//построим точку
Point (par.x, par.y, 0);
//найдем перпендикуляр к кривой в точке
double angl = ksGetCurvePerpendicular(pObj, par.x, par.y);
//построим перпендикуляр (отрезок длиной 10 мм)
double x1 = par.x, y1 = par.y, x2 = par.x, y2 = par.y;
MovePoint ( &x1, &y1, angl, 5);
MovePoint ( &x2, &y2, angl180, 5);
LineSeg (x1, y1, x2, y2, 2);
}
else
MessageBoxResult ();//ошибка доступа к объекту массива
}
}
```

```
else
MessageBoxResult ();//ошибка
}
else
Error ("Выбранный объект - не кривая");
}
}
```

ksMakeEncloseContours, ksHatch - пример использования

```
RequestInfo info;
 //обнулить структуру info;
 memset(&info, 0, sizeof(info));
 double x, y;
 info.prompt = "Укажите точку внутри области";
 int j = Cursor(&info, &x ,&y, 0);
 if (j) {
  HatchParam par;
  memset(&par, 0, sizeof(par));
  par.step = 2;
  par.ang = 45;
  //границы штриховки вычисляем
  par.pBoundaries = ksMakeEncloseContours(0, x, y);
  //штриховка
  reference h = ksHatch(&par);
  if (h) {
   LightObj(h, 1);
   Message("Создали штриховку в виде по точке внутри области");
   LightObj(h, 0);
  }
  else
   Error("Ошибка в задании штриховки");
 }
```

ksIsPointInsideContour, ksIsCurveClosed - пример использования

```
reference pObj;
```

```
 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите контур";
 int j = Cursor(&info, &x ,&y, 0);
 if (j) {
  //выбрать контур
  if(ExistObj(pObj = FindObj(x, y, 1e6)) && GetObjParam(pObj, 0,0,0) == CONTOUR_OBJ ){
   //проверим контур на замкнутость
   int close = ksIsCurveClosed(pObj);
   char buf[250];
   sprintf(buf, " контур %s", close ? "замкнут" : "разомкнут");
   Message(buf);
   info.prompt = "Укажите точку";
   do {
    //задать точку
    j = Cursor(&info, &x ,&y, 0);
    if (j) {
     //проверяем расположение точки
     int side = ksIsPointInsideContour(pObj, x, y, 0.001);
     sprintf(buf, " точка %s", side==1 ? "вне контура" : side==2 ? "на контуре" : "внутри конту-
ра");
      Message(buf);
    }
   } while(j);
  }
  else
   Error("Это не контур");
 }
```

ksTanCurvCurv - пример использования

```
double x = 0, y = 0;
  if ( Cursor( NULL, &x, &y, NULL) )
  {
   reference obj1 = FindObj( x, y, ::ksGetCursorLimit() );
   if ( obj1 )
   {
```

```
 if ( Cursor( NULL, &x, &y, NULL) )
 {
  reference obj2 = FindObj( x, y, ::ksGetCursorLimit() );
  if ( obj2 && obj2 != obj1 )
  {
   reference pointArr1 = CreateArray( POINT_ARR, NULL );
   reference pointArr2 = CreateArray( POINT_ARR, NULL );
   if ( pointArr1 && pointArr2 )
   {
    int res = ksTanCurvCurv( obj1, obj2, pointArr1, pointArr2 );
    if ( res == 1 )
    {
     int count = GetArrayCount( pointArr1 );
     for ( int i = 0; i < count; i )
     {
      MathPointParam point1;
      MathPointParam point2;
      GetArrayItem( pointArr1, i, &point1, sizeof(MathPointParam) );
      GetArrayItem( pointArr2, i, &point2, sizeof(MathPointParam) );
      LineSeg( point1.x, point1.y, point2.x, point2.y, 1 );
     }
    }
   }
   DeleteArray( pointArr1 );
   DeleteArray( pointArr2 );
  }
 }
}
}
```

GetViewNumber, SheetToView, ViewToSheet пример использования

```
//перенесем отрезки из текущего вида в вид номер 2
//получим номер текущего вида
int currentViewNumb = GetViewNumber(0);
reference obj;

//в текущем документе и виде создадим итератор для хождения по отрезкам
```

```
reference itObj = CreateIterator (LINESEG_OBJ,//тип поиска объекта
0);//указатель на объект
//(для движения по группе и внутри макро)
if (itObj)
{
if (ExistObj (obj = MoveIterator (itObj, 'F')))
{
do
{
LineSegParam par;
//берем параметры отрезка в системе координат текущего вида
GetObjParam (obj, &par, sizeof (LineSegParam), ALLPARAM);
//переводим координаты отрезка из вида в лист
ViewToSheet (par.x1, par.y1, &par.x1, &par.y1);
ViewToSheet(par.x2, par.y2, &par.x2, &par.y2);

//открываем  вид 2
OpenView (2);

//переводим координаты отрезка из  листа в вид 2
SheetToView (par.x1, par.y1, &par.x1, &par.y1);
SheetToView (par.x2, par.y2, &par.x2, &par.y2);

//создаем отрезок в виде 2
LineSeg (par.x1, par.y1, par.x2, par.y2, par.style);

//возвращаемся назад в вид currentViewNumb
OpenView (currentViewNumb);

//удаляем отрезок в текущем виде
DeleteObj (obj);
}
while (ExistObj (obj = MoveIterator (itObj, 'N')));
}
}
```

IntersectLinLin - пример использования

```
void IntersectLinLin_Example (void) {

double x, y;
int k;

Line ( 20, 20,  45);
Line ( 80, 20, 120);

IntersectLinLine (20, 20, 45, 80, 20, 120, &k, &x, &y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */

}; /* IntersectLinLin_Example */
```

IntersectLinSLine - пример использования

```
void IntersectLinSLine_Example (void) {

double x, y;
int k;

LineSeg ( 20, 20,  100, 100, 1);
Line ( 80, 20, 120);

IntersectLinSLine (20, 20, 100, 100, 80, 20, 120, &k, &x, &y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */
```

```
};
```

IntersectCirLin - пример использования

```
void IntersectCirLin_Example (void) {

double x, y;
int k;

Circle ( 60, 60, 30, 1);
Line ( 20, 20, 45);

IntersectCirLine (60, 60, 30, 20, 20, 45, &k, &x, &y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */

};
```

IntersectArcLine - пример использования

```
void IntersectArcLin_Example (void) {

double x[2], y[2];
int k, i;

Arc ( 60, 60,  30, 0, 90, 1, 1);
Line ( 20, 20, 45);

IntersectArcLin (60, 60, 30, 0, 90, 1, 20, 20, 45, &k, x, y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k)
for (I=0; i<k; I) Point (x[i], y[i], 1); /* точки пересечения */
};
```

IntersectLinSLinS - пример использования

```
void IntersectLinSLinS_Example (void) {

double x, y;
int k;

LineSeg ( 20, 20,  100, 100, 1);
LineSeg ( 80, 20,  0, 50, 1);

IntersectLinSLinS (20, 20, 100, 100, 80, 20, 0, 50, &k, &x, &y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */

};
```

IntersectLinSCir - пример использования

```
void IntersectLinSCir_Example (void) {

double x, y;
int k;

Circle ( 60, 60,  30, 1);
LineSeg ( 20, 20,  100, 100, 1);

IntersectLinSCir (20, 20, 100, 100, 60, 60, 30, &k, &x, &y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */
```

```
};
```

IntersectLinSArc - пример использования

```
void IntersectLinSArc_Example (void) {

double x, y;
int k;

Arc ( 60, 60, 30, 0, 90, 1, 1);
LineSeg ( 20, 20, 100, 100, 1);

IntersectArcLin (20, 20, 100, 100, 60, 60,  30, 0, 90, 1, &k, x, y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k) Point (x, y, 1); /* точка пересечения */

};
```

IntersectCirCir - пример использования

```
void IntersectCirCir_Example (void) {

double x[2], y[2];
int k, i;

Circle  ( 60, 60,   30,  1);
Circle  ( 75, 60,   30,  1);

IntersectCirCir (60, 60, 30, 75, 60, 30, &k, x, y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k)
```

```
for (l=0; i<k; l) Point (x[i], y[i], 1); /* точки пересечения */

};
```

IntersectArcArc - пример использования

```
void IntersectArcArc_Example (void) {

double x[2], y[2];
int k, i;

Arc  ( 60, 60,   30,  -90, 90,  1, 1);
Arc  ( 75, 60,   30,  -90, 90, -1, 1);

IntersectArcArc (60, 60, 30, -90, 90, 1, 75, 60, 30, -90, 90, -1, &k, x, y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k)
for (l=0; i<k; l) Point (x[i], y[i], 1); /* точки пересечения */

};
```

IntersectCirArc - пример использования

```
void IntersectCirArc_Example (void) {

double x[2], y[2];
int k, i;

Circle ( 60, 60,   30,  1);
Arc  ( 75, 60,   30,  -90, 90, -1, 1);

IntersectCirArc (60, 60, 30, 75, 60, 30, -90, 90, -1, &k, x, y);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
```

```
  ::Message(buf);
if (k)
for (l=0; i<k; l) Point (x[i], y[i], 1); /* точки пересечения */

};
```

IntersectCurvCurv - пример использования

```
void IntersectCurvCurv_Example (void) {

double x[8], y[8];
int k, l;
reference p1, p2;

p1 = Circle ( 60, 60,   30,  1);

p2 = Bezier ( 0, 2);
Point ( 20, 20, 0);
Point ( 40, 50, 0);
Point ( 50, 30, 0);
Point ( 70, 90, 0);
Point (100, 20, 0);
EndObj();

IntersectCurvCurv (p1, p2, &k, x, y, 99);

  char buf[128];
  sprintf(buf, "Количество пересечений = %i", k);
  ::Message(buf);
if (k)
for (l=0; i<k; l) Point (x[i], y[i], 1); /* точки пересечения */

};
```

ksIntersectCurvCurv - пример использования

```
reference line = ::LineSeg(-100, -100, 100, 100, 1); // отрезок
reference cir = ::Circle (20,  20,  35,    1); // окружность
```

```
reference array = ::CreateArray(POINT_ARR, 0);    // создать пустой массив точек пересече-
ния
if (::ksIntersectCurvCurv(line, cir, array) > 0) {  // найти точки пересечения кривых
 int count = ::GetArrayCount(array);        // количество элементов в массиве
  char buf[128];
  ::sprintf(buf, "кол-во элементов = %i", ::GetArrayCount(arr));
  ::Message(buf);
  MathPointParam par;
  for (int i = 0; i < count; i) {
 if (::GetArrayItem(array, i, &par, sizeof(MathPointParam))) {
  ::sprintf(buf, "координаты %i-й точки : %g ; %g", i, par.x, par.y);
  ::Message(buf);
  }
 }
}
```

Perpendicular - пример использования

```
{
double xp, yp;
Perpendicular (-10, 10, -50, -50,
            50, 50, &xp, &yp);
}
```

TanLinePointCircle- пример использования

```
void TanLinePointCircle_Example (void) {

double x[2], y[2];
int k, i;

Circle  ( 60, 60,   30,  1);
Point   ( 20, 50,   1);

TanLinePointCircle (20, 50, 60, 60, 30, &k, x, y);

gprintf(Количество касаний = %2d, k);
```

```
if (k)
for (I=0; i<k; I) {
LineSeg (20, 50, x[i], y[i], 1); /* линия касания */
Point (x[i], y[i], 1); /* точка касания */
}

}; /* TanLinePointCircle_Example */
```

TanLineAngCircle - пример использования

```
void TanLineAngCircle_Example (void) {

double x[2], y[2];
int k, i;

Circle  ( 60, 60,   30,  1);

TanLineAngCircle (60, 60, 30, 45, &k, x, y);

gprintf(Количество касаний = %2d, k);

}; /* TanLineAngCircle_Example */
```

TanCircleCircle - пример использования

```
void TanCircleCircle_Example (void)
{
TAN Tang[4];
int k, i;

Circle(60,60,30,1);
Circle (0, 0, 50, 1);

TanCircleCircle (60, 60, 30, 0, 0, 50, &tang[]);

gprintf(Количество касаний = %2d, k);
if (k)
for (I=0; i<k; I)
```

```
{
LineSeg (tang.x1[i], tang.y1[i], tang.x2[i], tang.y2[i], 1); /* линия касания */
Point (tang.x1[i], tang.y1[i], 1); /* точка касания */
}
};
```

CouplingLineLine - пример использования

```
{
CON con[4];

Line(0, 0, 45);
Line(50, 50, 175);

CouplingLineLine (0, 0, 45, 50, 50, 175, 30, &kp, &con);
}
```

ksCouplingLineLine - пример использования

```
{
CON con[4];

Line(0, 0, 45);
Line(50, 50, 175);

ksCouplingLineLine (0, 0, 45, 50, 50, 175, 30, &kp, &con);
}
```

ksCouplingCircleCircle - пример использования

```
//поиск окружностей сопряжения для двух окружностей
  double rad = 20;
  int kp;
  CON con[8];
  Circle(100, 100, 100, 1);
  Circle(100, 150, 100, 1);
  ksCouplingCircleCircle(100, 100, 100, //параметры первой окружности
          100, 150, 100, //параметры второй окружности
          rad, &kp, con);
```

```
  //отрисуем окружности и точки сопряжения
  for (short i = 0 ; i<8; i) {
   Circle(con[i].xc, con[i].yc, rad,2);
   Point(con[i].x1,con[i].y1,i);
   Point(con[i].x2,con[i].y2,i);
  }
```

ksCouplingLineCircle - пример использования

```
//поиск окружностей сопряжения для прямой и окружности
  double rad = 20;
  int kp;
  CON con[8];
  Circle(100, 100, 100, 1);
  Line(100, 100, 45);
  ksCouplingLineCircle(100, 100, 100, //параметры окружности
          100, 100, 45,  //параметры линии
          rad, &kp, con);
  //отрисуем окружности и точки сопряжения
  for (short i = 0 ; i<8; i) {
   Circle(con[i].xc, con[i].yc, rad,2);
   Point(con[i].x1,con[i].y1,i);
   Point(con[i].x2,con[i].y2,i);
  }
```

ksTanLinePointCurve - пример использования

```
//поиск окружностей сопряжения для прямой и окружности
 RequestInfo info;
 ::memset(&info, 0, sizeof(info));
 info.commands = "Укажите объект";
 double x, y;
 int j;
 reference pArray = ::CreateArray(POINT_ARR, 0);

 do { //найдем объект
  j = ::Cursor(&info, &x ,&y, 0);
  if (j) {
```

```
  reference pObj = ::FindObj(x, y, 1e6);
  if(::ExistObj(pObj)){
   info.commands = "Укажите точку";
   if (::Cursor(&info, &x ,&y, 0)) {
    ::ksTanLinePointCurve(x, y,   // координаты точки
              pObj,   // указатель на кривую
              pArray); // массив точек касания
    MathPointParam par;       // Структура параметров математической точки
    ::memset(&par, 0, sizeof(MathPointParam));
    for(int i = 0, count = ::GetArrayCount(pArray); i < count; i) {
     ::GetArrayItem(pArray, i, &par, sizeof(MathPointParam));
     ::LineSeg(x, y, par.x, par.y, 1);
    }
    ::ClearArray(pArray);
   }
  }
  else
   ::Error("объект не найден");
 }
} while(j);
```

DistancePntPnt - пример использования

```
{
double len;
len = DistancePntPnt (10, 10, 50, 50);
}
```

ksDistancePntLineSeg - пример использования

```
Point (10, 10, 1);
 LineSeg(20,15,20,25,1);
 double len = ksDistancePntLineSeg (10, 10, //координаты точки
                 20, 15,   //первая точка отрезка
                 20, 25); //вторая точка отрезка
 char buf[128];
 sprintf(buf, "len = %5.1f", len);
 Message(buf);
```

ksDistancePntLineForPoint - пример использования

```
reference pObj;
RequestInfo info;
double x, y;
memset (&info, 0, sizeof(info));
info.prompt = "Укажите отрезок";
reference g;
int j = Cursor (&info, &x ,&y, 0);
if (j)
{
//найдем объект и убедимся, что это отрезок
if (ExistObj(pObj = FindObj (x, y, 1e6)) && GetObjParam (pObj, 0, 0, 0) == LINESEG_OBJ)
{
info.prompt = "Укажите точку";
//получим точку
j = Cursor (&info, &x ,&y, 0);
if (j)
{
LineSegParam par;
//берем параметры отрезка
GetObjParam (pObj, &par, sizeof (LineSegParam), ALLPARAM);
//получаем расстояние от точки до отрезка
double len = ksDistancePntLineForPoint (x, y,//координаты точки
par.x1, par.y1,//координаты точки на прямой
par.x2, par.y2);//координаты точки на прямой
char buf[TEXT_LENGTH];
sprintf (buf, "расстояние = %f ", len);
Message (buf);
}
}
else
Error("Это не отрезок");
}
```

ksDistancePntPntOnCurve - пример использования

```
double x, y;
if (::Cursor(NULL, &x, &y, NULL)) {     // указываем кривую
reference curve = ::FindObj(x, y, 1000);  // найти ближайшую кривую
 double x1, y1;
 if (::ExistObj(curve) && ::Cursor(NULL, &x, &y, NULL) // проверим, существует ли кривая
   && ::Cursor(NULL, &x1, &y1, NULL)) {  // и укажем две точки на кривой
double len = ::ksDistancePntPntOnCurve(curve, x, y, x1, y1); // расстояние между точками
  char buf[128];
  ::sprintf(buf, "Расстояние между точками = %g", len);
  ::Message(buf);              // выведем результат
 }
}
```

ksGetCurvePointProjection - пример использования

```
reference curve = ::LineSeg (300, 100, -400, -200, 1);
double x = 100;
double y = 50;
::ksGetCurvePointProjection (curve, x, y, &x, &y); // проецируем точку на кривую
char buf[128];
::sprintf (buf, "координаты проекции точки на кривой: (%g, %g)", x, y);
::Message(buf);
```

Angle - пример использования

```
void Angle_Example (void)
{
double Ang;

//Вычисление угла отрезка 0,0; 60,40
LineSeg ( 0, 0, 60, 40, 1);
Ang = Angle (0, 0, 60, 40);
gprintf(Угол = %6.2f, Ang);
};
```

ksGetCurvePerimeter, ksCalcInertiaProperties, ksCalcMassInertiaProperties - пример использования

```
reference pObj;
 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите кривую";
 int j = Cursor(&info, &x ,&y, 0);
 if (j) {
  if(ExistObj(pObj = FindObj(x, y, 1e6))){
   //узнаем тип объекта
   int type = GetObjParam( pObj,0,0,0); //указатель на графический объект
   if (type == CIRCLE_OBJ    ||  //окружность
      type == ARC_OBJ      ||  //дуга
      type == NURBS_OBJ     ||  //nurbs
      type == LINESEG_OBJ    ||  //отрезок
      type == BEZIER_OBJ    ||  //bezier
      type == CONTOUR_OBJ    ||  //контур
      type == POLYLINE_OBJ   ||  //полилиния
      type == ELLIPSE_OBJ    ||  //эллипс
      type == ELLIPSE_ARC_OBJ  ||  //дуга эллипса
      type == RECTANGLE_OBJ   ||  //прямоугольник
      type == REGULARPOLYGON_OBJ   //многоугольник
     ) {
    //периметр кривой
    double perimeter = ksGetCurvePerimeter(pObj, ST_MIX_MM); //указатель на кривую
    char buf[255];
    sprintf(buf, "Периметр = %f", perimeter);
    Message(buf);

    InertiaParam inertiaPar;
    //рассчитаем плоские моментно - центровочные характеристики кривой
    int rez = ksCalcInertiaProperties(pObj,    //указатель на кривую или группу кривых
                &inertiaPar, //указатель на структуру моментно-центровочных характеристик
                 ST_MIX_MM); //размерность в интервале [ST_MIX_SM..ST_MIX__M]
    if (rez) {
    sprintf(buf, "Центр xc = %f yc = %f площадь = %f", inertiaPar.xc, inertiaPar.yc, inertiaPar.f);
     Message(buf);
```

```
    sprintf(buf, "Моменты инерции относительно исходных осей x и y\nIx = %f Iy = %f Ixy = %f", inertiaPar.Ix, inertiaPar.Iy, inertiaPar.Ixy);
    Message(buf);
   sprintf(buf, "Центральные моменты инерции\nmx = %f my = %f mxy = %f", inertiaPar.mx, inertiaPar.my, inertiaPar.mxy);
    Message(buf);
      sprintf(buf, "Главные центральные моменты инерции\njx = %f jy = %f угол = %f", inertiaPar.jx, inertiaPar.jy, inertiaPar.a);
    Message(buf);
    MassInertiaParam mInertiaPar;
    //для тела выдавливания моментно-центровочные характеристики
    rez = ksCalcMassInertiaProperties(pObj,     // указатель на кривую или группу кривых
                &mInertiaPar, // указатель на структуру массово-центровочных характеристик
                 ST_MIX_MMIST_MIX_GRIST_MIX_EXT , // набор флагов
                 2.7,  // плотность (г/мм3)
                 100);  // если тело вращения-угол раствора в градусах, тело выдавливания - толщина
    if (rez) {
       sprintf(buf, "плотность = %f масса = %f объем = %f", mInertiaPar.r, mInertiaPar.m, mInertiaPar.v);
     Message(buf);
      sprintf(buf, "координаты центра тяжести\n xc= %f yc = %f zc = %f", mInertiaPar.xc, mInertiaPar.yc, mInertiaPar.zc);
     Message(buf);
         sprintf(buf, "центробежные моменты инерции\n Ixy= %f Ixz = %f Iyz = %f", mInertiaPar.Ixy, mInertiaPar.Ixz, mInertiaPar.Iyz);
     Message(buf);
        sprintf(buf, "осевые моменты инерции\n Ix= %f Iy = %f Iz = %f", mInertiaPar.Ix, mInertiaPar.Iy, mInertiaPar.Iz);
     Message(buf);
         sprintf(buf, "плоскостные моменты инерции\n jy0z= %f jx0z = %f Ix0y = %f", mInertiaPar.jy0z, mInertiaPar.jx0z, mInertiaPar.jx0y);
     Message(buf);
      sprintf(buf, "Главные центробежные моменты инерции\n jxy= %f jxz = %f jyz = %f", mInertiaPar.jxy, mInertiaPar.jxz, mInertiaPar.jyz);
     Message(buf);
    sprintf(buf, "Главные осевые моменты инерции \n jx= %f jy = %f jz = %f", mInertiaPar.jx, mInertiaPar.jy, mInertiaPar.jz);
     Message(buf);
    }
```

```
   }
   else
    Error ("Ошибка");
  }
  else
   Error("Выбранный объект не кривая");
 }
}
```

ksViewGetDensity - пример использования

```
char buf[128];

//вызываем диалог выбора плотности
 double density = ksViewGetDensity(0); // дескриптор окна родителя или NULL
 sprintf(buf, " densyty = %f г/мм3", density);

 Message(buf);
```

Пример использования функций работы с фрагментами

```
void ReadFragment_Example (void) {

PlacementParam  par;

par.xBase = 30;  par.yBase = 40;
par.ang   = 0;  par.scale = 1 ;

ReadFragment("c:\\kompas5\\1.frw", 0, &par);

//Пример вставки фрагмента  внешней ссылкой
//определим фрагмент для вставки

pDefFrg = FragmentDefinition("c:\\kompas5\\1.frw", "frw1", 1);
if(pDefFrg) {
PlacementParam  par;
par.xBase = 80;  par.yBase = 70;
```

```
par.ang   = 45;   par.scale = 2 ;
reference  pInsFrg =  InsertFragment(pDefFrg, 0, &par);
LightObj (pInsFrg, 1);
Message("вставка фрагмента внешней ссылкой");
LightObj (pInsFrg, 0);
}

// Пример вставки локального фрагмента
// определим фрагмент для вставки

reference pDefFrg = 0;

//определим локальный фрагмент
 if(LocalFragmentDefinition("local")) {
LineSeg(0, 0,  10, 0,  1);
LineSeg(0, 0,   0, 10, 1);
ArcByPoint(0, 0, 10, 10, 0, 0,10, -1, 1);
pDefFrg =  CloseLocalFragmentDefinition();
}

if(pDefFrg) {
PlacementParam  par;
par.xBase = 100; par.yBase = 40;
par.ang   = 45;  par.scale = 1 ;
reference  p =  InsertFragment(pDefFrg, 0, &par);
LightObj (p, 1);
Message("вставка локального фрагмента");
LightObj (p, 0);
}

}; /* ReadFragment */
```

WriteFragment - Пример использования

```
//Пример использования WriteFragment

    double x, y, x1,y1;
    RequestInfo info;
```

```
    memset(&info, 0, sizeof(info));
    info.prompt = "Укажите начальную точку окна";
    if (Cursor(&info, &x, &y, 0))
    {
    info.prompt = "Укажите конечную точку окна";
 if (Cursor(&info, &x1, &y1, 0))
{
//создадим рабочую группу
reference gr = NewGroup(0);
        EndGroup();
        if (SelectGroup(gr, 1 ,x, y,  x1, y1))
{
          LightObj(gr, 1);//подсветим выделенную группу
          char name[128];
          //запросить имя файла фрагмента
          if(SaveFile("*.frw",0,0, name,128))
{
            info.prompt = "Укажите точку привязки фрагмента";
            if (Cursor(&info, &x, &y, 0))
{
//записать содержимое группы во фрагмент
WriteFragment(gr,  name, "фрагмент из рабочей группы" ,x, y);
              Message("Фрагмент записан");
            }
          }
          LightObj(gr, 0);//снять подсветку выделенной группы
        }
      }
    }
```

ksReadFragmentToGroup, ksChoiceFile - Пример использования

```
char frwName[250];
int j1;
do
{
//выберем  фрагмент
```

```
if ((j1 = ksChoiceFile ("*.frw","фрагменты(*.frw)|*.frw|Все файлы (*.*)|*.*|",
frwName,128,1))!=0)
{
double x, y;

//подготовим структуры фантома и запросов для Placement
struct Phantom  rub;
rub.type1.xBase = 0;
rub.type1.yBase = 0;
rub.type1.scale = 1;
rub.phType = 1;

//во временную группу положим вставку фрагмента,
//взятую из библиотеки фрагментов
PlacementParam  par;
par.xBase = 0;
par.yBase = 0;
par.ang = 0;
par.scale = 1 ;
int j;
do
{
//если нужно вставить несколько фрагментов,
//группу лучше порождать новую,
//так как вместе с геометрией могут прийти атрибуты,
//объекты спецификации, стили, которые связаны с геометрией.
//При простом копировании группы эта связь будет потеряна.

rub.type1.gr  = ksReadFragmentToGroup (frwName,// имя фрагмента
0,// на свои слои,
&par);//параметры привязки

if (rub.type1.gr && (j = Placement(NULL, &x, &y, &rub.type1.ang, &rub))!=0)
{
//сдвигаем группу
MoveObj(rub.type1.gr, x, y);
//поворачиваем группу
if (fabs(rub.type1.ang) > 0.001)
```

```
RotateObj (rub.type1.gr, x, y, rub.type1.ang);
//ставим группу в модель
StoreTmpGroup (rub.type1.gr);
ClearGroup (rub.type1.gr);
DeleteObj (rub.type1.gr);
}
else
{
if (rub.type1.gr)
DeleteObj(rub.type1.gr);
j = 0;
}
}
while (j);
}
}
while(j1);
```

ksChoiceFragmentFromLib - Пример использования

```
char libName[250];
int j1;

//выберем библиотеку фрагментов
if (ChoiceFile ("*.lfr","Библиотеки фрагментов(*.lfr)|*.lfr|Все файлы (*.*)|*.*|", libName, 128))
{
char buf[250];
do
{
//выбрать фрагмент в библиотеке фрагментов
if ((j1 = ksChoiceFragmentFromLib(libName, buf, 250))!=0)
{
//выделим имя вставки
char * insertName = strrchr (buf, '|');
if (insertName)
{
double x, y;
```

```
//подготовим структуры фантома и запросов для Placement
struct Phantom rub;
rub.type1.xBase = 0;
rub.type1.yBase = 0;
rub.type1.scale = 1;
rub.phType   = 1;

reference pDefFrg;

// создадим описание вставки фрагментов
pDefFrg = FragmentDefinition (buf,//имя файла фрагмента
insertName1,//имя вставки
1);// внешней ссылкой

if (pDefFrg)
{
//во временную группу положим вставку фрагмента,
//взятую из библиотеки фрагментов
rub.type1.gr = NewGroup (1);// временная группа

PlacementParam par;
par.xBase = 0;
par.yBase = 0;
par.ang  = 0;
par.scale = 1 ;
//создаем объект "вставка фрагмента"
reference p = InsertFragment (pDefFrg,// Указатель
//определения фрагмента
0,// на свои слои
&par);//параметры привязки
EndGroup();
int j;
do
{
rub.type1.ang = 0;
//запрос точки и угла поворота фрагмента
if ((j = Placement (NULL, &x, &y, &rub.type1.ang, &rub))!=0)
{
```

```
CopyObj (p,// указатель на графический объект
0,0,// базовая точка объекта
x, y,// точка, в которую копировать
1,// масштаб
rub.type1.ang );// угол поворота а градусах
}
}
while (j);
DeleteObj (rub.type1.gr);
}
else
Error ("ошибка создания описания вставки фрагмента");
}
else
Error("имя вставки не определено");
}
}
while(j1);
}
```

ksExistFragmentInLibrary - Пример использования

```
char frwName [250];
 strcpy (frwName, "C:\\0\\Детали.lfr|ФланцыИсполнение 1");
 if (ReadString ("Введите имя фрагмента или папки",  // строка приглашения
        frwName, 250)) {

  int j = ksExistFragmentInLibrary(frwName);
  char buf [250];

  sprintf (buf, "%s\n%s",frwName, j==0 ? "нет фрагмента или папки" : j==-1 ? "нет библиоте-
ки" : "фрагмент или папка есть");
  Message(buf);
 }
```

ksFragmentLibrary - Пример использования

```
char libName[250];
```

```
int j1;
//выберем библиотеку фрагментов
if (ChoiceFile ("*.lfr","Библиотеки фрагментов(*.lfr)|*.lfr|Все файлы (*.*)|*.*|", libName,
128))
{
char buf[250];
do
{
//выбрать фрагмент в библиотеке фрагментов
if ((j1 = ksChoiceFragmentFromLib(libName, buf, 250))!=0)
{
//выделим имя вставки
char * insertName = strrchr (buf, '|');
if (insertName)
{
double x, y;
//подготовим структуры фантома и запросов для Placement
struct Phantom rub;
rub.type1.xBase = 0;
rub.type1.yBase = 0;
rub.type1.scale = 1;
rub.phType = 1;
reference pDefFrg;
// создадим описание вставки фрагментов
pDefFrg = FragmentDefinition (buf,//имя файла фрагмента
insertName1,//имя вставки
1);// внешней ссылкой
if (pDefFrg)
{
//во временную группу положим вставку фрагмента,
//взятую из библиотеки фрагментов
rub.type1.gr = NewGroup (1);// временная группа
PlacementParam par;
par.xBase = 0;
par.yBase = 0;
par.ang  = 0;
par.scale = 1 ;
//создаем объект "вставка фрагмента"
```

```
reference p = InsertFragment (pDefFrg,// указатель определения
//фрагмента
0,//на свои слои
&par);//параметры привязки
EndGroup();
int j;
do
{
rub.type1.ang = 0;
//запрос точки и угла поворота фрагмента
if ((j = Placement(NULL, &x, &y, &rub.type1.ang, &rub))!=0)
{
CopyObj (p,// указатель на графический объект
0,0,// базовая точка объекта
x, y,//точка, куда копировать
1,// масштаб
rub.type1.ang);// угол поворота а градусах
}
}
while (j);
DeleteObj (rub.type1.gr);
}
else
Error ("Ошибка создания описания вставки фрагмента");
}
else
Error("Имя вставки не определено");
}
}
while (j1);
}
```

ksAddFragmentToLibrary - Пример использования

```
char libName[250];
char buf [250];
//выберем библиотеку фрагментов
```

```
if (ChoiceFile("*.lfr","Библиотеки фрагментов(*.lfr)|*.lfr|Все файлы (*.*)|*.*", libName, 128))
{
RequestInfo info;
memset(&info, 0, sizeof(info));
info.commands = "!Новый_фрагмент !Редактировать_фрагмент !Удалить_фрагмент ";
int j;
int typeEdit;
string nameFrg;
do
{
j = CommandWindow(&info);
switch (j)
{
case 1://!Новый_фрагмент

if (ReadString("Введите имя нового фрагмента",// строка приглашения
buf,//возвращаемая строка
250))// допустимая длина
// строки
{
nameFrg = libName;
if (buf[0] != '|')
nameFrg = "|";
nameFrg = buf;
typeEdit = 2;//запустить на редактирование
}
else
typeEdit = 0;
break;

case 2://Редактировать_фрагмент

case 3://Удалить_фрагмент

//выберем имя файла фрагмента
if ((j == 2 || j == 3) && ksChoiceFragmentFromLib(libName, buf, 250)!=0)
{
```

```
nameFrg = buf;
typeEdit = j;// 2- запустить на редактирование, 3-удалить
}
else
typeEdit = 0;

break;
}
if (j > 0 && typeEdit)
{
if (::ksFragmentLibrary((char*)nameFrg.c_str(), typeEdit))
{
if (typeEdit == 2)
{
::ksFragmentLibrary ((char *)nameFrg.c_str(), 4);
//минимизировать окно библиотекаря
//редактируем фрагмент из библиотеки
Text (0, 100,//точка привязки текста
0,//угол наклона текста
5,//высота текста
1,//сужение текста
0,//свойства строки
"Редактируем фрагмент из библиотеки");//строка символов

LineSeg (0, 100, 110, 100, 1);
//редактируем фрагмент в интерактивном режиме
//после выбора в меню "Сервис" команды
//"Закончить редактирование фрагмента",
//возвращаемся в библиотеку
::SystemControlStart("Закончить редактирование фрагмента");
::ksFragmentLibrary((char *)nameFrg.c_str(), 0); //закрыть
//с сохранением
}
}
else
MessageBoxResult();
}
}
```

```
while (j != -1);
}
```

ksCheckFragmentLibrary - Пример использования

```
if (!ksCheckFragmentLibrary ("1.lfr",// имя файла библиотеки фрагментов
1))// сообщать, если файл уже открыт
Message ("Библиотека 1.lfr из текущей папки не загружена");
```

Point - Пример использования

```
void Point_Example (void)
{
// задание точек
Point (10, 10, 1);
Point (20, 10, 2);
};
```

PointArraw - Пример использования

```
{
Macro (0);
LineSeg (50, 100, 100, 100, 2);
PointArraw (50,100,0 /*ang*/,1 /* стрелка изнутри*/);
PointArraw (100,100,180/*ang*/,1 /* стрелка изнутри*/);
reference pMacro = EndObj();
LightObj (pMacro, 1);
Message ("Имитация размерной линии");
LightObj (pMacro, 0);
}
```

ksPointsOnCurveByStep - Пример использования

```
double x = 100, y = 100, step = 10.5;
    // указанная кривая
    reference pObj = ::FindObj(x, y, 1000);

    if (::ExistObj(pObj)) {      // проверим существование объекта
```

```
        reference arr = ::ksPointsOnCurveByStep(pObj, step); // массив точек
        uint count    = ::GetArrayCount(arr);               // количество полученных точек

MathPointParam point;
// нарисуем точки
        for (uint i = 0; i < count; i++)
          if (::GetArrayItem(arr, i, &point, sizeof(MathPointParam)))
            ::Point(point.x, point.y, 4);
    }
```

Line - Пример использования

```
void Line_Example (void)
{
reference l1, l2, l3, l4 ;
double B=100, H=40;

// задание четырех прямых прямоугольника длиной B и высотой H
l1 = Line (0, 0, 90);
l2 = Line (0, H,  0);
l3 = Line (B, H, 90);
l4 = Line (B, 0,  0);
};
```

LineSeg - Пример использования

```
void LineSeg_Example (void)
{
int t = 1;//сплошная основная
reference l1, l2, l3, l4 ;
double B=100, H=40;

//задание 4-х отрезков прямоугольника
//длиной B и высотой H
l1 = LineSeg (0, 0, 0, H, t);
l2 = LineSeg (0, H, B, H, t);
l3 = LineSeg (B, H, B, 0, t);
```

```
l4 = LineSeg (B, 0, 0, 0, t);
};
```

Circle - Пример использования

```
void Circle_Example (void)
{
int tc = 1;/сплошная основная
int tl = 3;/осевая линия
reference l1, l2, c1, c2 ;
double x=30, y=20;

double rad1=5; //радиус внутренней окружности
double rad2=8; //радиус внешней окружности

//задание осевых линий
l1 = LineSeg (x-rad2-5, y,x+rad2+5, y, tl);
l2 = LineSeg (x, y-rad2-5, x, y+rad2+5, tl);

//задание окружностей
c1 = Circle (x, y, rad1, tc);
c2 = Circle (x, y, rad2, tc);
};
```

Arc - Пример использования

```
void Arc_Example (void)
{
int t1 = 1;// сплошная основная
int t2 = 2; // сплошная тонкая

//задание дуги
ArcByAngle (50, 50, 40, 0, 90, 1, t1);

//задание дуги по трем точкам
ArcBy3Points (80, 50, 50, 80, 50, 50, t2);

// задание дуги по центру, радиусу и 2-м точкам
```

```
ArcByPoint (50, 50, 100, 50, 50, 100, 1, t2);
};
```

ksConicArc - Пример использования

```
ConicArcParam par;
//заполним структуру канонического уравнения для конического сечения- дуги эллипса
// A*x*x B*x*y C*y*y D*x E*y F = 0;
par.A = 16.;
par.B = 0;
par.C = 9.;
par.D = 0;
par.E = 0;
par.F = -144.;
par.x1 = 3; par.y1 =0;
par.x2 = -3; par.y2 = 0;
par.style = 2;
reference p = ksConicArc(& par);
//подсветим объект
LightObj (p, 1);
Message ("Коническое сечение");
//выключим подсветку объекта
LightObj (p, 0);
```

ksEllipseArc - Пример использования

```
EllipseArcParam par;
  memset(&info, 0, sizeof(info));
  par.xc  = 20;
  par.yc  = 20;    //координаты центра
  par.a   = 30;
  par.b   = 10;             //полуоси эллипса
  par.ang = 45;               //угол наклона эллипса к оси X
  par.angFirst = 10;         //начальный угол дуги
  par.angSecond =300;        //конечный угол  дуги
  par.dir = 1;               //направление
  par.style = 1;    //тип линии
  reference p = KsEllipseArc(&par);
```

```
 LightObj (p, 1);  ConicArcParam par;
```

ksRectangle - Пример использования

```
RectangleParam par; // структура параметров прямоугольника

 ::memset(&par, 0, sizeof(RectangleParam));

 par.x      = -73.55; // базовая точка 1
 par.y      = 39.95;  //
 par.ang    = 0.00; // угол вектора направления от 1-ой точки ко 2-ой
 par.height  = -66.68; // высота
 par.weight  = 79.90; // ширина
 par.pCorner = ::CreateArray(CORNER_ARR, 0);  // Создание массива параметров углов
 par.style = 1; // стиль линии
 ::ksRectangle(&par, 0); // создать прямоугольник
```

ksColouring - Пример использования

```
//заливка цветом
Mtr(30,20,0,0.5);
char buf [128];

//построить заштрихованный квадрат
LineSeg (20, 30, 70, 30, 2);
LineSeg (70, 30, 70, 80, 2);
LineSeg (70, 80, 20, 80, 2);
LineSeg (20, 80, 20, 30, 2);

// залить квадрат цветом
if(ksColouring(RGB(168,0,168))) {
 LineSeg (20, 30, 70, 30, 2);
 LineSeg (70, 30, 70, 80, 2);
 LineSeg (70, 80, 20, 80, 2);
 LineSeg (20, 80, 20, 30, 2);
 reference p = EndObj();

 //взять параметры заливки -  цвет
```

```
  unsigned long   col;
  int t = GetObjParam(p, &col, sizeof(col), ALLPARAM);
  sprintf(buf, " t = %d, color=%d ", t, col);
  Message(buf);
  DeleteMtr();

  Mtr(0,0,0,2);

  col =  RGB(0, 255, 255);
  //изменить параметры заливки - цвет
  if(SetObjParam(p, &col, sizeof(col), ALLPARAM))
    Message(" Изменили объект");
  else
    MessageBoxResult();
  DeleteMtr();
}
else
  MessageBoxResult();
```

ksColouringEx - Пример использования

```
double x = 0; double y = 0;
// Указать точку в замкнутой области
if ( Cursor( NULL, &x, &y, NULL ) ) {
  // Создать контур из объектов окружающих данную точку
    reference group = ksMakeEncloseContours( 0, x, y );
    if ( group ) {
  // Создать заливку для группы
      ksColouringEx( RGB(255, 0, 0), group );
      // Удалить временную группу
DeleteObj( group );
    }
  }
```

ksEllipse - Пример использования

```
  EllipseParam  par;
  par.xc  = 20;
  par.yc  = 20;    //координаты центра
```

```
  par.a   = 30;
  par.b   = 10;            //полуоси эллипса
  par.ang = 45;        //угол наклона эллипса к оси X
  par.style = 1;    //тип линии

  reference p = ksEllipse(&par);
  LightObj (p, 1);
```

Equidistant - Пример использования

```
reference pContour;
RequestInfo  info;
//обнулить структуру  info;
memset (&info, 0, sizeof(info));
double x, y;
info.prompt = "Укажите контур";
int j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj (pContour = FindObj (x, y, 1e6)) &&
GetObjParam (pContour,  0, 0, 0) == CONTOUR_OBJ) {

//если нашли объект и этот объект - контур, строим эквидистанту

     EquidistantParam  equidParam;
     //обнулить структуру  info;
     memset(&equidParam, 0, sizeof(equidParam));
     equidParam.geoObj   = pContour; // геометрический объект
//-базовая кривая эквидистанты
     equidParam.side     = 0 ;     // признак, с какой стороны строить эквидистанту
    equidParam.cutMode  = 1;  // тип обхода углов контура: 0-обход срезом, 1- обход дугой
     equidParam.degState = 0; // флаг разрешения вырожденных сегментов эквидистанты:
// 0-вырожденные сегменты запрещены,
//1-вырожденные сегменты разрешены
     equidParam.radRight = 1; // радиус эквидистанты справа по направлению кривой
     equidParam.radLeft  = 1;  // радиус эквидистанты слева
     equidParam.style    = 1;    // тип линии
```

```
    for (int i = 0; i < 4; i++) {
      equidParam.radRight += i;
      equidParam.radLeft += i;
      Equidistant(&equidParam); //параметры эквидистанты
    }
  }
}
```

ksRegularPolygon - Пример использования

```
RegularPolygonParam par;
par.count= 8;                    // количество вершин многоугольника
par.xc= 100;                     // центр окружности
par.yc= 100;
par.ang= 35;                     // угол первой вершины
par.radius= 40;                  // радиус окружности
par.describe= 1;                 // признак описанного многоугольника.
par.style= 1;                    // стиль линии
par.pCorner= ::CreateArray (CORNER_ARR, 0); // динамический массив
// структур параметров углов CORNER_ARR

CornerParam cpar;                // структура параметров угла
cpar.index = 5;                  // индекс угла
cpar.fillet = 0;                 // признак фаски
cpar.l1 = 40;                    // длина фаски 1 сегмента
cpar.l2 = 45;                    // длина фаски 2 сегмента
::AddArrayItem(par.pCorner, -1, &cpar, sizeof(CornerParam)); // добавить угол в массив

cpar.index = 2;                  // индекс угла
cpar.fillet = 1;                 // признак скругления
cpar.l1 = 20;                    // радиус
cpar.l2 = 20;                    // радиус
::AddArrayItem(par.pCorner, -1, &cpar, sizeof(CornerParam)); // добавить угол в массив

::ksRegularPolygon(&par, 3);// создаётся многоугольник с осями
```

AnnLineSeg, AnnArcByPoint - Пример использования

```
//аннотационные отрезок и дуга

//аннотационный отрезок с большими стрелками изнутри (7мм)
reference p = AnnLineSeg(10, 10, 20,20, 6, 6,  1);
Message("аннотационный отрезок");
//трансформируем объект
Mtr(-10,-10,0, 2);
TransformObj(p);
DeleteMtr();
Message("удалим");
DeleteObj(p);

//аннотационная дуга с большими стрелками изнутри (7мм)
reference p = AnnArcByPoint (30, 40, 20,  30, 20, 50, 40, -1, 6, 6, 1);
Message ("аннотационная дуга");
//трансформируем объект
Mtr (-10,-10,0, 2);
TransformObj (p);

DeleteMtr ();
Message ("удалим");
DeleteObj (p);
```

ksInsertRaster - Пример использования

```
RasterParam par;// параметры растрового обекта
    par.place.xBase = 100;// точка привязки
    par.place.yBase = 100;
    par.place.ang   = -20;// угол наклона
    par.place.scale = 0.5;// масштаб
    ::strcpy (par.fileName, "C:\\BROWNBEAR.jpg");// имя файла растра
    ::ksInsertRaster (&par);// вставка растрового объекта
   reference iter = ::CreateIterator (RASTER_OBJ, 0);// создать итератор по растровым объ-
ектам
    if (iter) {
      reference obj = ::MoveIterator (iter, 'F');// смещаемся на первый объект
      if (::ExistObj(obj)) {// если объект существует
```

```
        Message("Меняем файл");// сообщение
        ::GetObjParam(obj, &par, sizeof(RasterParam), ALLPARAM);
// считываем параметры объекта
        ::strcpy(par.fileName, "C:\\WHITEBEAR.jpg");// имя нового файла растра
        par.place.ang   = 0;// угол наклона
        par.place.scale = 0;// масштаб
        ::SetObjParam (obj, &par, sizeof(RasterParam), ALLPARAM);
// изменяем параметры объекта
      }
      ::DeleteIterator (iter);                  // удаляем итератор
    }
```

Hatch - Пример использования

```
void Hatch_Example (void) {

reference p;

LineSeg (10,  10,  10,  20,  1);
LineSeg (10,  20,   40, 20,  1);
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

Hatch(1, 45, 3, 0, 0,0); /* определение штриховки */

LineSeg (10,  10,  10,  20,  1);
LineSeg (10,  20,   40, 20,  1);
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

p = EndObj();   /* закончить формирование штриховки */

}; /* Hatch_Example */
```

ksCreateViewObject, ksEditViewObject - Пример использования

```
//  создать объект в интерактивном режиме (позиционная линия выноски)
  reference posLeader = ksCreateViewObject(POSLEADER_OBJ );
  if (posLeader){
    LightObj(posLeader, 1);
    Message("Позиционная линия выноски");
    LightObj(posLeader, 0);
    if (ksEditViewObject(posLeader)) {
      LightObj(posLeader, 1);
      Message("Отредактированная позиционная линия выноски");
      LightObj(posLeader, 0);
    }
  }
```

Bezier, _Bezier - пример использования

```
void Bezier_Example (void)
{
double x[5]={15, 35, 55, 75, 95 };
double y[5]={15, 75, 25, 95, 15 };

Bezier (1, 0);// определение кривой

Point (10, 10, 1);
Point (30, 70, 1);
Point (50, 20, 1);
Point (70, 90, 1);
Point (90, 10, 1);

EndObj ();// закончить формирование кривой

// задание производных в узлах
BezierPointParam par;
double x[]  = { 0, 20, 50, 70, 100, 50 };
double y[]  = { 0, 20, 10, 20, 0, -50 };
double ang[] = { 0, 30, 40, 45, -45, -30 };
double left[] = { 1, 1, 2, 3, 1, 1 };
double right[]= { 2, 2, 1, 1, 3, 4 };
```

```
reference p = Bezier (0, 1);
for (int i=0; i<5; i)
{
par.x = x[i]; par.y = y[i];
par.ang = ang[i];
par.left = left[i]; par.right = right[i];
BezierPoint(&par);
}
EndObj();// закончить формирование кривой

_Bezier (&par, 6, 0, 1);
};
```

Nurbs, NurbsPoint - пример использования

```
void Nurbs_Example (void) {

static NurbsPointParam par[]={{ 0,0,1}, {20,20,1}, {50,10,1}, {70,20,1}, {100,0,1}, {50,-50,1}};

//построить Nurbs-кривую
if (Nurbs(3, 0, 1))
for (int i=0; i<6; i) {
NurbsPoint(&par[i]);
}
reference p = EndObj();

LightObj(p, 1);
Message(NURBS);
LightObj(p, 0);
}
}; /* Nurbs_Example */
```

ksNurbsKnot - пример использования

```
static NurbsPointParam par[]= {{ 0,0,1}, {20,20,1}, {50,10,1}, {70,20,1} };
static double   knotArr []= { 0, 0 ,0 ,1 ,2 ,2 ,2 };
```

```
//---------------------------------------------------------------------------
// Создать Nurbs - сплайн
//---------------------------------------------------------------------------
void DrawNurbs() {
 //построить Nurbs сплайн как составной объект
 Nurbs(3, 0, 1);
  //ввод точек
  for (int i=0; i<4; i) {
   NurbsPoint(&par[i]);
  }
  //ввод узлов - для разомкнутого сплайна и степени 3 должно быть 7 узлов
  for (int i=0; i<7; i) {
   ksNurbsKnot(knotArr[i]);
  }

 reference p = EndObj();
 LightObj(p, 1);
 Message("NURBS");
 LightObj(p, 0);
}
```

NurbsForConicCurve - пример использования

```
// построить дугу эллипса (параметры эллипса: центр - 0,0, a = 20, b= 10);
double x[4], y[4];
x[0] = -19.3202; y[0] = 2.5850;// начальная точка эллиптической дуги (1)
x[1] = -10.0;  y[1] = 20.0;  // пересечение касательных к дуге из точкек 1 и 2
x[2] = 14.6144; y[2] = 6.8268; // конечная точка эллиптической дуги (2)
x[3] = 0.0;  y[3] = 10.0;  // точка на дуге
reference p = NurbsForConicCurve (x, y, 1);
if (p)
{
LightObj (p, 1);
Message ("Эллиптическая дуга построена");
LightObj(p, 0);
}
else
Error ("Неверно заданы характерные точки");
```

ksPolyline, _ksPolyline - пример использования

```
void ksPolyline_Example (void) {

static MathPointParam  pr1[] = { {10,10}, {20,20}, {30, 10}, {40,20} };
static MathPointParam  pr2[] = { {10,15}, {20,25}, {30, 15}, {40,25} };
reference p;

if(ksPolyline(1)) {
for(int i = 0; i < 4; i)
Point(pr1[i].x, pr1[i].y, 1);
p = EndObj();

LightObj(p, 1);
Message(Полилиния);
LightObj(p, 0);
}
PolylineParam par;
//создадим массив неопределенной длины для математических точек
par.pMathPoint = CreateArray(POINT_ARR , 0);
//наполним массив
for(int i = 0; i < 4; i)
AddArrayItem(pMathPoint, -1, &pr2[i], sizeof(MathPointParam));

par.style = 2;
//создадим полилинию (тип линии - тонкая)
reference p = _ksPolyline(&par);

LightObj(p, 1);
Message(Полилиния);
LightObj(p, 0);

 }; /* ksPolyline_Example */
```

ksPolylineEx - пример использования

```
  PolylineParamEx par;
```

```
  ::memset(&par, 0, sizeof(PolylineParamEx));
  par.pMathPoint = ::CreateArray(POINT_ARR, 0);
  MathPointParam p;
  p.x = 0;
  p.y = 0;
  ::AddArrayItem(par.pMathPoint, -1, &p, sizeof(p));
  p.x = 100;
  p.y = 0;
  ::AddArrayItem(par.pMathPoint, -1, &p, sizeof(p));
  p.x = 150;
  p.y = 100;
  ::AddArrayItem(par.pMathPoint, -1, &p, sizeof(p));
  p.x = 0;
  p.y = 100;
  ::AddArrayItem(par.pMathPoint, -1, &p, sizeof(p));

  par.style   = 1;
  par.closed   = 1;
  reference line = ::ksPolylineEx(&par);
  if (line) {
  ::Message("Изменим стиль");
  ::memset(&par, 0, sizeof(PolylineParamEx));
   ::GetObjParam(line, &par, sizeof(par), ALLPARAM);
  par.style = 3;
   ::SetObjParam(line, &par, sizeof(par), ALLPARAM);
  }
```

Mtr, DeleteMtr - Пример использования

```
void Mtr_Example (void) {
int t = 1;   /*сплошная основная */
reference l1, l2, l3, l4 ;
double B=100, H=40;

      /* задание 4-х отрезков прямоугольника */
      /* длиной B и высотой H                */
      /* в системе координат с началом 50,50, */
      /* и поворотом 45 градусов             */
```

```
Mtr (50, 50, 45, 1);

l1 = LineSeg (0, 0, 0, H, t);
l2 = LineSeg (0, H, B, H, t);
l3 = LineSeg (B, H, B, 0, t);
l4 = LineSeg (B, 0, 0, 0, t);

DeleteMtr ();

}; /* Mtr_Example */
```

ksMtr - пример использования

```
reference gr = NewGroup (0);
LineSeg (50, 50, 100, 100, 1);
LineSeg (100, 100, 150, 50, 1);
Circle (50, 50, 10, 1);
Circle (150, 50, 10, 1);
EndGroup ();
//масштаб по оси X - 2 и по оси Y - 1
ksMtr (0, 0, 0, 2, 1);
TransformObj (gr);
DeleteMtr();
```

MtrForIGES - Пример использования

```
double commonArray[2][2]; // матрица 2*2 поворота,
                          //умноженная предварительно на масштаб
  double moveArray[2];      // матрица сдвига умноженная предварительно на масштаб
// заполним матрицы
  commonArray[0][0] = 2.0;
  commonArray[0][1] = 0.0;
  commonArray[1][0] = 0.0;
  commonArray[1][1] = -2.0;
  moveArray[0] = 20.0;   // сдвиг по X
  moveArray[1] = 0.0;   // сдвиг по Y
  LineSeg(0, 0, 10, 10, 1);
```

```
//создание матрицы трансформации - симметрия по X и с масштабом 2 и сдвиг на 10мм

 if (::MtrForIGES(commonArray,  moveArray)) {
   LineSeg(0, 0, 10, 10, 1);
   DeleteMtr();
 }
```

IsGeomObject - Пример использования

```
reference itAllObj = CreateIterator( ALL_OBJ,   //тип поиска объекта
                          0 );     //указатель на объект (для движения по группе и внутри макро)
 if ( itAllObj )  {
   reference obj;
   if ( ExistObj( obj = MoveIterator( itAllObj, 'F' )) ){
     do {
       // проверим геометрический объект или нет
       if ( IsGeomObject(obj) )
         Message( "геометрический объект" );

       // проверим кому принадлежит объект
       if ( IsObjFromAssociativeView(obj) )
         Message( "Объект из ассоциативного вида" );

       // подсветим все объекты имеющие видимые и невидимые участки
       if ( IsVisibleOrHiddenArraysInObject(obj) )
         LightObj( obj, 1 );


     } while( ExistObj ( obj = MoveIterator( itAllObj, 'N' )) );
   }
 }
```

Text, TextLine, Paragraph - Пример использования

```
void Text_Example (void) {
reference p;

// Простой текст
```

```
Text (30 , 90, 0, 5, 1, 0, ”Простой текст”);
Text (30 , 80, 0, 5, 1, 0,”Пример... дроби$dЧислитель;Знаменатель$”);
Text (30 , 70, 0, 5, 1, 0, ”Пример... отклонений 20$0.5;-0.3$”);
Text (30 , 60, 0, 5, 1, 0, ”Пример... спецсимвола &32”);

// Структурированный текст

ParagraphParam par;
par.x = 30; par.y = 40; par.ang = 0;
par.hFormat = 0; par.vFormat = 0;
par.height = 25; par.width = 20;
int j=2;

p = Paragraph (&par);

TextLine (NEW_LINE ,0,0 , ”Первая строка”);

//числитель, наклон, высота дроби в 1.5 раза меньше высоты текста
TextLine (NUMERATOR|ITALIC_ON,FRACTION_TYPE ,&j , ”Числитель”);
//знаменатель, утолщение
TextLine (DENOMINATOR|BOLD_ON,0 ,0 , ”Знаменатель”);
//конец дроби, снятие утолщения,снятие наклона
TextLine (END_FRACTION|BOLD_OFF | ITALIC_OFF, 0, 0 , ”4444”);

 TextLine (NEW_LINE ,0,0 , ”Вторая строка”);
int tip = 65;
//спецзнак шероховатость
TextLine (SPECIAL_SYMBOL ,SPECIAL, &tip , ”Rz80”);
//конец спецзнака
TextLine (SPECIAL_SYMBOL_END ,0, 0 , ”222”);

 EndObj();

}; /* Text_Example */
```

GetTextLength - Пример использования

```
/*GetTextLenght_example*/
{
reference pText;
double   len1, len2;

len1 = GetTextLenght(« Пример... текста», 1);  // Пример... функции GetTextLenght

pText = Text (30 , 90, 0, 5, 1, 0,Простой текст);
len2 = GetTextLenghtFromReference(pText); // Пример... функции GetTextLenghtFromReference

}
```

ksConvertTextToCurve - Пример использования

```
reference pText;
  RequestInfo info;
  //обнулить структуру info;
  memset (&info, 0, sizeof (info));
  double x, y;
  info.prompt = "Укажите текст";
  int j = Cursor (&info, &x ,&y, 0);
  if (j) {
   if (ExistObj (pText = FindObj (x, y, 1e6)) &&
    GetObjParam (pText, 0, 0, 0) == TEXT_OBJ) {
    reference gr = ksConvertTextToCurve (pText);
    if (gr) {
     Phantom phantom;   // ltdefine.h
     memset (&phantom, 0, sizeof (phantom));
     phantom.type1.gr = gr; //временная группа
     phantom.phType = 1; //сдвиг группы
     phantom.type1.scale = 1; //сдвиг группы
     j = Cursor (&info, &x ,&y, &phantom);
     if (j == -1) { //поставить в модель
      MoveObj (phantom.type1.gr, x, y); //смещаем группу в новый  цетр
      StoreTmpGroup (phantom.type1.gr);//временную группу делаем постоянной
```

```
      ClearGroup (phantom.type1.gr);
      DeleteObj (phantom.type1.gr);
     }
    }
   }
   else
    Error ("Это не текст");
  }
```

Table - Пример использования

```
void Table_Example (void)
{
reference p;
p = Table ();
LineSeg(50, 50, 90, 50, 1);
LineSeg(50, 40, 90, 40, 1);
LineSeg(50, 30, 90, 30, 1);
LineSeg(50, 50, 50, 30, 1);
LineSeg(70, 50, 70, 30, 1);
LineSeg(90, 50, 90, 30, 1);

Text(52, 48, 0, 5, 1,0, Простой текст);//первая ячейка по точке привязки

ParagraphParam par;
par.x = 72; par.y = 48;//вторая ячейка
par.ang = 0; par.hToleranceat = 0; par.vToleranceat = 0;
par.height = 20;//высота блока форматирования
par.width = 20;//ширина блока форматирования

Paragraph (&par);
int j=2;
TextLine (NEW_LINE, 0, 0, Сложный);
TextLine (NEW_LINE|BOLD_ON, 0, 0, структурированный);
TextLine (NEW_LINE|BOLD_OFF | ITALIC_ON, 0, 0, текст);
EndObj();
EndObj();
};
```

ksSetTableColumnText - Пример использования

```
// редактирование таблицы
reference pObj;

RequestInfo info;
double x, y;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите таблицу";

// взять таблицу на чертеже
int j = Cursor(&info, &x ,&y, 0);
if (j)
{
if (ExistObj (pObj = FindObj (x, y, 100000)))
{
//узнаем тип объекта
int type =GetObjParam (pObj,0,0,0);//указатель на графический объект
//проверить, таблица ли полученный объект
if (type == TABLE_OBJ)
{
unsigned int numb;
reference p;
//открыть таблицу для редактирования
ksOpenTable (pObj);

TextParam par;
//в цикле будем брать все существующие ячейки
while ((ksGetTableColumnText (&numb, &par))!=0)
{
p=par.pTextLine;
TextLineParam linePar;
for (int i=0; i < GetArrayCount(p); i)
{
GetArrayItem(p , i, &linePar, sizeof(TextLineParam));
TextItemParam itemPar;
for (int j=0; j< GetArrayCount(linePar.pTextItem); j)
```

```
{
GetArrayItem (linePar.pTextItem, j, &itemPar,
sizeof (TextItemParam));
if (strlen(itemPar.s))
{
strcat(itemPar.s, "!!!");
SetArrayItem (linePar.pTextItem, j, &itemPar,
sizeof(TextItemParam));
}
}
SetArrayItem(p , i, &linePar, sizeof(TextLineParam));
}
//очистим массив текстовых строк
ksSetTableColumnText(numb, &par);
}

EndObj();//закрыли объект "таблица"
}
else
Error("Указанный объект - не таблица");
}
else
Error("нет объекта");
}
```

ksReadTableFromFile - Пример использования

```
char tabName[250];
 int j1;
 do {

   //выберем файл таблицы
  if((j1 = ksChoiceFile("*.tbl","таблицы(*.tbl)|*.tbl|Все файлы (*.*)|*.*|", tabName, 250, 0)) !=
0){
   double x, y;

   //подготовим структуры фантома и запросов для Placement
   struct Phantom rub;
```

```
 rub.type1.xBase = 0;
 rub.type1.yBase = 0;
 rub.type1.scale = 1;
 rub.phType   = 1;

//во временную группу положим вставку фрагмента, взятую из библиотеки фрагментов
 PlacementParam par;
 par.xBase = 0;
 par.yBase = 0;
 par.ang  = 0;
 par.scale = 1 ;
 //создаем временную группу
 rub.type1.gr = NewGroup (1);
 reference pTab = ksReadTableFromFile(tabName); //полное имя к файлу таблицы
 EndGroup();

 if (pTab && Cursor(NULL, &x, &y, &rub)) {
  //сдвигаем группу
  MoveObj(rub.type1.gr, x, y);
  //ставим группу в модель
  StoreTmpGroup(rub.type1.gr);
  ClearGroup(rub.type1.gr);
  DeleteObj(rub.type1.gr);
 }
 else {
  if (pTab)
   Error("Ошибка при считывании таблицы");
  DeleteObj(rub.type1.gr);
 }

}
} while(j1);
```

ksGetTextAlign, ksSetTextAlign - Пример использования

```
 reference iter = ::CreateIterator(TEXT_OBJ, 0); // итератор по всем текстам документа
 if (iter) {
 reference txt = ::MoveIterator(iter, 'F');  // первый текст
```

```
   while (::ExistObj(txt)) {          // пройдем по всем текстам
if (::ksGetTextAlign(txt) == txta_Left)   // если точка привязки текста слева
    ::ksSetTextAlign(txt, txta_Right);    // установим ее справа
   txt = ::MoveIterator(iter, 'N');      // следующий текст
   }
  }
  ::DeleteIterator(iter);             // удалим итератор
```

ksSetTextLineAlign - Пример использования

```
 if (::OpenStamp()) {
  ::ColumnNumber(2); // 2-справа
  ::ksSetTextLineAlign(0);
  ::TextLine(NEW_LINE, 0, 0 , "1111111");
  ::CloseStamp();
 }
```

Пример использования функций для редактирования таблицы

```
// редактирование таблицы
reference pObj;

RequestInfo info;
double x, y;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите таблицу";

// взять таблицу на чертеже
int j = Cursor(&info, &x ,&y, 0);
if (j)
{
if (ExistObj (pObj = FindObj (x, y, MAXDOUBLE)))
{
//узнаем тип объекта
int type =GetObjParam( pObj,0,0,0);//указатель на графический объект
//проверить, таблица ли полученный объект
if (type == TABLE_OBJ)
{
```

```
unsigned int numb;
reference p;
char buf[128];
//открыть таблицу для редактирования
ksOpenTable(pObj);

TextParam par;
//в цикле будем брать все существующие ячейки
while ((ksGetTableColumnText(&numb, &par))!=0)
{
sprintf(buf, "numb =%d  ", numb);
Message(buf);
p=par.pTextLine;
TextLineParam par2;
for (int i=0; i < GetArrayCount(p); i)
{
GetArrayItem (p, i, &par2, sizeof(TextLineParam));
sprintf(buf, "i =%d style=%d  ", i, par2.style);
Message(buf);
TextItemParam par3;
for (int j=0; j< GetArrayCount(par2.pTextItem); j)
{
GetArrayItem (par2.pTextItem, j, &par3,
sizeof(TextItemParam));
if (!par3.tip)
sprintf(buf, "компонента=%d
h=%5.1f\ns=%s\n fontName=%s\nбитвектор =%d ",j, par3.font.height, par3.s,
par3.font.fontName, par3.font.bitVector);
else
sprintf (buf, "компонента=%d тип = %d
номер спецзнака=%d ",j,par3.tip,par3.iSNumb);
Message(buf);
}
//очистим массив компонент
DeleteArray(par2.pTextItem);
}
//очистим массив текстовых строк
DeleteArray(p);
```

```
}
//берем ячейку 2
ColumnNumber(2);
Text(0, 0, 0, 5, 1 ,0,"вторая ячейка");

ksDivideTableItem(3, 1, 2);
ColumnNumber(4);
Text(0, 0, 0, 5, 1 ,0,"4");

EndObj();//закрыли объект "таблица"
}
else
Error("это не таблица");
}
else
Error("нет объекта");
}
```

Пример редактирования допуска формы

```
// редактирование допуска формы
reference pObj;

RequestInfo info;
double x, y;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите допуск формы";
//взять допуск формы на чертеже
int j = Cursor(&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj(x, y, MAXDOUBLE)))
{
//узнаем тип объекта
int type =GetObjParam( pObj,0,0,0);//указатель на графический объект
//проверим,является ли пришедший объект допуском фомы
if (type == TOLERANCE_OBJ)
{
```

```
unsigned int numb;
reference p;
char buf[128];
//открыть допуск формы для редактирования
ksOpenTolerance(pObj);

ToleranceParam par;
//возьмем параметры допуска формы
GetObjParam (pObj,//указатель на графический объект
&par,//указатель на структуру параметров
sizeof (ToleranceParam),//размер
//структуры параметров
ALLPARAM);//тип считывания параметров

sprintf (buf, "базовая точка=%d стиль=%d
расположение-%s\nx=%5.1f y=%5.1f",
par.tBase, par.style, par.type ?
"вертикальное":"горизонтальное", par.x, par.y);
Message(buf);

//в цикле будем брать все существующие ячейки
while ((p=ksGetToleranceColumnText(&numb))!=0)
{
sprintf(buf, "numb =%d  ", numb);
Message(buf);

TextLineParam par2;
for (int i=0; i < GetArrayCount(p); i)
{
GetArrayItem(p , i, &par2, sizeof(TextLineParam));
sprintf(buf, "i =%d style=%d  ", i, par2.style);
Message(buf);
TextItemParam par3;
for (int j=0; j< GetArrayCount(par2.pTextItem); j)
{
GetArrayItem (par2.pTextItem, j, &par3,
sizeof(TextItemParam));
if (!par3.tip)
```

```
sprintf (buf, "компонента=%d h=%5.1f\ns=%s\n
fontName=%s\nбитвектор =%d
",j,par3.font.height,par3.s
par3.font.fontName, par3.font.bitVector);
else
sprintf (buf, "компонента=%d тип = %d номер
спецзнака=%d ",j,par3.tip,par3.iSNumb);
Message(buf);
}
DeleteArray(par2.pTextItem);//очистим массив компонент
}
//очистим массив текстовых строк
DeleteArray(p);
}
//заменим параметры допуска формы
par.x = 10;
par.y = 10;
int rez = SetObjParam (pObj,//указатель на графический объект
&par,//указатель на структуру параметров
sizeof(ToleranceParam),//размер структуры параметров
ALLPARAM); //тип считывания параметров

// заменим текст в ячейке 2
ColumnNumber(2);
TextLine (NEW_LINE ,0,0 ,"вторая ячейка");
//разделим ячейку 3 вертикальным ребром с типом " тонкая"
ksDivideTableItem(3, 1, 2);
//перестроим виртуальную сетку
ksRebuildTableVirtualGrid();
// введем текст в ячейку 4
ColumnNumber(4);
TextLine (NEW_LINE ,0,0 ,"4");

EndObj();//закрыли объект "допуск формы"
}
else
Error("это не допуск формы");
}
```

```
else
Error("нет объекта");
}
```

Пример использования функций для редактирования допуска формы

```
// редактирование допуска формы
 reference pObj;

 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите допуск формы";
 int j = Cursor(&info, &x ,&y, 0);
 if (j) {
  if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
   //узнаем тип объекта
   int type =GetObjParam( pObj,0,0,0);  //указатель на графический
объект
   if (type == TOLERANCE_OBJ) {
    unsigned int numb;
    char buf[128];
    //открыть допуск формы для редактирования
    ksOpenTolerance(pObj);

    ToleranceParam par;
    //параметры допуска формы
    GetObjParam( pObj,  //указатель на графический объект
         &par, //указатель на структуру параметров
         sizeof(ToleranceParam),  //размер структуры
параметров
         ALLPARAM); //тип считывания параметров

    sprintf(buf, "базовая точка=%d стиль=%d расположение-%s\nx=%5.1f
y=%5.1f ",
           par.tBase, par.style, par.type ?
"вертикальное":"горизонтальное", par.x, par.y);
    Message(buf);
```

```
   TextLineParam par1;
   //в цикле будем брать все существующие ячейки
   while (ksGetToleranceColumnText(&numb, &par1)!=0) {
    sprintf(buf, "numb =%d  ", numb);
    Message(buf);

    sprintf(buf, "style=%d ", par1.style);
    Message(buf);
    TextItemParam par3;
    for (int j=0; j< GetArrayCount(par1.pTextItem); j) {
     GetArrayItem(par1.pTextItem, j, &par3, sizeof(TextItemParam));
     if (!par3.tip)
      sprintf(buf, "компонента=%d h=%5.1f\ns=%s\n
fontName=%s\nбитвектор =%d ",j,par3.font.height,par3.s,
          par3.font.fontName, par3.font.bitVector);
     else
      sprintf(buf, "компонента=%d тип = %d номер спецзнака=%d
",j,par3.tip,par3.iSNumb);
     Message(buf);
    }
    DeleteArray(par1.pTextItem); //очистим массив компонент
   }
  //заменим параметры
   par.x = 10;
   par.y = 10;
   SetObjParam( pObj,  //указатель на графический объект
        &par, //указатель на структуру параметров
        sizeof(ToleranceParam ),  //размер структуры
параметров
        ALLPARAM); //тип считывания параметров

   ColumnNumber(2);
   TextLine (NEW_LINE ,0,0 ,"вторая ячейка");
   ksDivideTableItem(3, 1, 2);
   ColumnNumber(4);
   TextLine (NEW_LINE ,0,0 ,"4");
```

```
//      ksSetTableBorderStyle(1,2,1);
//      ksClearTableColumnText(0);
    EndObj();//закрыли объект "допуск формы"
   }
   else
    Error("это не допуск формы");
  }
  else
   Error("нет объекта");
 }
```

ksGetPointOnToleranceTable - пример использования

```
reference pObj;

RequestInfo info;
double x, y;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите допуск формы";
int j = Cursor(&info, &x ,&y, 0);
if (j) {
 if(ExistObj(pObj = FindObj(x, y, 1e6))){
  //узнаем тип объекта
  int type =GetObjParam( pObj,0,0,0);  //указатель на графический объект
  if (type == TOLERANCE_OBJ) {
   for(unsigned char i = 1; i < 9 ; i) {
    MathPointParam parPoint;
    if (ksGetPointOnToleranceTable(pObj,     // указатель на допуск формы
                  i,   // тип точки на таблице допуска формы
                  &parPoint)) // координаты точки
      Point(parPoint.x, parPoint.y,  //координаты точки
        (unsigned short)(i-1)); //стиль отрисовки точки
   }
  }
  else
   Error("это не допуск формы");
 }
 else
```

```
  Error("нет объекта");
}
```

ksSetToleranceColumnText - пример использования

```
// редактирование допуска формы
reference pObj;

RequestInfo  info;
double x, y;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите допуск формы";
int j = Cursor(&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj (x,y, 1000000.)))
{
//узнаем тип объекта
int type =GetObjParam (pObj,0,0,0);//указатель на графический объект
if (type == TOLERANCE_OBJ)
{
unsigned int numb;
//открыть допуск формы для редактирования
ksOpenTolerance(pObj);
TextLineParam linePar;
//в цикле будем брать все существующие ячейки
while (ksGetToleranceColumnText(&numb, &linePar)!=0)
{
TextItemParam itemPar;
for (int j=0; j< GetArrayCount(linePar.pTextItem); j++)
{
GetArrayItem (linePar.pTextItem, j, &itemPar,
sizeof (TextItemParam));
if (strlen (itemPar.s))
{
strcat (itemPar.s, "!!!");
SetArrayItem (linePar.pTextItem,
 j, &itemPar,
sizeof(TextItemParam));
```

```
}
}
ksSetToleranceColumnText(numb, &linePar);
}

EndObj();//закрыли объект "допуск формы"
}
else
Error ("Указанный объект - не допуск формы");
}
else
Error("нет объекта");
}
```

Пример задания штампа

```
void Stamp_Example (void)
{
if (OpenStamp())
{
ColumnNumber(2);
TextLine (NEW_LINE, 0, 0, 1К11.00.00.021);

ColumnNumber(1);
TextLine (NEW_LINE, 0, 0, Крестовина);

CloseStamp();
}
ClearStamp(2);  //очистить вторую графу
};
```

SetStampColumnText, GetStampColumnText пример использования

```
if (OpenStamp())
{
unsigned int numb;
reference p;
```

```
//в цикле будем брать все существующие графы
while ((p=GetStampColumnText (&numb))!=0)
{
// p - текст текущей графы
char buf[128];
sprintf (buf, "numb =%d   ", numb);
Message(buf);

TextLineParam parLine;
for (int i=0, count = GetArrayCount(p); i < count; i++)
{
//берем текущую строку графы
GetArrayItem (p , i, &parLine, sizeof(TextLineParam));
sprintf (buf, "i =%d style=%d   ", i, parLine.style);
Message(buf);
TextItemParam parItem;
for (int j=0, count1 = GetArrayCount(parLine.pTextItem); j < count1; j++)
{
//берем текущую компоненту строки
GetArrayItem (parLine.pTextItem, j,
&parItem, sizeof (TextItemParam));
sprintf (buf, "компонента=%d h=%5.1f\ns=%s\n fontName=%s ",j,
parItem.font.height,parItem.s, parItem.font.fontName);
strcat (parItem.s, "!!!");
//меняем текущую компоненту строки
SetArrayItem (parLine.pTextItem, j, &parItem, sizeof (TextItemParam));
Message(buf);
}
DeleteArray (parLine.pTextItem); //очистим массив компонент
}
//меняем текущую строку в массиве
SetArrayItem (p , i, &parLine, sizeof(TextLineParam));
//меняем текущую графу
SetStampColumnText (numb, p);
//очистим массив текстовых строк
DeleteArray (p);
}
CloseStamp();
```

```
}
else
Error ("Штамп не найден");
```

GetReferenceDocumentPart пример использования

```
if (ksGetCurrentDocument (1))// документ должен быть графическим
{
//получим указатель на технические требования
reference pTT = GetReferenceDocumentPart (1);
if (pTT)
{
TechnicalDemandParam par;
//получим параметры описания ТТ
GetObjParam (pTT, &par, sizeof (par), TECHNICAL_DEMAND_PAR);
char buf[128];
sprintf (buf, "число строк ТТ =%d",par.strCount);
Message (buf);
//открываем ТТ на редактирование
//(ТТ должны быть открыты, чтобы менять тексты)
OpenTechnicalDemand (par.pGab,//динамический массив
// листов технических требований или 0
par.style);//стиль текста для технических требований
// (если 0 - умолчательное значение)
//пройдемся по ТТ и получим текст
TextLineParam parLine;
for (int i = 0; i < par.strCount; i++)
{
//берем текущую строку
GetObjParam (pTT, &parLine,
sizeof (TextLineParam),
TT_FIRST_STR+i);
TextItemParam parItem;
for (int j=0, count1 = GetArrayCount(parLine.pTextItem); j < count1; j++)
{
//берем текущий компонент строки
GetArrayItem (parLine.pTextItem,
j, &parItem,
```

```
sizeof (TextItemParam));
strcat (parItem.s, "!!!");
//меняем текущий компонент строки
SetArrayItem (parLine.pTextItem,
j, &parItem,
sizeof (TextItemParam));
Message (parItem.s);
}
//меняем текущую строку
SetObjParam (pTT, &parLine,
sizeof (TextLineParam),
TT_FIRST_STR+i);
}
//закрываем ТТ
CloseTechnicalDemand ();
return;
}
}
Error ("Документ должен быть листом");
```

ksGetZona пример использования

```
RequestInfo  info;
вания
K ksGetZona
//обнулить структуру  info;
memset (&info, 0, sizeof (info));
double x, y;
info.prompt = "Укажите точку";
char zona [128];
while (Cursor (&info, &x ,&y, 0) == -1) {
int rez =  ksGetZona(x, y, zona, 128);      //размер присланного буфера
if (rez == -1 || !rez) {
Error(rez ? "В текущем документе нет разбиения на зоны" : "Ошибка");
break;
}
else
   Message (zona);
```

```
}
```

Пример использования функций работы со слоями

```
void Layers_Example (void) {

reference lay1, lay2;
int number;
char buf[128];
LayerParam par;

lay1 = Layer (1); /* создание слоя 1 */
strcpy(par.name, Квадрат);
par.state = 0; par.color = RGB(255, 0, 0); /* красный */
SetObjParam (lay1,&par,sizeof(par), ALLPARAM);

LineSeg (0,   0,   0,  100, 1); /* объекты записываются  */
LineSeg (0,  100, 100,  100, 1); /* в слой 1 - Квадрат    */
LineSeg (100, 100, 100,   0, 1);
LineSeg (100,  0,   0,   0, 1);

lay2 = Layer (2); /* создание слоя 2 */
strcpy(par.name, Треугольник);
par.state = 0; par.color = RGB(0, 255, 0); /* зеленый */
SetObjParam (lay2,&par,sizeof(par), ALLPARAM);

LineSeg (10,   10,  50,  90, 2); /* объекты записываются  */
LineSeg (50,  90,  90,  10, 2); /* в слой Треугольник   */
LineSeg (90,  10,  10,  10, 2);

number = GetLayerNumber (lay1); /* получить номер 1*/
sprintf(buf,number=%d,number);
Message(buf);
lay2 = GetLayerReference(2); /* получить указатель на слой с номером 2 */

}; /* Layers_Example */
```

ChangeObjectLayer - пример использования

```
{
reference obj;

Layer (1);
obj = LineSeg (10, 10 , 50, 50, 1);

Layer (2);
ChangeObjectLayer(obj, 2);
}
```

CreateSheetView, OpenView пример использования

```
void OpenView_Example (void)
{
reference view;
int n;
ViewParam *par

par->x=100;   par->y=200;   par->scale=1;   par->ang=0; par.state=0;
CreateSheetView(par,1);  /* создать вид 1 */

LineSeg (0,      0,     0,   100, 1);  /* объекты записываются   */
LineSeg (0,    100, 100,   100, 1);  /* в вид 1                */
LineSeg (100, 100, 100,      0, 1);
LineSeg (100,    0,     0,      0, 1);

par->x=300;   par->y=200;   par->scale=1;   par->ang=0; par.state=0;
CreateSheetView(par,2); /* создать вид 2 */
Circle (50, 50, 20, 2);  /* объект записывается в вид 2 */

OpenView(1); /* открыть вид 1 */
};
```

NewViewNumber пример использования

```
reference v;
char buf[128];
```

```
int number = NewViewNumber();
ViewParam par;
memset (&par, 0, sizeof (ViewParam));
par.x = 10;
par.y = 20;
par.scale = 0.5;
par.ang = 45;
par.color = RGB(10,20,10);
par.state = stACTIVE;
strcpy (par.name, "user view");

//создали вид
v=CreateSheetView(&par, &number);
int numb = GetViewNumber( v);
sprintf (buf,"создали вид numb=%d, number=%d", numb, number);
Message(buf);
```

GetViewReference пример использования

```
//получим указатель на первый вид
reference v = GetViewReference (1);
if (v)
{
ViewParam par;
//возьмем параметры вида
if (GetObjParam (v, &par, sizeof (ViewParam), ALLPARAM))
{
char buf[255];
sprintf (buf,
"Параметры вида\nx = %0.1f, y = %0.1f\nscale=
%0.1f ang =%0.1f state=%d\nname=%s",
par.x, par.y, par.scale, par.ang, par.state, par.name);
Message(buf);
}
}
```

Пример задания технических требований

```
void TechnicalDemand_Example (void) {
```

```
reference p;

reference pGab = CreateArray(RECT_ARR,0) ;
  RectParam par;

  par.pTop.x= 230;  par.pTop.y = 80;
  par.pBot.x= 415;  par.pBot.y = 65;
  AddArrayItem(pGab,-1, &par, sizeof(RectParam));

  par.pTop.x= 45;  par.pTop.y = 30;
  par.pBot.x= 230;  par.pBot.y = 15;
  AddArrayItem(pGab,-1, &par, sizeof(RectParam));

  // размещение на 2 страницах
reference tDem;
  if(OpenTechnicalDemand( pGab)) {
TextLine (NEW_LINE  ,0,0 ,1111111);
TextLine (NEW_LINE  ,0,0 ,222222);
TextLine (NEW_LINE  ,0,0 ,3333333);
TextLine (NEW_LINE  ,0,0 ,4444444);
TextLine (NEW_LINE  ,0,0 ,5555555);
TextLine (NEW_LINE  ,0,0 ,6666666);
tDem =CloseTechnicalDemand();
  }
  DeleteArray(pGab);


// размещение на 1 странице
  if(OpenTechnicalDemand( 0)) {
TextLine (NEW_LINE  ,0,0 ,1111111);
TextLine (NEW_LINE  ,0,0 ,222222);
TextLine (NEW_LINE  ,0,0 ,3333333);
TextLine (NEW_LINE  ,0,0 ,4444444);
TextLine (NEW_LINE  ,0,0 ,5555555);
TextLine (NEW_LINE  ,0,0 ,6666666);
CloseTechnicalDemand();
  }
```

```
}; /* TechnicalDemand_Example */
```

SpecRough пример использования

```
void SpecRough_Example (void) {

SpecRough(0, 1, “Rz40”);

}; /* SpecRough_Example */
```

ksSpecRough пример использования

```
void ksSpecRough_Example (void)
{
SpecRoughParam *par;

par.style = 1;//стиль
par.sign = 0;
par.t = 1;
par.s [4] = «Rz40»;

SpecRoughObj = ksSpecRough(&par);
};
```

LinDimension - Пример использования

```
void LinDimension_Example (void)
{
reference p;
    LDimParam linPar;
    memset(&linPar, 0,sizeof(LDimParam));

    //параметры текста
    linPar.tPar.bitFlag = _AUTONOMINAL|_PREFIX|_TOLERANCE|_DEVIATION|_UNIT;
    linPar.tPar.sign = 1;  //диаметр
    linPar.tPar.pText = CreateArray(CHAR_STR_ARR ,0);
```

```
    AddArrayItem (linPar.tPar.pText, -1,"2отв.", 6); // _PREFIX
    AddArrayItem(linPar.tPar.pText, -1,"H12", 5); // _TOLERANCE
    AddArrayItem(linPar.tPar.pText, -1," мм", 5); // _UNIT

    //параметры привязки
    linPar.sPar.ps = 0; // 0-горизонтальный
    linPar.sPar.x1 = 50; linPar.sPar.y1 = 50; // 1-ая точка
    linPar.sPar.x2 = 70; linPar.sPar.y2 = 60;  // 2-ая точка
    linPar.sPar.dy = -20;  // вектор, определяющий
    linPar.sPar.dx = 0; // положение размерной линии
    linPar.sPar.basePoint = 1; // dx, dy откладывать от первой точки

    // параметры отрисовки линейного размера
    linPar.dPar.textPos   = 0; // автоматическая простановка
    linPar.dPar.textBase = 0; // от средины размера
    linPar.dPar.pl1       = 0; // 1-ая  выносная линия есть
    linPar.dPar.pl2       = 0; // 2-ая  выносная линия есть
    linPar.dPar.pt1       = 1; // тип стрелки у 1-ой выносной линии  1-изнутри
    linPar.dPar.pt2       =  1; // тип стрелки у 2-ой выносной линии  1-изнутри

    linPar.dPar.shelfDir   = -1; // полка направлена влево
    linPar.dPar.ang       = -30; // угол наклона ножки
    linPar.dPar.length     = 20; // длина ножки

    p = LinDimension(&linPar); //параметры линейного размера
};
```

ksOrdinatedDimension - Пример использования

```
 OrdinatedDimParam par; // параметры размера высоты
 memset(&par, 0, sizeof(par));

 // автоматическое простановка номинала, с подчеркиванием
 par.tPar.bitFlag = _AUTONOMINAL|_UNDER_LINE;

 par.tPar.pText = CreateArray(CHAR_STR_ARR ,0);
```

```
 par.dPar.type = OD_FRONTVIEW; // тип - для вида спереди, с полкой и стрелкой, возможна выносная линия
 par.sPar.x0 = 0;   // координаты точки, задающей нулевой уровень
 par.sPar.y0 = 0;

 par.sPar.x1 = 0;   // координаты точки, задающей измеряемый уровень
 par.sPar.y1 = 100;

 par.sPar.x2 = 100; // координата точки, задающей положение размерной надписи
 par.sPar.y2 = 1;   // полка выше линии выноски

 reference p = OrdinatedDimension(&par); // создание размера высоты

 memset(&par, 0, sizeof(par));
 GetObjParam(p, &par, sizeof(par), ALLPARAM); // получить параметры размера
 par.sPar.y2 = -1;                            // полка ниже лини выноски
 SetObjParam(p, &par, sizeof(par), ALLPARAM); // сохранить параметры размера
```

AngDimension - Пример использования

```
void AngDimension_Example (void)
{
reference p;
ADimParam angPar;
memset (angPar, 0, sizeof (ADimParam));

LineSeg (40, 0, 40, 40, 1);
LineSeg (40, 0, 60, 20, 1);

//параметры текста
angPar.tPar.bitFlag = _AUTONOMINAL|_DEVIATION|;
angPar.tPar.sign = 0;
angPar.tPar.pText = CreateArray (CHAR_STR_ARR ,0);

AddArrayItem (angPar.tPar.pText, -1, "+0.3", 5);
AddArrayItem (angPar.tPar.pText, -1, "-0.3", 5);

//привязка углового размера
angPar.sPar.xc = 40; angPar.sPar.yc = 0;// центр
```

```
angPar.sPar.x1 = 40; angPar.sPar.y1 = 40;// 1 - точка выхода выносной линии
angPar.sPar.x2 = 60; angPar.sPar.y2 = 20;// 2 - точка выхода выносной линии
angPar.sPar.rad = 50;//радиус размерной дуги
angPar.sPar.dir = -1;// размерная линия по часовой стрелке

//отображение размера
angPar.dPar.textPos   = 0;//автоматическая простановка
angPar.dPar.textBase  = 0;//над серединой размерной линии
angPar.dPar.pl1 = 0;// 1-ая выносной линия есть
angPar.dPar.pl2 = 0;// 2-ая выносной линия есть
angPar.dPar.pt1 = 1;// стрелка изнутри у 1-ой выносной линии
angPar.dPar.pt2 = 1;// стрелка изнутри у 2-ой выносной линии

//параметры выносной полки
angPar.dPar.shelfDir     = 0;//полки нет

p = AngDimension(&angPar);//параметры углового размера
};
```

ksGetQualityNames - Пример использования

```
 reference names = ::CreateArray(CHAR_STR_ARR, 0); // динамический массив строк
 // получить поля допусков для размера 139 мм, верхнее отклонение 0.16 мм,
 // нижнее - 0 мм, система отверстия, с учётом ограничений
 if (::ksGetQualityNames(names, 139, 0.16, 0, 1/*system*/, 1/*withLimitation*/)) {
   string s("");                                  // строка сообщения
   int count = ::GetArrayCount(names);        // кол-во элементов в массиве
   for (int i = 0; i < count; i++) {                  // пройдём по всему массиву
     char item[255];                                  // буфер
     if (::GetArrayItem(names, i, item, 255)) {    // взять текущую строку из массива
       s += item;                                 // добавить её к сообщению
       s += "  ";                                 // разделитель
     }
   }                                          // выводим сообщение
   ::Message((char*)s.c_str());                   // удалить массив
   ::DeleteArray(names);
 }
```

ksGetQualityDefects - Пример использования

```
 double high = 0, low = 0;                    // отклонения

 char name[20];                               // поле допуска
 ::strcpy(name, "H7");                        // умолчательное значение
 if (ReadString("Укажите поле допуска", name, 4)) { // запрос поля допуска у пользователя
   double dimValue = 15;                      // значение размера в мм
  if (ReadDouble("Введите размер в мм:", 15, 0, 500, &dimValue)) { // запрос значения размера у пользователя
    ::ksGetQualityDefects(name, dimValue, &high, &low, true/*inMM*/); // получить отклонения в мм
     char buf[255];                           // строка сообщения
     ::sprintf(buf, "Поле допуска %s\nЗначение = %g, high = %g, low = %g",
             name, dimValue, high, low);      // формируем сообщение
     ::Message(buf);                          // выводим сообщение
```

ksGetQualityContensParam - Пример использования

```
 char name[20];                               // поле допуска
 ::strcpy(name, "H7");                        // умолчательное значение
 if (ReadString("Укажите поле допуска", name, 4)) { // запрос поля допуска у пользователя
   char buf[255];                             // буфер
   string s("");                              // строка сообщения
   if (!::strcmp(name, "")) {                 // пустая строка - идём итератором
    // итератор по системе отверстия с учётом ограничений наложенных в системе
    reference iter = ::ksCreateQualityIterator(1/*system*/, 1/*withLimitation*/);
    if (iter) {                               // итератор создан
     QualityContensParam param;               // структура параметров квалитета
     int first = 0;                           // начало чтения
     // продолжаем чтение пока не считаем все квалитеты
     while (::ksMoveQualityIterator(iter, &param, false/*inMM*/, !first ? 'F' : 'N')) {
      ::sprintf(buf, "Система %s, Тип квалитета %i, Поле допуска %s\n",
              (param.systemQuality == 1 ? "Вала" : "Отверстия"), param.kindQuality,
              param.name);                    // формируем строку для текущего квалитета
      s += buf;                               // добавляем к сообщению

      int count = ::GetArrayCount(param.pQualityItems); // кол-во интервалов
```

```
        for (int i = 0; i < count; i++) {         // проходим по всем интервалам
          QualityItemParam item;                  // запись об одном интервале для какого-то
квалитета
         // считываем текущюю запись
         if (::GetArrayItem(param.pQualityItems, i, &item, sizeof(QualityItemParam))) {
          // выводим параметры интервала в строку
          ::sprintf(buf, "Значение: %i < X < %i, high = %g, low = %g\n",
                  item.minLimit, item.maxLimit, item.high, item.low);
          s += buf;                               // добавляем к сообщению
         }
        }
        ::Message((char*)s.c_str());              // выводим сообщение
        s = "";                                   // очищаем строку сообщения
        ::DeleteArray(param.pQualityItems);       // удаляем массив интервалов
        param.pQualityItems = 0;

        if (!first)                               // пришли первый раз
          first = 1;                              // взводим флаг
       }
       ::DeleteIterator(iter);                    // удалить итератор
     }
    }
    else {                                        // имя есть
     QualityContensParam param;                   // структура параметров квалитета
     // считываем параметры указанного квалитета
     if (::ksGetQualityContensParam(name, &param, false/*inMM*/)) {
      ::sprintf(buf, "Система %s, Тип квалитета %i, Поле допуска %s\n",
              (param.systemQuality == 1 ? "Вала" : "Отверстия"), param.kindQuality,
              param.name);                        // формируем строку для текущего квалитета
      s += buf;                                   // добавляем к сообщению

      int count = ::GetArrayCount(param.pQualityItems); // кол-во интервалов
      for (int i = 0; i < count; i++) {           // проходим по всем интервалам
        QualityItemParam item;                    // запись об одном интервале для какого-то
квалитета
       // считываем текущюю запись
       if (::GetArrayItem(param.pQualityItems, i, &item, sizeof(QualityItemParam))) {
        // выводим параметры интервала в строку
```

```
        ::sprintf(buf, "Значение: %i < X < %i, high = %g, low = %g\n",
                item.minLimit, item.maxLimit, item.high, item.low);
        s += buf;                          // добавляем к сообщению
       }
      }
      ::Message((char*)s.c_str());          // выводим сообщение
      s = "";                               // очищаем строку сообщения
      ::DeleteArray(param.pQualityItems);   // удаляем массив интервалов
      param.pQualityItems = 0;
     }
    }
  }
```

DiamDimension - Пример использования

```
void DiamDimension_Example (void)
K DiamDimension
{
reference p;
RDimParam linPar;
memset (&linPar, 0, sizeof (RDimParam));

Circle(50,50,70,1);

//параметры текста
linPar.tPar.bitFlag = _AUTONOMINAL;// автоматическая простановка текста
linPar.tPar.pText = 0;
linPar.tPar.sign = 0;// знак диаметра ставится автоматически

//параметры привязки диаметрального  размера
linPar.sPar.xc = 50; linPar.sPar.yc = 50;// координаты центра
linPar.sPar.rad = 70;

//параметры отрисовки диаметрального  размера
linPar.dPar.textPos   = 75;// положение текста
inPar.dPar.pt1 = 1;// тип стрелки -изнутри
```

```
linPar.dPar.pt2 = 1;// тип стрелки -изнутри

//параметры выносной полки
linPar.dPar.shelfDir   = 1;// полка направлена вправо
linPar.dPar.ang        = -30; // угол наклона размерной линии

p = DiamDimension(&linPar);// диаметральный размер
};
```

RadDimension, RadBreakDimension - Пример использования

```
void RadDimension_Example (void)
{
reference p;
RDimParam linPar;
memset (&linPar, 0, sizeof(RDimParam));

Circle(50,50,70,1);

//параметры текста
linPar.tPar.bitFlag = _AUTONOMINAL;
linPar.tPar.pText = 0
linPar.tPar.sign = 0;// значок радиуса ставится автоматически

//параметры привязки диаметрального  размера
linPar.sPar.xc = 50; linPar.sPar.yc = 50;//центр
linPar.sPar.rad = 70;

//параметры отрисовки диаметрального размера
linPar.dPar.textPos   = 75;// положение текста
linPar.dPar.pt1 = 2;// тип стрелки снаружи
linPar.dPar.pt2 = 0;// размерная линия  рисуется
linPar.dPar.shelfDir = 1;//полка направлена вправо
linPar.dPar.ang = 30;//угол наклона размерной линии

p = RadDimension (&linPar);// радиальный размер

reference p1;
```

```
RBreakDimParam linPar;
memset (&linPar, 0, sizeof (RDimParam));

Circle(150,50,70,1);

//параметры текста
linPar.tPar.bitFlag = 0;
linPar.tPar.pText = CreateArray(CHAR_STR_ARR ,0);
AddArrayItem (linPar.tPar.pText, -1, "100", 8);
linPar.tPar.sign = 0;

//параметры привязки
linPar.sPar.xc = 50; linPar.sPar.yc = 50;//центр
linPar.sPar.rad = 70;

//параметры отрисовки
linPar.dPar.pt   = 1;// тип стрелки изнутри
linPar.dPar.ang = 30;// угол
linPar.dPar.pb  = 20;// длина излома

p = RadBreakDimension(&linPar);// радиальный размер с изломом
};
```

Rough - Пример использования

```
void Rough_Example (void)
{
reference p;
RoughParam roughPar;
memset (&roughPar, 0, sizeof (RoughParam));

//заполним параметры шероховатости
roughPar.rPar.type=0;// без обработки
roughPar.rPar.around=0;
roughPar.rPar.x = 50;  roughPar.rPar.y = 50;// опорная точка
roughPar.rPar.ang = 90;// угол наклона оси знака
roughPar.rPar.cText0=1;// строк над знаком
```

```
roughPar.rPar.cText1=0;// строк над полкой
roughPar.rPar.cText2=0;// строк под полкой
roughPar.rPar.cText3=0;// строк под полкой

// массив неопределенной длины строк текста
roughPar.rPar.pText =CreateArray (CHAR_STR_ARR ,0);

AddArrayItem (roughPar.rPar.pText, -1, 6,3, 4);

//параметры выносной полки
roughPar.shPar.psh   = 0;// полки нет
roughPar.shPar.ang = 130;// угол наклона ножки
roughPar.shPar.length= 20;// длина ножки

p = Rough (&roughPar);// шероховатость
};
```

Base - Пример использования

```
void Base_Example (void)
{
reference p;
BaseParam basePar;
memset (&basePar, 0, sizeof (BaseParam));

basePar.x1 = 50; basePar.y1 =50;// базовая точка
basePar.x2 = 50; basePar.y2 =30;// конечная точка ножки
strcpy (basePar.str, A);// надпись

p = Base (&basePar);
};
```

Tolerance - пример использования

```
void Tolerance_Example (void) {

ToleranceParam par;
```

```
memset(&par, 0, sizeof(ToleranceParam));

MathPointParam parPoint;
reference p;

  par.branch1.pMathPoint = CreateArray(POINT_ARR,  0);
  parPoint.x = 40; parPoint.y = 10;
  AddArrayItem(par.branch1.pMathPoint, -1, &parPoint , sizeof(parPoint));
  par.branch1.arrowType = 2;
  par.branch1.tCorner = 1;

  par.branch2.pMathPoint = CreateArray(POINT_ARR,  0);
  parPoint.x = 100; parPoint.y = 50;
  AddArrayItem(par.branch2.pMathPoint, -1, &parPoint , sizeof(parPoint));
  parPoint.x = 100; parPoint.y = 10;
  AddArrayItem(par.branch2.pMathPoint, -1, &parPoint , sizeof(parPoint));
  par.branch2.arrowType = 1;
  par.branch2.tCorner = 5;

  par.x = 40 ; par.y = 40;  par.type = 0;

  if (Tolerance(&par)) {
ColumnNumber(1);
int tip = 26;   //значек допуск соосности
TextLine (SPECIAL_SYMBOL  ,SPECIAL, &tip ,);

ColumnNumber(2);
tip = 2;        //значек диаметр
TextLine (SPECIAL_SYMBOL  ,SPECIAL, &tip ,);
 TextLine (NEW_LINE  ,0,0 ,0,004);
tip=30;        //зависимый допуск
TextLine (SPECIAL_SYMBOL  ,SPECIAL, &tip ,);

ColumnNumber(3);
 TextLine (NEW_LINE  ,0,0 ,A);
    p = EndObj();
  }
  DeleteArray(par.branch2.pMathPoint);
```

```
  DeleteArray(par.branch1.pMathPoint);

}; /* Tolerance_Example */
```

CutLine - пример использования

```
void CutLine_Example (void) {

  reference p;
  CutLineParam  cutPar;
memset(&cutPar, 0, sizeof(CutLineParam));

  cutPar.right =1;  // стрелка справа
  strcpy (cutPar.str, A); // надпись
  cutPar.x1 = 30;  //  координаты надписи у первого участка
  cutPar.y1 = 65;
  cutPar.x2 = 95; // координаты надписи у второго участка
  cutPar.y2 = 15;
  cutPar.pMathPoint = CreateArray(POINT_ARR, 0);

  MathPointParam par;

  par.x=50; par.y=50; // первая точка
  AddArrayItem(cutPar.pMathPoint, -1, &par, sizeof(par));

   par.x=50; par.y=30; // вторая  точка
  AddArrayItem(cutPar.pMathPoint, -1, &par, sizeof(par));

  p = CutLine(&cutPar);

}; /* CutLine_Example */
```

ViewPointer - пример использования

```
void ViewPointer_Example (void)
{
ViewPointerParam  par;
memset (&par, 0, sizeof (ViewPointerParam));
```

```
par.x1 = 55; par.y1 = 50;// координаты вершины (острия) стрелки
par.x2 = 40; par.y2 = 50;// координаты конечной точки стрелки
par.xt = 40; par.yt = 52;// координаты текста
strcpy(par.str, A);// надпись
reference  p = ViewPointer (&par);//параметры стрелки вида
if (ExistObj(p))
LightObj(p, 1);
};
```

Leader - пример использования

```
void Leader_Example (void) {

  reference p;
  TextLineParam tLinePar;
memset(&tLinePar, 0, sizeof(TextLineParam));

  tLinePar.style=0;  //номер стиля строки текста
// массив неопределенной длины компонент строки текста
  tLinePar.pTextItem = CreateArray(TEXT_ITEM_ARR,0);

  TextItemFont  tFont;  // параметры шрифта компоненты строки текста
  tFont.fontName[0]=\0; // имя шрифта
  tFont.height=0;             // высота текста
  tFont.ksu=0;               // сужение текста
  tFont.color = RGB(0,0,0); // цвет
  tFont.bitVector=0;   // флаг параметров

  TextItemParam  ItemPar;    // параметры компоненты строки текста
  ItemPar.tip = 0;
  ItemPar.font = tFont; // параметры шрифта для компоненты текста
  ItemPar.iSNumb = 0; // номер символа

  LeaderParam leaderPar;
  memset(&leaderPar, 0, sizeof(LeaderParam));
  leaderPar.x=50; leaderPar.y=50; // начало полки
  leaderPar.arrowType = 1; // тип стрелки
```

```
  leaderPar.dirX=1;      // полка вправо
  leaderPar.signType=0;  // тип знака
  leaderPar.around=0;   // знак обработки по контуру выключен
  leaderPar.cText0=1;   // число строк над полкой
  leaderPar.cText1=1;   // число строк под полкой
  leaderPar.cText2=1;   // число строк над ножкой
  leaderPar.cText3=1;   // число строк под ножкой

  leaderPar.pTextline = CreateArray(TEXT_LINE_ARR,0);

  strcpy(ItemPar.s,строка над полкой);
// массив символов  для компоненты текста
  AddArrayItem(tLinePar.pTextItem, -1, &ItemPar, sizeof(ItemPar));
  AddArrayItem(leaderPar.pTextline, -1, &tLinePar, sizeof(tLinePar));

  ClearArray(tLinePar.pTextItem);
  strcpy(ItemPar.s,строка под полкой);
// массив символов  для компоненты текста
  AddArrayItem(tLinePar.pTextItem, -1, &ItemPar, sizeof(ItemPar));
  AddArrayItem(leaderPar.pTextline, -1, &tLinePar, sizeof(tLinePar));

  ClearArray(tLinePar.pTextItem);
  strcpy(ItemPar.s,2);   // массив символов  для компоненты текста
  AddArrayItem(tLinePar.pTextItem, -1, &ItemPar, sizeof(ItemPar));
  AddArrayItem(leaderPar.pTextline, -1, &tLinePar, sizeof(tLinePar));

  ClearArray(tLinePar.pTextItem);
  strcpy(ItemPar.s,3);
// массив символов  для компоненты текста
  AddArrayItem(tLinePar.pTextItem, -1, &ItemPar, sizeof(ItemPar));
  AddArrayItem(leaderPar.pTextline, -1, &tLinePar, sizeof(tLinePar));

  leaderPar.pPolyline  =CreateArray(POLYLINE_ARR,0);
  reference pPoly = CreateArray(POINT_ARR , 0);
  //две ножки по одной точке в каждой
  MathPointParam mPar;
  mPar.x = 10;  mPar.y = 10;
  AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
```

```
 AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

 mPar.x = 30;  mPar.y = 10;
 ClearArray(pPoly);
 AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
 AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

 p = Leader(&leaderPar);

}; /* Leader_Example */
```

PositionLeader - пример использования

```
void PositionLeader_Example (void) {

 reference p;
 PosLeaderParam  leaderPar;
 memset(&leaderPar, 0, sizeof(PosLeaderParam));

 leaderPar.x=50; leaderPar.y=50; // начало полки
 leaderPar.arrowType = 1; // тип стрелки
 leaderPar.dirX=-1;         // полка влево)

 leaderPar.pText = CreateArray(CHAR_STR_ARR,0);
 AddArrayItem(leaderPar.pText, -1, 11, 3);

 leaderPar.pPolyline  =CreateArray(POLYLINE_ARR,0);
 reference pPoly = CreateArray(POINT_ARR , 0);

 MathPointParam mPar;
 mPar.x = 10;  mPar.y = 10;
 AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
 AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

 mPar.x = 30;  mPar.y = 10;
 ClearArray(pPoly);
 AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
 AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));
```

```
 p = PositionLeader(&leaderPar);

}; /* PositionLeader_Example */
```

BrandLeader - пример использования

```
void BrandLeader_Example (void) {

  reference p;
  BrandLeaderParam  leaderPar;
  memset(&leaderPar, 0, sizeof(BrandLeaderParam));

  leaderPar.cText0 = 1;  // число строк в знаке
  leaderPar.cText1 = 1;  // число строк над ножкой
  leaderPar.cText2 = 1;  // число строк под ножкой

  leaderPar.x=50; leaderPar.y=50;  // начало полки
  leaderPar.arrowType = 1; // тип стрелки
  leaderPar.dirX=-1;      // полка влево

  leaderPar.pText = CreateArray(CHAR_STR_ARR,0);

  AddArrayItem(leaderPar.pText, -1, п.11, 5);
  AddArrayItem(leaderPar.pText, -1, Ну, 3);
  AddArrayItem(leaderPar.pText, -1, Ту, 3);

  leaderPar.pPolyline  =CreateArray(POLYLINE_ARR,0);
  reference pPoly = CreateArray(POINT_ARR , 0);

  MathPointParam mPar;
  mPar.x = 10;  mPar.y = 10;
  AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
  AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

  mPar.x = 30;  mPar.y = 10;
  ClearArray(pPoly);
  AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
```

```
  AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

  p = BrandLeader(&leaderPar);

}; /* BrandLeader_Example */
```

MarkerLeader - пример использования

```
void MarkerLeader_Example (void) {

  reference p;
  MarkerLeaderParam  leaderPar;
memset(&leaderPar, 0, sizeof(BrandLeaderParam));

  leaderPar.cText0 = 1;  // число строк в знаке
  leaderPar.cText1 = 1;  // число строк над ножкой
  leaderPar.cText2 = 1;  // число строк под ножкой

  leaderPar.x=50; leaderPar.y=50;  // начало полки
  leaderPar.arrowType = 2; // тип стрелки
  leaderPar.pText = CreateArray(CHAR_STR_ARR,0);

  AddArrayItem(leaderPar.pText, -1, п.11, 5);
  AddArrayItem(leaderPar.pText, -1, Ну, 3);
  AddArrayItem(leaderPar.pText, -1, Ту, 3);

  leaderPar.pPolyline  =CreateArray(POLYLINE_ARR,0);
  reference pPoly = CreateArray(POINT_ARR , 0);

  MathPointParam mPar;
  mPar.x = 10;  mPar.y = 10;
  AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
  AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));

  mPar.x = 30;  mPar.y = 10;
  ClearArray(pPoly);
  AddArrayItem(pPoly , -1, &mPar, sizeof(mPar));
  AddArrayItem(leaderPar.pPolyline , -1, &pPoly, sizeof(pPoly));
```

```
  p = MarkerLeader(&leaderPar);

}; /* MarkerLeader_Example */
```

ksCentreMarker - пример использования

```
CentreParam cPar;// структура параметров объекта "обозначение центра"
  ::memset(&cPar, 0, sizeof(cPar));
  cPar.x = 50;                  // точка привязки
  cPar.y = 50;
  cPar.type = 2;                // тип обозначения центра - две оси
  cPar.lenXpTail = 40;          // длина "хвостиков"
  cPar.lenXmTail = 60;
  cPar.lenYpTail = 20;
  cPar.lenYmTail = 45;
  ::ksCentreMarker(&cPar);    // создать объект "обозначение центра"
```

NewGroup, EndGroup - Пример использования

```
void NewGroup_Example (void) {

reference gr ;
double x,y,m,ang;
Phantom FANTOM;

gr = NewGroup(1); /* задание группы объектов */

LineSeg (-15,    0,  15,    0,  3);  /* объекты записываются   */
LineSeg (   0, -15,    0,  15,  3);  /* во временный список    */
Circle (0,  0,  10,  1);

EndGroup();   /* закончить формирование группы */

/* ввод точки с отображением фантома и копирование группы*/
FANTOM.phType=1;
FANTOM.type1.gr=gr;
```

```
FANTOM.type1.xBase=0; FANTOM.type1.yBase=0;
FANTOM.type1.scale=1; FANTOM.type1.ang=0;

RequestInfo  info;
memset(&info, 0, sizeof(info));
info.promt = " Точка привязки ";

if (Cursor(&info, &x, &y, &FANTOM))
CopyObj(gr, x, y, m, ang);

if (YesNo(Удалять?))
DeleteObj(gr);

}; /* NewGroup_Example */
```

AddObjGroup - Пример использования

```
void AddObjGroup_Example (void) {

reference gr,p ;

p = LineSeg(0, 0, 80, 0, 3); /* осевая линия */
gr = NewGroup(0); /* задание группы объектов */

LineSeg (10,  10,  10,  20,  1);  /* объекты записываются   */
LineSeg (10,  20,   40, 20,  1);  /* в модель текущего вида */
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

EndGroup();   /* закончить формирование группы */

AddObjGroup(gr, p); /* добавить объект p */
LightObj(gr, 1);/* подсветить группу */

if (Yes_No(Удалять группу?))
DeleteObj(gr);
```

```
}; /* AddObjGroup_Example */
```

SelectGroup - Пример использования

```
void SelectGroup_Example (void) {

reference gr ;

LineSeg (10,  10,  10,  20,  1);  /* объекты записываются   */
LineSeg (10,  20,   40, 20,  1);  /* в модель текущего вида */
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

LineSeg(0, 0, 80, 0, 3); /* осевая линия */

gr = NewGroup(0); /* определение группы объектов */
EndGroup();   /* закончить формирование группы */

/* добавить  объекты, расположенные внутри прямоугольника */
SelectGroup(gr, 1, 5, 5, 75, 35);

LightObj(gr, 1);/* подсветить группу */

if (Yes_No(Удалять группу?))
DeleteObj(gr);

}; /* SelectGroup_Example */
```

ExcludeObjGroup - Пример использования

```
void ExcludeObjGroup_Example (void) {

reference gr,p ;

gr = NewGroup(0); /* задание группы объектов */
```

```
LineSeg (10,  10,  10,  20,  1);  /* объекты записываются   */
LineSeg (10,  20,   40, 20,  1);  /* в модель текущего вида */
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

p = LineSeg(0, 0, 80, 0, 3); /* осевая линия */

EndGroup();   /* закончить формирование группы */

ExcludeObjGroup(gr, p); /* исключить объект p */
LightObj(gr, 1);/* подсветить группу */

if (Yes_No(Удалять группу?))
DeleteObj(gr);

}; /* ExcludeObjGroup_Example */
```

ksGetGroupName - Пример использования

```
//создаем рабочую группу
  reference gr = NewGroup (0);
     LineSeg (20, 20, 40, 20, 1);
     LineSeg (40, 20, 40, 40, 1);
     LineSeg (40, 40, 20, 40, 1);
     LineSeg (20, 40, 20, 20, 1);
  EndGroup ();

  //рабочую группу сохраняем в именной
  //именная группа хранится в документе
  SaveGroup (gr, "group1");

  char grName [255];
  //получим имя группы
  ksGetGroupName (gr,            // указатель на группу
```

```
            grName, // указатель строки для имени группы
255);// размер строки
  Message (grName);
```

ClearGroup - Пример использования

```
void ClearGroup_Example (void)
{
reference gr ;

gr = NewGroup(0);// задание группы объектов

LineSeg (10,  10,  10,  20,  1);// объекты записываются
LineSeg (10,  20,   40, 20,  1);// в модель текущего вида
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

EndGroup();// закончить формирование группы

//симметрия относительно оси X

if (Yes_No (Выполнять осевую симметрию))
{
SymmetryObj (gr,  0, 0, 100, 0, 0);
ClearGroup(gr);// очистка группы после использования
}
};
```

ksClearGroup - Пример использования

```
 reference gr = NewGroup(1); // временная группа геометрии
   ::LineSeg(100, 100, -100, -100, 1);
   ::LineSeg(10, 10, -10, -10, 1);
 EndGroup();

 // Удалить или нет временные объекты
```

```
bool deleteTmp = ::YesNo("Удалить временные объекты") == 1 ? true : false;

::ksClearGroup(gr, deleteTmp/*true - удалять временные объекты, false - не удалять*/);

::StoreTmpGroup(gr); // записываем результаты в документ
```

StoreTmpGroup- Пример использования

```
void StoreTmpGroup_Example (void) {
reference gr ;

gr = NewGroup(1); /* задание группы объектов */

LineSeg (-15,    0,  15,    0,  3);  /* объекты записываются   */
LineSeg (   0, -15,    0,  15,  3);  /* во временный список    */
Circle (0,  0,  10,  1);

EndGroup();   /* закончить формирование группы */

if (Yes_No(Фиксируем группу?)) {
StoreTmpGroup(gr);

}; /* StoreTmpGroup*/
```

GetGroup- Пример использования

```
void GetGroup_Example (void)
{
reference gr, p ;

gr = NewGroup(1);// задание группы объектов

LineSeg (-15,    0,  15,    0,  3);// объекты записываются
LineSeg (   0, -15,    0,  15,  3);// во временный список
Circle (0,  0,  10,  1);

EndGroup();// закончить формирование группы
```

```
SaveGroup (gr, Отверстие);
p = GetGroup (Отверстие);
LightObj (p, 1);
};
```

SaveGroup - Пример использования

```
void SaveGroup_Example (void) {

reference gr ;
char name[40];

gr = NewGroup(0); /* определение группы объектов */

LineSeg (10,  10,  10,  20,  1);  /* объекты записываются   */
LineSeg (10,  20,   40, 20,  1);  /* в модель текущего вида */
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

EndGroup();   /* закончить формирование группы */

LightObj(gr, 2);/* подсветить группу */

if (Yes_No(Сохранять группу в модели?)) {
ReadString(Имя группы, name, 40);
SaveGroup(gr, name);
}

LightObj(gr, 0);/* снять выделение группы */

}; /* SaveGroup_Example */
```

ksViewGetObjectArea - Пример использования

```
//определим группу выделения
reference  gr = ksViewGetObjectArea();
```

```
if (gr) {
//поставим временную группу в модель
   StoreTmpGroup(gr);

   //подсветим
   LightObj(gr, 1);
   Message("Область выделения");

   //выключим подсветку
   LightObj(gr, 0);
 }
else
   Message("Группы нет");
```

Пример использования функций навигации по объектам

```
//позиционирование в текущем виде текущего документа по всем элементам

void Iterator_Example (void)
{
reference itAllObj;
reference obj;
int count = 0;
char buf[128];

//Создать итератор
itAllObj = CreateIterator(ALL_OBJ , 0);

if (itAllObj)
{
if (ExistObj(obj = MoveIterator(itAllObj, 'F')))
{
// поиск первого объекта и следующего в цикле
do
{
LightObj(obj, 1);
count ;
sprintf(buf,"номер = %d", count);
```

```
Message(buf);
LightObj(obj, 0);
}
while (ExistObj (obj = MoveIterator (itAllObj, 'N')));
}
}

//Удалить итератор
DeleteIterator(itAllObj);
};
```

ReleaseReference - пример использования

```
reference p = LineSeg (100, 50, 200, 50, 1);
LightObj (p, 1);
/освобождается указатель на объект,
/после вызова ReleaseReference к объекту p обращаться нельзя
ReleaseReference (p);
LightObj (p, 0);
//взводится ошибка "В текущем документе объект не найден"
MessageBoxResult();
```

GetViewObjCount - пример использования

```
void GetViewObjCount_Example (void)
{
long n;
char buf[80];
LineSeg(20, 10, 20, 30, 1);
LineSeg(20, 30, 40, 30, 1);
LineSeg(40, 30, 40, 10, 1);
LineSeg(40, 10, 20, 10, 1);

n = GetViewObjCount(0); // в текущем виде/фрагменте
sprintf (buf,"Количество объектов=%5d);
Message(buf);
};
```

FindObj - пример использования

```
{
reference pLine;

LineSeg(0, 0, 50, 50, 1);
LineSeg(50, 50, 70, 20, 1);

pLine = FindObj(20, 20, 5);

LightObj(pLine, 1);
}
```

GetObjGabaritRect - пример использования

```
{
reference   pObj;
RectParam  Rect;

pObj = LineSeg(10, 10, 50, 50, 1);
GetObjGabaritRect(pObj, &Rect);
}
```

ksGetDocVariableArray, ksSetDocVariableArray - пример использования

```
//получим указатель на текущий графический документ
reference doc = ksGetCurrentDocument(1); // 0 - любой документ,
// 1- только графический документ
// 2 - только спецификацию
if (doc)
{
char buf [250];
//получить массив внешних переменных документа
reference arrayVar = ksGetDocVariableArray(doc); //указатель на документ
//или вставку фрагмента
for (int i=0, count = GetArrayCount(arrayVar); i < count; i)
{
VariableParam par;
```

```
//получить текущую переменную
GetArrayItem(arrayVar, i, &par, sizeof(VariableParam));
sprintf(buf, "имя = %s\nзначение = %f\nкомментарий = %s", par.name, par.value, par.note);
Message(buf);
par.value = 100;
strcat(par.note, "!!!");
//заменить текущую переменную в массиве
SetArrayItem(arrayVar, i, &par, sizeof(VariableParam));
}
//заменить значения внешних переменных документа
ksSetDocVariableArray (doc,// указатель на документ или вставку фрагмента
arrayVar,// указатель на динамический массив VARIABLE_ARR
1);// комментарии менять
}
else
Error ("Документ должен быть графическим");
```

ksSetObjConstraint - пример использования

```
//Установить ограничение "равенство радиусов двух дуг/окружностей"
reference p;
RequestInfo info;
//обнулить структуру info;
memset(&info, 0, sizeof(info));
double x, y;
info.prompt = "Укажите первую дугу или окружность";
int j = Cursor (&info, &x ,&y, 0);
if (j) {
if (ExistObj(p = FindObj (x, y, 1e6))) {
info.prompt = "Укажите вторую дугу или окружность";
j = Cursor (&info, &x ,&y, 0);
reference p1;
if (j) {
if (ExistObj (p1 = FindObj (x, y, 1e6))) {
ConstraintParam par;
memset (&par, 0, sizeof (par));
par.constrType = CONSTRAINT_EQUAL_RADIUS;
par.partner = p1;
```

```
ksSetObjConstraint (p, &par);
}
}
}
}
```

ksGetObjConstraints, ksDestroyObjConstraint - пример использования

```
static char *str [] = {"фиксировать точку", "точка на кривой",
"горизонталь", "вертикаль",
"параллельность двух прямых или отрезков",
"перпендикулярность двух прямых или отрезков",
"равенство длин двух отрезков",
"равенство радиусов двух дуг/окружностей",
"выравнивать две точки по горизонтали",
"выравнивать две точки по вертикали",
"совпадение двух точек"};
reference p;
RequestInfo info;
//обнулить структуру info;
memset(&info, 0, sizeof(info));
double x, y;
info.prompt = "Укажите объект";
int j = Cursor(&info, &x ,&y, 0);
if (j) {
if (ExistObj (p = FindObj (x, y, 1e6))) {
//вернуть ограничения данного объектка
reference arr = ksGetObjConstraints (p);
int count = GetArrayCount (arr);
char buf [255];
sprintf (buf, " число ограничений = %d ", count);
Message (buf);
ConstraintParam par;
for (int i = 0; i < count; i) {
GetArrayItem (arr, i, &par, sizeof (ConstraintParam));
sprintf (buf, " constrType=%s ",str [par.constrType-1]);
Message (buf);
switch (par.constrType) {
```

```
case CONSTRAINT_POINT_ON_CURVE : {//точка на кривой
LightObj (par.partner, 1);
sprintf (buf, "Подсветили партнера->index=%d ", par.partnerIndex);
Message (buf);
LightObj (par.partner, 0);
break;
}
case CONSTRAINT_MERGE_POINTS ://объединение точек
case CONSTRAINT_HOR_ALIGN_POINTS ://выравнивать точки по оси X
case CONSTRAINT_VER_ALIGN_POINTS : { //выравнивать точки по оси Y
LightObj(par.partner, 1);
sprintf (buf, "Подсветили партнера->index=%d\n наш index = %d ",
par.partnerIndex, par.index);
Message (buf);
LightObj (par.partner, 0);
break;
    }
case CONSTRAINT_FIXED_POINT : {//фиксировать точку
sprintf (buf, "наш index = %d ", par.partnerIndex);
Message (buf);
break;
}
case CONSTRAINT_PARALLEL ://параллельность двух
//прямых или отрезков
case CONSTRAINT_PERPENDICULAR ://перпендикулярность двух
//прямых или отрезков
case CONSTRAINT_EQUAL_LENGTH ://равенство длин двух отрезков
case CONSTRAINT_EQUAL_RADIUS : { //равенство радиусов двух
//дуг/окружностей
LightObj (par.partner, 1);
Message ("Подсветили партнера");
LightObj (par.partner, 0);
break;
    }
   }
if (YesNo("Удалить ограничение?"))
ksDestroyObjConstraint (p, &par);
}
```

```
}
 }
```

ksGetDimensionVariableName - пример использования

```
reference iter = ::CreateIterator(LDIMENSION_OBJ, 0); // создаем итератор по линейным размерам
 if (iter) {
 reference obj = ::MoveIterator(iter, 'F');     // первый объект
char buf[128];                    // буфер для имени переменной
  while (::ExistObj(obj)) {              // размер существует
if (::ksGetDimensionVariableName(obj, buf, 128)) // получим имя параметрической переменной размера
   ::Message(buf);                 // выведем имя переменнной
   else                        // у размера нет переменной
   ::Message("Размер не имеет переменной");    // сообщение
  obj = ::MoveIterator(iter, 'N');         // следующий объект
  }
 ::DeleteIterator(iter);               // удалим итератор
```

ExistObj, ExistGroupObj - пример использования

```
void ExistObj_Example (void) {

reference l1, gr ;

LineSeg (10, 10, 10, 20, 1); /* объекты записываются  */
LineSeg (10, 20,  40, 20, 1); /* в модель текущего вида */
l1 = LineSeg (40, 20,  40, 30, 1);
LineSeg (40, 30,  70, 30, 1);
LineSeg (70, 30,  70, 10, 1);

LineSeg(0, 0, 80, 0, 3); /* осевая линия */

gr = NewGroup(0); /* определение группы объектов */
LineSeg (70, 10,  10, 10, 1);
EndGroup();  /* закончить формирование группы */
```

```
/* добавить объекты в интерактивном режиме */
SelectGroup(gr, 0, 0, 0, 0, 0);

LightObj(gr, 1);/* подсветить группу */

if ((!(ExistGroupObj(gr))) && (YesNo("Удалять группу?"))) //удалить группу, если она пустая ?
DeleteObj(gr);
if (ExistObj(l1)) LightObj(l1,1);

}; /* ExistObj_Example */
```

DeleteObj - пример использования

```
void DeleteObj_Example (void) {

reference gr ;

gr = NewGroup(0); /* задание группы объектов */

LineSeg (10, 10, 10, 20, 1); /* объекты записываются  */
LineSeg (10, 20,  40, 20, 1); /* в модель текущего вида */
LineSeg (40, 20,  40, 30, 1);
LineSeg (40, 30,  70, 30, 1);
LineSeg (70, 30,  70, 10, 1);
LineSeg (70, 10,  10, 10, 1);

EndGroup();  /* закончить формирование группы */
if (Yes_No(Удалять группу?))
DeleteObj(gr);
}; /* DeleteObj_Example */
```

CopyObj - пример использования

```
void CopyObj_Example (void)
{
reference gr ;
```

```
double x,y,ang; scale;

/* задание группы объектов */
gr = NewGroup(0);

/* объекты записываются в модель текущего вида */
LineSeg (-15,  0, 15,  0, 3);
LineSeg (  0, -15,  0, 15, 3); /*  */
Circle (0, 0, 10, 1);

/* закончить формирование группы */
EndGroup();

RequestInfo info;
memset(&info, 0, sizeof(info));
info.promt = " Новое положение базовой точки ";

/* задание параметров копирования */
if ((Cursor (&info, &x, &y, 0)) &&
  (ReadFloat (Угол поворота, 0, 0, 360, ang)) &&
  (ReadFloat(Масштаб, 1, 0, 999, scale)))
CopyObj(gr, 0, 0, x, y, ang, scale);/* копирование группы gr */
/* базовая точка совпадает с 0,0 */
};
```

ksCopyObj - пример использования

```
ViewParam par;
  int number = 5;

  par.x = 20;
  par.y = 60;
  par.scale = 1;
  par.ang = 0;
  par.color = RGB(10,20,10);
  par.state = stACTIVE;
  strcpy(par.name, "user view");
```

```
//создали вид
reference v = CreateSheetView(&par, &number);

//создали слой
Layer(5);

LineSeg(20, 10, 20, 30, 1);
LineSeg(20, 30, 40, 30, 1);
LineSeg(40, 30, 40, 10, 1);
LineSeg(40, 10, 20, 10, 1);

//копируем вид (для вида точки задаются в листовых координатах)
reference p = ksCopyObj(l1, 20, 60, 40, 80, 1, 0 );

LightObj (p, 1);
Message("Подсветили скопированный вид");
LightObj (p, 0);
//сдвинули скопированный вид
MoveObj(p, 50, 0);
```

SymmetryObj - пример использования

```
void SymmetryObj_Example (void) {

reference gr ;

gr = NewGroup(0); /* задание группы объектов */

LineSeg (10, 10, 10, 20, 1); /* объекты записываются  */
LineSeg (10, 20,  40, 20, 1); /* в модель текущего вида */
LineSeg (40, 20,  40, 30, 1);
LineSeg (40, 30,  70, 30, 1);
LineSeg (70, 30,  70, 10, 1);
LineSeg (70, 10,  10, 10, 1);

EndGroup();  /* закончить формирование группы */
```

```
/* симметрия относительно оси X */

if (Yes_No(Выполнять осевую симметрию))
SymmetryObj(gr, 0, 0, 100, 0, 0);

}; /* SymmetryObj_Example */
```

ksSymmetryObj - пример использования

```
reference cir =Circle(20, 10, 20, 1);

 reference p = ksSymmetryObj(cir, 20, 35, 40, 35, 1);   // указатель на объект

 LightObj (p, 1);
 Message("подсветили полученный объект");
 LightObj (p, 0);
```

TransformObj - пример использования

```
reference pObj;
RequestInfo info;
double x, y;
memset(&info, 0, sizeof (info));
info.prompt = "Укажите объект";
reference g;
int j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj (pObj = FindObj (x, y, 1e6)))
{
Mtr (-10,-10,0, 2);
TransformObj (pObj);
DeleteMtr();
}
}
```

ksMovePointOnCurve - пример использования

```
//неравномерное размещение точек и перпендикуляров к ним на кривой
 static double pointLen[] = { 0, 5, 10, 15, 10, 5 };
 int count = 6;
 reference pObj;
 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите точку на кривой";
 int j = Cursor(&info, &x ,&y, 0);
 if (j) {
  if(ExistObj(pObj = FindObj(x, y, 1e6))){
   //узнаем тип объекта
   int type = GetObjParam(pObj, 0, 0, 0); //указатель на графический объект
   if (type == CIRCLE_OBJ    ||  //окружность
      type == ARC_OBJ      ||  //дуга
      type == NURBS_OBJ     ||  //nurbs
      type == LINESEG_OBJ    ||  //отрезок
      type == BEZIER_OBJ    ||  //bezier
      type == CONTOUR_OBJ    ||  //контур
      type == POLYLINE_OBJ   ||  //полилиния
      type == ELLIPSE_OBJ    ||  //эллипс
      type == ELLIPSE_ARC_OBJ  ||  //дуга эллипса
      type == RECTANGLE_OBJ   ||  //прямоугольник
      type == REGULARPOLYGON_OBJ   //многоугольник
     ) {

    for (int i = 0; i < count; i) {
     if (ksMovePointOnCurve(pObj, //указатель на кривую
            &x, &y,   //координаты точки
            pointLen[i], //расстояние на которое нужно сместить точку
           1)) {   //направление продвижения точки(1 - в направлении построения кривой, -
1 в обратном направлении)
      //построим точку
      Point(x, y, 0);
      //найдем перпендикуляр к кривой в точке
      double angl = ksGetCurvePerpendicular(pObj, x, y);
      //построим перпендикуляр (отрезок длиной 10 мм)
```

```
    double x1 = x, y1 = y, x2 = x, y2 = y;
    MovePoint ( &x1, &y1, angl, 5);
    MovePoint ( &x2, &y2, angl180, 5);
    LineSeg (x1, y1, x2, y2, 2);
   }
  }
 }
 else
  Error("Выбранный объект не кривая");
}
}
```

ksWriteGroupToClip, ksReadGroupFromClip - пример использования

```
//делаем активным документ 1.cdw
 OpenDocument("c:\\1.cdw", 0);
 double x, y, x1,y1;
 RequestInfo info;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите начальную точку окна";
 if (Cursor(&info, &x, &y, 0)) {
  info.prompt = "Укажите конечную точку окна";
  Phantom phantom;   // ltdefine.h
  memset(&phantom, 0, sizeof(phantom));
  //Задаем тип резиноыой нити - прямоугольник и передаем в Cursor, чтобы ввести область выделения
  phantom.phType = 3; //прямоугольник
  phantom.type3.xBase = x;
  phantom.type3.yBase = y;
  if (Cursor(&info, &x1, &y1, &phantom)) {
   //создадим пустую рабочую группу
   reference gr = NewGroup(0);
   EndGroup();
   //помещаем в группу геометрию, которая попала в область выделения
   if (SelectGroup(gr, 1 ,x, y, x1, y1)) {
    // группу выделения записываем в буфер обмена
    int rez = ksWriteGroupToClip(gr,     // указатель на группу
                1); // 1 - с копированием; 0 - с удалением из текущего документа
```

```
    if (rez) {
   //делаем активным другой документ, например 1.frw, куда нужно перенести геометрию
из буфера обмена
     OpenDocument("d:\\0\\1.frw", 0);

     //обнулить структуру info;
     memset(&info, 0, sizeof(info));
     memset(&phantom, 0, sizeof(phantom));
     int j = 1;
     //группу считываем из буфера обмена
     phantom.type1.gr = ksReadGroupFromClip();
     phantom.phType = 1; //сдвиг группы
     phantom.type1.scale = 1; //сдвиг группы
     while (j) {
      j = Cursor(&info, &x ,&y, &phantom);
      if ( j == -1) {
       MoveObj(phantom.type1.gr, x, y); //смещаем группу в новый  цетр
       StoreTmpGroup(phantom.type1.gr); //временную группу делаем постоянной
       ClearGroup(phantom.type1.gr);
       DeleteObj(phantom.type1.gr);
       //группу считываем из буфера обмена
       phantom.type1.gr = ksReadGroupFromClip(); //временная группа
      }
     }
    }
   }
  }
 }
```

MoveObj - пример использования

```
void MoveObj_Example (void) {

reference gr ;
double x1,y1,x2,y2;

gr = NewGroup(0); /* задание группы объектов */
```

```
LineSeg (-15,  0, 15,  0, 3); /* объекты записываются  */
LineSeg (  0, -15,  0, 15, 3); /* в модель текущего вида */
Circle (0, 0, 10, 1);

EndGroup();  /* закончить формирование группы */

RequestInfo info;
 memset(&info, 0, sizeof(info));
info.commands = " Базовая точка ";/* задание вектора сдвига */
if (Cursor(&info,&x1,&y1,0)){
   info.commands = " Ее новое положение ";/* задание вектора сдвига */
  if (Cursor(&info,&x2,&y2, 0))
    MoveObj(gr, x2-x1, y2-y1, 0); /* сдвиг группы gr */
}
}; /* MoveObj_Example */
```

RotateObj - пример использования

```
void RotateObj_Example (void)
{
reference gr ;
double x,y,ang;

gr = NewGroup (0);//задание группы объектов

LineSeg (-15, 0, 15, 0, 3);// объекты записываются
LineSeg (0, -15, 0, 15, 3);// в модель текущего вида
Circle (0, 0, 10, 1);

EndGroup();  /* закончить формирование группы */
RequestInfo info;
memset (&info, 0, sizeof(info));
info.promt = " Центр поворота ";

// задание параметров поворота
if ((Cursor(&info,&x,&y, 0)) && (ReadFloat (Угол поворота, 0, 0, 360, ang)))
RotateObj (gr, x, y, ang);// поворот группы gr
};
```

DecomposeObj - пример использования

```
{
reference p;
LDimParam linPar;
memset(&linPar, 0,sizeof(LDimParam));

//параметры текста
 linPar.tPar.bitFlag = _AUTONOMINAL|_PREFIX|_TOLERANCE|_DEVIATION|_UNIT;
 linPar.tPar.sign = 1; //диаметр
 linPar.tPar.pText = CreateArray(CHAR_STR_ARR ,0);

 AddArrayItem(linPar.tPar.pText, -1, 2отв., 6); // _PREFIX
 AddArrayItem(linPar.tPar.pText, -1, H12, 4);  // _TOLERANCE
 AddArrayItem(linPar.tPar.pText, -1, мм, 5);   // _UNIT

//параметры привязки
 linPar.sPar.ps = 0;      // 0-горизонтальный
 linPar.sPar.x1 = 50; linPar.sPar.y1 = 50; // 1-ая точка
 linPar.sPar.x2 = 70; linPar.sPar.y2 = 60; // 2-ая
 linPar.sPar.dy = -20; // вектор определяющий положение размерной линии
 linPar.sPar.dx = 0;
 linPar.sPar.basePoint = 1; // признак 1-dx, dy -откладывать от первой точки, 2- от второй

// параметры отрисовки линейного размера
 linPar.dPar.textPos  = 0;  // автоматическая простановка
 linPar.dPar.textBase = 0;  // от средины размера
 linPar.dPar.pl1     = 0;  // 1-ая выносная линия есть
 linPar.dPar.pl2     = 0;  // 2-ая выносная линия есть
 linPar.dPar.pt1     = 1;  // тип стрелки у 1-ой выносной линии 1-изнутри
 linPar.dPar.pt2     = 1; // тип стрелки у 2-ой выносной линии 1-изнутри
 linPar.dPar.shelfDir  = -1; // полка направлена влево
 linPar.dPar.ang    = -30; // угол наклона ножки
 linPar.dPar.length   = 20; // длина ножки

 p = LinDimension(&linPar); //параметры линейного размера
```

```
DecomposeObj(p, 0, 5, 0);
}
```

ksApproximationCurve - пример использования

```
if (ksGetCurrentDocument(1)) {
 reference pObj;
 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите кривую для аппроксимации";
 int j;
 double eps    = 0.1;  //точность аппроксимации
 bool curentLayer = true;//строить на текущий слой
 bool delParent  = false; //не удалять подложку
 do {
  j = Cursor(&info, &x ,&y, 0);
  if (j) {
   if(ExistObj(pObj = FindObj(x, y, 1e6))){
    j = GetObjParam(pObj, 0,0,0);
    if (j == BEZIER_OBJ || j == NURBS_OBJ || j == ELLIPSE_OBJ || j == ELLIPSE_ARC_OBJ
      || j == EQUID_OBJ || j == CONTOUR_OBJ) {

     reference pApp = ksApproximationCurve(pObj,      // указатель на кривую
                  eps,     // точность аппроксимации 1e-7...1
                  curentLayer,
                  0, 1);// тип размещения по слоям 0 - на слой кривой 1- в текущий слой
     LightObj(pApp, 1);
     if (delParent)
      DeleteObj(pObj);
     Message("Аппроксимированная кривая");
     LightObj(pApp, 0);
    }
    else
     Error("Неверно указана кривая для аппроксимации");
   }
  }
 }while (j);
```

```
}
else
 Error("Документ не активизирован или не является листом/фрагментом");
```

ksClearRegion - пример использования

```
reference gr = NewGroup(0); // группа геометрии, представляющая область очистки
 ::Circle(0, 0, 40, 1);
EndGroup();

reference gr1 = NewGroup(1); // группа геометрии, которую нужно очистить
 ::LineSeg(100, 100, -100, -100, 1);
 ::LineSeg(10, 10, -10, -10, 1);
EndGroup();

// очистка заданной области
::ksClearRegion(gr1/*группа которую надо очистить*/, gr/*область очистки*/, 1);

::StoreTmpGroup(gr1); // записываем результат в документ
```

ksTrimCurve - пример использования

```
reference iter = ::CreateIterator(ALL_OBJ, 0); // итератор по всем объектам
  if (iter) {
   reference obj = ::MoveIterator(iter, 'F');  // первый объект
  // если объект геометрический - усечем его
   if (::IsGeomObject(obj)) {
   reference newCurve = ::ksTrimCurve(obj, 0, 20, 0, -20, 1, false);
   if (newCurve) {
     ::LightObj(newCurve, 1);        // подсветим объект
     ::Message ("Усекли кривую");       // выдадим сообщение
     ::LightObj(newCurve, 0);       // погасим подсветку объекта
}
   }
   ::DeleteIterator(iter);           // удалить итератор
  }
```

LightObj - пример использования

```
void LightObj_Example (void) {

reference gr ;
char name[40];

gr = NewGroup(0); /* определение группы объектов */

LineSeg (10, 10, 10, 20, 1); /* объекты записываются  */
LineSeg (10, 20,  40, 20, 1); /* в модель текущего вида */
LineSeg (40, 20,  40, 30, 1);
LineSeg (40, 30,  70, 30, 1);
LineSeg (70, 30,  70, 10, 1);
LineSeg (70, 10,  10, 10, 1);

EndGroup();  /* закончить формирование группы */

LightObj(gr, 1);/* подсветить группу */

if (Yes_No(Сохранять группу в модели?)) {
ReadString(Имя группы, name, 40);
SaveGroup(gr, name);
}

LightObj(gr, 0);/* снять выделение группы */

}; /* LightObj_Example */
```

GetObjParam, SetObjParam - пример использования

```
void GetObjParam_Example(void) {

char buf [128];
reference p;
LineSegParam par;
int t;
Mtr(30, 20, 45, 1);
```

```
p = LineSeg(0, 0, 30, 0, 1);

/* взять параметры отрезка */

t = GetObjParam(p, &par, sizeof(par), ALLPARAM);
sprintf(buf, " t=%d, x1=%4.1f y1=%4.1f x2=%4.1f y2=%4.1f tl=%d",t,
            par.x1, par.y1 ,par.x2, par.y2 , par.style);
Message(buf);

/* заменить параметры отрезка */
par.x2 = 0; par.y2 = 40;  par.style = 2;

if(SetObjParam(p, &par, sizeof(par), ALLPARAM))
Message(" Изменили объект");
else  MessageBoxResult();

DeleteMtr();

} /* _Example */
/
**********************************************************************
**/
/*Второй пример использования GetObjParam и SetObjParam*/
// редактирование позиционной линии выноски
  reference pObj;
  RequestInfo info;
  double x, y;
  memset(&info, 0, sizeof(info));
  info.prompt = "Укажите позиционную линию выноски";
  int j = Cursor(&info, &x ,&y, 0);
  if (j) {
   if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
    if (GetObjParam(pObj, 0,0,0) == POSLEADER_OBJ) {
     PosLeaderParam leaderPar;
     GetObjParam(pObj,  //указатель на графический объект
          &leaderPar, //указатель на структуру параметров
          sizeof(LeaderParam),  //размер структуры параметров
          ALLPARAM); //тип считывания параметров
```

```
    leaderPar.x = 10;
    leaderPar.y = 10;
    int rez = SetObjParam(pObj,  //указатель на графический объект
              &leaderPar, //указатель на структуру параметров
              sizeof(LeaderParam),  //размер структуры параметров
              ALLPARAM); //тип считывания параметров
    if (rez) {
     LightObj(pObj,1);
     Message ("линия выноски по параметрам");
     LightObj(pObj,0);
    }
    else
     MessageBoxResult(); //сообщение об ошибке
   }
   else
    Error("Объект не позиционная линия выноски");
  }
 }
```

ksChangeObjectInLibRequest - Пример использования

```
{
#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

int type, flag=0;

//функция обратной связи для выполнения цикла в Cursor
int far __export pascal CallBackC (int com, double *x, double *y, RequestInfo *info,
                                   void *phantom, int/* dynamic */) {
 Phantom *valueub = (Phantom *)phantom;

rub->type1.ang  = 90;
ksChangeObjectInLibRequest (&info, &rub);

switch (com) {
   case 1:
```

```
    case 2:
      type = com;
    break;
    case -1: //поставить в модель
      MoveObj(rub->type1.gr, *x, *y);
      StoreTmpGroup(rub->type1.gr); //поставить временную группу в вид
      ClearGroup(rub->type1.gr);
      break;
  }

//группа для фантома должна быть временная и обновляться при изменении вида
//отрисовки
  if (rub->type1.gr)
    DeleteObj(rub->type1.gr);
  rub->type1.gr = NewGroup(1); // временная группа
    if((flag==1 && com==1)||(flag==2 && com==2))
    type = 3;
//обновляется не только изображение но и меню для запроса
    switch (type ) {
      case 1:
        Circle (0, 0, 20, 1);
        info->commands =  !Квадрат !Треугольник ;
        flag = 1;
        break;
      case 2:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 0, 20, 1);
        LineSeg(0, 20,  -10, 0, 1);
        info->commands =  !Окружность !Квадрат;
        flag = 2;
        break;
      case 3:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 10, 20, 1);
        LineSeg(10, 20,  -10, 20, 1);
        LineSeg(-10, 20,  -10, 0, 1);
        info->commands =  !Окружность !Треугольник;
        flag = 0;
```

```
      break;
    }
  EndGroup();

  return 1;
}

void Cursor_Example (void) {
  type = 1;
  int j = 1;
  struct Phantom  rub;
  rub.type1.xBase = 0;
  rub.type1.yBase = 0;

  rub.type1.scale = 1;
  rub.phType       = 1;

  rub.type1.ang  = 0;
  double x, y;
  rub.type1.gr = NewGroup(1); // временная группа
    Circle (0, 0, 20, 1);
  EndGroup();
  RequestInfo  info;
  memset(&info, 0, sizeof(info));
  info.commands = "!Квадрат !Треугольник";
  info.callBack = CallBackC;//указываем адрес функции обратной связи для Cursor,
Cursor(&info , &x, &y, &rub);

}; /* Cursor*/
}
```

ksSaveFile - Пример использования

```
//создать массив строк
reference arrName = CreateArray (CHAR_STR_ARR, 0);

//выберем файлы, которые хотим перезаписать
if (ChoiceFiles ("*.cdw",    // расширение имени файла
```

```
"Чертежи(*.cdw)|*.cdw|Фрагменты(*.frw)|*.frw|Все файлы (*.*)|*.*|", // фильтр поиска
//(0 - формируется автоматически)
arrName))// массив неопределенной длины на строки CHAR_STR_ARR
{
int count = GetArrayCount (arrName);
char fileName[ MAX_TEXT_LENGTH ];
for (uint i = 0; i < count; i++)
{
char oldFileName[ MAX_TEXT_LENGTH ];
//выберем старое имя файла
GetArrayItem (arrName,// указатель на массив
i,// индекс в массиве
//(нумерация начинается с 0)
oldFileName,// указатель на структуру элемента
MAX_TEXT_LENGTH);// размер структуры элемента
//выберем новое имя файла
if (ksSaveFile ("*.cdw",// расширение имени файла
oldFileName,// имя файла по умолчанию
"Чертежи(*.cdw)|*.cdw|Фрагменты(*.frw)|*.frw|Все файлы (*.*)|*.*|",
// фильтр поиска (0 - формируется автоматически)
fileName,// буфер для имени файла
MAX_TEXT_LENGTH , // размер отведенного буфера name
1)) // 1-с подключением окна
// предварительного просмотра
{
//если новое и старое имя не совпадают, перезапишем документ
if (lstrcmp (oldFileName, fileName))
{
reference doc = OpenDocument (oldFileName, 1);
if (doc)
{
SaveDocument (doc, fileName);
CloseDocument (doc);
}
}
}
else
break;
```

```
}
}
//удалить массив
DeleteArray(arrName);
```

ksChoiceFileAppointedDir - Пример использования

```
char fileName[ MAX_TEXT_LENGTH ];
if (ksChoiceFileAppointedDir ("*.cdw",// расширение имени файла
"Чертежи(*.cdw)|*.cdw|Фрагменты(*.frw)|*.frw|Все файлы (*.*)|*.*|",
// фильтр поиска (0 - формируется автоматически)
fileName// буфер для имени файла
MAX_TEXT_LENGTH, // размер отведенного буфера name
1,  // -с подключением окна предварительного просмотра,
sptSYSTEM_FILES))
{
OpenDocument (fileName, 0);
}
```

ksGetCurrentDocument - Пример использования

```
if (ksGetCurrentDocument (1)) // документ должен быть графическим
{
//получим указатель на технические требования
reference pTT = GetReferenceDocumentPart(1);
if (pTT)
{
TechnicalDemandParam par;
//получим параметры описания ТТ
GetObjParam(pTT, &par, sizeof(par), TECHNICAL_DEMAND_PAR);
char buf[128];
sprintf (buf, "число строк ТТ =%d",par.strCount);
Message(buf);
//открываем ТТ на редактирование
//(ТТ должны быть открыты, чтобы менять тексты)
OpenTechnicalDemand (par.pGab,//динамический массив
//листов технических требований
//или 0
```

```
par.style); //стиль текста для технических требований
//(если 0 - умолчательное значение)
//пройдемся по ТТ и получим текст
TextLineParam parLine;
for(int i = 0; i < par.strCount; i++)
{
//берем текущую строку
GetObjParam(pTT, &parLine, sizeof(TextLineParam), TT_FIRST_STR+i);
TextItemParam parItem;
for (int j=0, count1 = GetArrayCount(parLine.pTextItem); j < count1; j++)
{
//берем текущий компонент строки
GetArrayItem(parLine.pTextItem, j, &parItem, sizeof(TextItemParam));
strcat(parItem.s, "!!!");
//меняем текущий компонент строки
SetArrayItem(parLine.pTextItem, j, &parItem, sizeof(TextItemParam));
Message(parItem.s);
}
//меняем текущую строку
SetObjParam(pTT, &parLine, sizeof(TextLineParam), TT_FIRST_STR+i);
}
//закрываем ТТ
CloseTechnicalDemand();
return;
}
}
Error ("Документ должен быть графическим");
```

ksGetRelativePathFromSystemPath - пример использования

```
char * catalogName[] = {"папка системных файлов",
            "папка библиотек"    ,
            "папка временных файлов",
            "папка конфигурации"  ,
            "INI-файл" };
  //сформировать полный путь к заданному файлу
```

```
  RequestInfo info;
  memset(&info, 0, sizeof(info));
  info.title = "Папки файлов системы ";
  info.commands = "!Системные !Библиотеки !Временные !Конфигурация !INI-файл ";
  info.prompt = "Выберите нужную папку";
  int j;
  static char *buf = "user.ttt";
  char fileName[250];
  int typeCatalog;
  do {
   j = ::CommandWindow(&info);
   if (j > 0) {
    switch (j) {
     case 1 : typeCatalog = sptSYSTEM_FILES; break;// Относительно папки системных файлов
      case 2 : typeCatalog = sptLIBS_FILES ; break;// Относительно папки файлов библиотек
       case 3 : typeCatalog = sptTEMP_FILES ; break;// Относительно папки сохранения временных файлов
        case 4 : typeCatalog = sptCONFIG_FILES; break;// Относительно папкисохранения конфигурации системы
         case 5 : typeCatalog = sptINI_FILE  ; break;// Относительно полного имени INI-файла системы
    }
    //полный путь
   ::ksGetFullPathFromSystemPath(buf ,     // относительный путь к файлу (без системного пути)
                fileName,   // (результат) полный путь к файлу
                250,      // размер буфера
                typeCatalog); // путь установленного типа см. ksSystemPath
    string mess = "Полный путь к файлу user.ttt \n";
    mess = catalogName[j -1];
    mess = " :\n";
    mess = fileName ;
    Message ((char*)mess.c_str());

    char relName[250];
    //относительный путь
    ::ksGetRelativePathFromSystemPath(fileName, //полный путь к файлу
                 relName,  //(результат) относительный путь к файлу (без системного пути)
```

```
                250, //размер буфера
                typeCatalog);  //путь установленного типа см. ksSystemPath
   mess = "Относительный путь к файлу \n";
   mess = fileName;
   mess = "\n";
   mess = catalogName[j -1];
   mess = " :\n";
   mess = relName;
   Message ((char*)mess.c_str());

  }
 } while (j > 0);
```

ksGetFullPathFromSystemPath - пример использования

```
char * catalogName[] = {"папка системных файлов",
            "папка библиотек"    ,
            "папка временных файлов",
            "папка конфигурации"  ,
            "INI-файл" };
  //сформировать полный путь к заданному файлу
  RequestInfo info;
  memset(&info, 0, sizeof(info));
  info.title = "Папки файлов системы ";
  info.commands = "!Системные !Библиотеки !Временные !Конфигурация !INI-файл ";
  info.prompt = "Выберите нужную папку";
  int j;
  static char *buf = "user.ttt";
  char fileName[250];
  int typeCatalog;
  do {
   j = ::CommandWindow(&info);
   if (j > 0) {
    switch (j) {
     case 1 : typeCatalog = sptSYSTEM_FILES; break;// Относительно папки системных фай-
лов
     case 2 : typeCatalog = sptLIBS_FILES ; break;// Относительно папки файлов библиотек
```

```
    case 3 : typeCatalog = sptTEMP_FILES ; break;// Относительно папки сохранения временных файлов
    case 4 : typeCatalog = sptCONFIG_FILES; break;// Относительно папки сохранения конфигурации системы
    case 5 : typeCatalog = sptINI_FILE  ; break;// Относительно полного имени INI-файла системы
   }
   //полный путь
   ::ksGetFullPathFromSystemPath(buf ,     // относительный путь к файлу(без системного пути)
                fileName,   // (результат) полный путь к файлу
                250,      // размер буфера
                typeCatalog); // путь установленного типа см. ksSystemPath
   string mess = "Полный путь к файлу user.ttt \n";
   mess = catalogName[j -1];
   mess = " :\n";
   mess = fileName ;
   Message ((char*)mess.c_str());

   char relName[250];
   //относительный путь
   ::ksGetRelativePathFromSystemPath(fileName, //полный путь к файлу
                relName,  //(результат) относительный путь к файлу(без системного пути)
                250, //размер буфера
                typeCatalog);  //путь установленного типа см. ksSystemPath
   mess = "Относительный путь к файлу \n";
   mess = fileName;
   mess = "\n";
   mess = catalogName[j -1];
   mess = " :\n";
   mess = relName;
   Message ((char*)mess.c_str());

  }
 } while (j > 0);
```

ksGetRelativePathFromFullPath - пример использования

```
char mainName[250];
  //имя задающего файла
  if((::ChoiceFile("*.*", "Все файлы (*.*)|*.*|", mainName, 250))){
   char fileName[250];
   if((::ChoiceFile("*.*", "Все файлы (*.*)|*.*|", fileName, 250))){
    char relName[250];
    //относительный путь
    ::ksGetRelativePathFromFullPath(mainName, //полный путь к задающему файлу
                fileName,  //полный путь к требуемому файлу
                relName,   //(результат) относительный путь к требуемому файлу (без общей с задающим файлом части пути)
                250); // размер буфера

    string mess = "Задающий файл - ";
    mess = mainName;
    mess = " \n";
    mess = "Полный путь -";
    mess = fileName ;
    mess = " \n";
    mess = "Относительный путь -";
    mess = relName ;
    Message ((char*)mess.c_str());

    //полный путь
    char fullName[250];
    ::ksGetFullPathFromRelativePath(mainName,  //полный путь к задающему файлу
                relName,  //относительный путь к требуемому файлу (без общей с задающим файлом части пути)
                fullName,    //(результат) полный путь к требуемому файлу
                250);  // размер буфера

    mess = "Задающий файл - ";
    mess = mainName;
    mess = " \n";
    mess = "Относительный путь -";
    mess = relName ;
    mess = " \n";
    mess = "Полный путь -";
```

```
    mess = fullName;
    mess = " \n";
    Message ((char*)mess.c_str());
   }
  }
```

ksGetFullPathFromRelativePath - пример использования

```
char mainName[250];
  //имя задающего файла
  if((::ChoiceFile("*.*", "Все файлы (*.*)|*.*|", mainName, 250))){
   char fileName[250];
   if((::ChoiceFile("*.*", "Все файлы (*.*)|*.*|", fileName, 250))){
    char relName[250];
    //относительный путь
    ::ksGetRelativePathFromFullPath(mainName, //полный путь к задающему файлу
                fileName,  //полный путь к требуемому файлу
                relName,   //(результат) относительный путь к требуемому файлу (без общей
с задающим файлом части пути)
                250); // размер буфера

    string mess = "Задающий файл - ";
    mess = mainName;
    mess = " \n";
    mess = "Полный путь -";
    mess = fileName ;
    mess = " \n";
    mess = "Относительный путь -";
    mess = relName ;
    Message ((char*)mess.c_str());

    //полный путь
    char fullName[250];
    ::ksGetFullPathFromRelativePath(mainName,  //полный путь к задающему файлу
                relName,  //относительный путь к требуемому файлу (без общей с задающим
файлом части пути)
                 fullName,    //(результат) полный путь к требуемому файлу
```

```
                250);  // размер буфера

    mess = "Задающий файл - ";
    mess = mainName;
    mess = " \n";
    mess = "Относительный путь -";
    mess = relName ;
    mess = " \n";
    mess = "Полный путь -";
    mess = fullName;
    mess = " \n";
    Message ((char*)mess.c_str());
   }
  }
```

FullFileName, GetRightFileName - пример использования

```
void FileName_Example (void) {

char *s2;

//выделим буфер
s2 = new char [6];
strcpy(s2, "1.cdw");
unsigned int size2,size1 ;

//получим полное имя и поместим его в s2
size1=strlen(s2) 1;
if ((size2=FullFileName(s2, s2, size1)) >  size1) {

//требуемая длина size2 > size1, которую мы предоставили
delete [] s2;
s2 = new char[size2];
GetRightFileName(s2, size2);
 }

}; /* FileName_Example */
```

FullFileNameW, GetRightFileName - пример использования, Unicode

```
void FileNameW_Example (void) {

LPWSTR s2;

//выделим буфер
s2 = new char [6];
strcpy(s2, "1.cdw");
unsigned int size2, size1;

//получим полное имя и поместим его в s2
size1=strlen( s2 ) 1;
if ((size2=FullFileNameW( s2, s2, size1 )) > size1) {

//требуемая длина size2 > size1, которую мы предоставили
delete [] s2;
s2 = new char[size2];
GetRightFileName(s2, size2);
 }

}; /* FileName_Example */
```

UniqueFileName, RemoveUniqueFile - пример использования

```
char fileName[255];
char buf[128];
FILE * f;

void CreateSysFile() {
f = fopen(fileName ,"rt");
fputs("Это служебная информация",f);
fclose(f);
}

void ReadSysFile() {
f = fopen(fileName ,"rt");
fgets(buf, 128, f);
Message(buf);
```

```
fclose(f);
}

void UniqueFile_Example (void) {

//запросили имя служебного файла
UniqueFileName(fileName, 255);

CreateSysFile();
ReadSysFile();

//удалить служебный файл
RemoveUniqueFile(fileName);

}; /* UniqueFile_Example */
```

ksGetSpcObjForGeomWithLimit - пример использования

```
reference pObj;
RequestInfo  info;
double x, y;
memset (&info, 0, sizeof(info));
info.prompt = "Укажите макроэлемент болт";
reference g;
int j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj(x, y, 1e6)) && GetObjParam(pObj, 0, 0, 0) == MACRO_OBJ )
{
//найдем объект спецификации по заданной геометрии
reference spcObj = ::ksGetSpcObjForGeomWithLimit ("graphic.lyt",//имя
//библиотеки стилей
1,//номер
//стиля спецификации
pObj ,
0,//присланная
//геометрия входит
// в объект спецификации
```

```
1, //-первый объект
25,//номер раздела
313277777065.0);
if (spcObj)
{
char buf[TEXT_LENGTH];
//получим наименование объекта спецификации
ksGetSpcObjectColumnText (spcObj,//объект спецификации
SPC_CLM_NAME ,//тип колонки
1,//номер колонки данного типа
1,//номер блока
buf,//указатель
TEXT_LENGTH);//длина строки s
Message(buf);
}
else
Error("Это не объект спецификации Болт");
}
else
Error("Это не макроэлемент");
}
```

CreateSpcIterator, ksGetSpcObjectColumnText, ksSetSpcObjectColumnText - пример использования

```
reference spcObj = 0;
double numbObj; //уникальный номер объекта
 //создадим объект спецификации
 if( ::ksSpcObjectCreate("graphic.lyt", //имя библиотеки типов
     1,        //   номер типа спецификации
20,  0,    //номер раздела и подраздела
0,        //тип атрибута
0)) {    //базовый объект

   spcObj = ::ksSpcObjectEnd();
   //запомним номер объекта
   numbObj = ::ksGetSpcObjectNumber  ( spcObj);
 }
```

```
 //создадим итератор для навигации по объектам спецификации
 reference iter = ::CreateSpcIterator("graphic.lyt", 1, 0);

 if (!iter)
    ::MessageBoxResult();  // неудачное завершение - выдадим результат работы нашей функции
 else {
  reference spcObj;
  char buf[120];
  // найдем первый объект
  spcObj = ::MoveIterator(iter, 'F');
  if (spcObj && ::ExistObj(spcObj)) {
   //у всех объектов в обозначении добавим "-01"
   //а у объекта с номером  numbObj  - "-02"
   do {
    //возьмем текст обозначения
    ::ksGetSpcObjectColumnText  (spcObj
                  SPC_CLM_MARK, //тип колонки SPC_CLM_FORMAT...SPC_CLM_USER
                     1,    //номер колонки данного типа начиная с 1
                     buf, 120);          //указатель

    double n = ::ksGetSpcObjectNumber  (spcObj);
    //добавим к обозначению нужную добавку
    strcat(buf, n == numbObj ? "-02"  : "-01");
    //откроем объект на редактирование
    ::ksSpcObjectEdit(spcObj);
     //в этом блоке можно редактировать все колонки спецификации,
     //подключать или отключать ассоциированную геометрию
     //в нашем случае заменим обозначение
                  ::kSetSpcObjectColumnText     (SPC_CLM_MARK,  //тип  колонки SPC_CLM_FORMAT...SPC_CLM_USER
                   1,    //номер колонки данного типа начиная с 1
                   buf);          //указатель
    //закроем объет после редактирования
    ::ksSpcObjectEnd();
    //найдем следующий объект
```

```
      spcObj = ::MoveIterator(iter, 'N');
    } while(spcObj  &&  ::ExistObj(spcObj));
  }
  //удалим итератор
  ::DeleteIterator(iter);
 }
}
```

ksGetSpcDocumentPagesCount - пример использования

```
//функция для декомпозирования текущей спецификации и помещения ее во фрагмент

//найдем текущий документ спецификацию
reference iDoc = CreateIterator (SPC_DOCUMENT_OBJ , 0);

if (iDoc)
{
reference pDoc = MoveIterator (iDoc, 'F');
if (pDoc)
{
do
{
int state;
if (GetObjParam (pDoc, &state, sizeof (state), DOCUMENT_STATE)
&& state == stACTIVE)
break;
pDoc = MoveIterator(iDoc, 'N');
}
while (pDoc);
}
DeleteIterator(iDoc);//указатель на итератор
if (pDoc)
{
//найдем количество листов спецификации
int pageCount = ksGetSpcDocumentPagesCount (pDoc);

//найдем габариты одного листа спецификации
RectParam spcGabarit;
```

```
GetObjGabaritRect (pDoc, &spcGabarit);

//создадим фрагмент
DocumentParam doc;
memset (&doc, 0, sizeof (doc));
doc.regim = 0;
doc.type = 3;
CreateDocument (&doc);
for (int i = 0; i < pageCount; i)
{
//получили временную группу i-го листа спецификации
reference  group =  DecomposeObj (pDoc,//указатель на объект
0,//уровень разбиения:
//отрезки,дуги,тексты,точки;
0.4,//стрелка прогиба
uint8(i1));// 0 - разбиение объекта
// в СК вида,
// 1- в СК листа
if (group)
{
int column=i%3;
double x = (spcGabarit.pTop.x-spcGabarit.pBot.x 5)*column;
int row = i/3;
double y = (spcGabarit.pTop.y-spcGabarit.pBot.y 5)*row;

//сдвинули группу
MoveObj (group, x, -y);

//поставили в модель
StoreTmpGroup (group);//указатель группы
ClearGroup (group);
DeleteObj(group);
}
}
char buf[128];
sprintf (buf, "Преобразовано %d листов СП", pageCount);
Message (buf);
}
```

```
else
Error("Спецификация должна быть текущей");
}
```

ksGetSpcTableColumn, ksGetSpcColumnType , ksGetSpcColumnNumb - пример использования

```
char spwName[250];
 int j1;
 //выберем спецификацию
    if((j1 = ksChoiceFile("*.spw","спецификации(*.spw)|*.spw|Все файлы (*.*)|*.*|",
spwName, 250, 1)) != 0){
   //открыть документ
   reference pDoc = OpenDocument (spwName, 0);
   //создать итератор по объектам СП
   reference iter = CreateSpcIterator(0, 0, 0);
   //встаем на первый объект спецификации
   reference spcObj = MoveIterator(iter, 'F');
   if (spcObj && ExistObj(spcObj)) {
    do {
     //узнаем количество колонок у базового объекта спецификации
     int count = ksGetSpcTableColumn(0, 0, 0);
     // пройдем по всем колонкам
     for (uint i = 1; i <= count; i) {
      unsigned int  columnType, typeNumb, block;
      //для текущего номера определим тип колонки, номер исполнения и блок
      if (ksGetSpcColumnType(spcObj,         //объект спецификации
                    i,     // номер колонки, начиная с 1
                    &columnType, //тип колонки SPC_CLM_FORMAT...SPC_CLM_USER
                    &typeNumb,     //номер колонки данного типа
                    &block)) {  //номер блока
       char buf[250];
       //возьмем текст
       ksGetSpcObjectColumnText  (spcObj,       //объект спецификации
                         columnType, //тип колонки SPC_CLM_FORMAT...SPC_CLM_USER
                         typeNumb,     //номер колонки данного типа
                         block,
```

```
                        buf,        //указатель
                        250);       //длина строки s
        char buf1 [128];
        sprintf (buf1, "\ntype = %d номер =%d block = %d", columnType,typeNumb,block);
        strcat (buf, buf1);
        Message(buf);
        //по типу колонки, номеру исполнения и блоку определим номер колонки
        int colNumb = ksGetSpcColumnNumb( spcObj,       //объект спецификации
                        columnType, //тип колонки SPC_CLM_FORMAT...SPC_CLM_USER
                            typeNumb,    //номер колонки данного типа начиная
                            block);   //номер блока
        sprintf (buf, "i = %d colNumb =%d", i, colNumb);
        Message(buf);
       }
      }
      //встаем на следующий объект спецификации
      spcObj = MoveIterator(iter, 'N');
    } while(spcObj  &&  ExistObj(spcObj));
   }
   DeleteIterator(iter);
   CloseDocument(pDoc);
 }
```

ksGetSpcSheetSB, ksSetSpcSheetSB - пример использования

```
//откроем спецификацию
reference pDoc = OpenDocument ("d:\\0\\4.spw", 0);
if (pDoc)
{
//заполним массив листов сборки, которые нужно подключить к спецификации
reference arr = CreateArray(CHAR_STR_ARR, 0);
char buf [255];
strcpy (buf, "d:\\0\\1.cdw");
AddArrayItem (arr, -1, buf, strlen(buf)1);
strcpy (buf, "d:\\0\\2.cdw");
AddArrayItem (arr, -1, buf, strlen(buf)1);

//подключим листы сборочного чертежа к спецификации
```

```
ksSetSpcSheetSB (pDoc , arr);
DeleteArray (arr);

//просмотрим листы сборки, подключенные к спецификации
reference p = ksGetSpcSheetSB (pDoc);
if (p)
{
for (uint i=0, count = GetArrayCount( p); i < count; i)
{
char buf1[255];
GetArrayItem(p, i, buf1, 255);
Message(buf1);
}
}

SaveDocument(pDoc, 0);
CloseDocument(pDoc);
}
```

ksGetSpcSectionName - пример использования

```
// создадим итератор по вспомогательным объектам спецификации в текущем документе
reference iter = ::CreateSpcIterator (0, 0, 1/*вспомогательные объекты спецификации*/);
if (iter) {
// встаем на первый объект спецификации
reference spcObj = ::MoveIterator (iter, 'F');
// проверяем, что объект действительно существует
if (spcObj && ::ExistObj (spcObj)) {
do {
char spcSectionName[128]; // имя раздела спецификации
// запрашиваем имя раздела
if (::ksGetSpcSectionName (spcObj, spcSectionName, sizeof (spcSectionName)))
::Message (spcSectionName);
// встаем на следующий объект спецификации
spcObj = ::MoveIterator (iter, 'N');
}  while (spcObj  && ::ExistObj (spcObj));
}
```

```
::DeleteIterator(iter); // удаляем итератор
}
```

ksGetSpcStyleParam - пример использования

```
SpcStyleParam par;                              // параметры стиляСП
    ::memset(&par.layoutName1, 0, sizeof(SpcStyleParam)); // очищаем структуру
      // Получить параметры для стиля спецификации с номером numb из библиотеки
nameLib
::ksGetSpcStyleParam ("graphic.lyt",// имя библиотеки стилей
            1,// номер стиля спецификации
&par,// структура параметров
sizeof (SpcStyleParam),// размер структуры параметров
ALLPARAM);// вернуть все параметры стиля спецификации
    ::Message (par.layoutName1); // имя файла библиотеки оформлений для первого листа
    ::Message(par.tuning.predefinedTextFileName); // имя файла предопределенных текстов
     int count = ::GetArrayCount (par.arrColumn);// количество элементов в массиве
     for (int i = 0; i < count; i) {                // проходим по массиву стилей колонок СП
      SpcStyleColumnParam param;                    // структура параметров стиля
//колонки спецификации
      ::GetArrayItem(par.arrColumn, i, &param, sizeof(SpcStyleColumnParam));
// считываем i-й элемент массива
     }
```

Пример использования функций работы с описаниями спецификации

```
reference doc = ::ksGetCurrentDocument(0);          // текущий документ
  if (doc) {
    SpcDescrParam param;                            // параметры описания
      ::memset(&param, 0, sizeof(SpcDescrParam));       // очистить параметры
      ::strcpy(param.layoutName, "C:\\gr\\graphic.lyt"); // имя файла библиотеки
//стилей спецификаций
      param.styleId = 15;                           // номер стиля в библиотеке

      if (::ksAddSpcDescription(doc, &param)) {         // добавить описание в документ
      bool state;                                   // состояние описания СП
       if (::ksGetSpcDescription(doc, 0, &param, &state)) { // считать описание с индексом 0
         char buf[1000];                            // создать сообщение
```

```
        ::sprintf(buf, "индекс СП: %d,\nИмя библиотеки: %s,\nНомер стиля: %d,\nИмя файла: %s,\nСостояние: %s",
                  0, param.layoutName, param.styleId, param.spcName, state ? "Активный" : "Неактивный");
        ::Message(buf);                    // вывести сообщение
      }

    param.styleId = 2;                     // номер стиля в библиотеке
    if (::ksSetSpcDescription(doc, 0, &param, state))     // изменить описание с индексом 0
     if (::ksGetSpcDescription(doc, 0, &param, &state)) { // считать описание с индексом 0
       char buf[1000];                     // создать сообщение
      ::sprintf(buf, "индекс СП: %d,\nИмя библиотеки: %s,\nНомер стиля: %d,\nИмя файла: %s,\nСостояние: %s",
                  0, param.layoutName, param.styleId, param.spcName, state ? "Активный" : "Неактивный");
        ::Message(buf);                    // вывести сообщение
      }

    ::ksDeleteSpcDescription(doc, 0);      // удалить описание с индексом 0
    }
  }
```

ksSpcObjectCreate - пример использования

a)

```
#define   SPC_NAME      5     //наименование

//---------------------------------------------------------------------------------------------
//  создадим объект спецификации для раздела "Детали"
//---------------------------------------------------------------------------------------------
reference  EditSpcObj(reference geom){

 reference spcObj = 0;
 //если редактируем макроэлемент, то войдем в режим редактирования объекта
 if (::EditMacroMode()) {
  //найдем объект спецификации по геометрии
  spcObj = ::ksGetSpcObjForGeom("graphic.lyt" , //имя библиотеки типов
                          1,       //   номер типа спецификации
                          0,       //для макроредактирования
```

```
                        1, 1);
  //войдем в режим редактирования
  if (!::ksSpcObjectEdit(spcObj))
    spcObj  = 0;
 }

//если объекта нет, войдем в режим создания объекта спецификации
if(spcObj ||  ::ksSpcObjectCreate("graphic.lyt", //имя библиотеки типов
                        1,        //   номер типа спецификации
                        20,  0,    //номер раздела и подраздела
                        0,0)) {    //тип атрибута

  //наименование
  ::ksSpcChangeValue(SPC_NAME/*номер колонки*/ , 1, "Втулка", STRING_ATTR_TYPE );

  //подключим геометрию
  if (geom)
    ::ksSpcIncludeReference(geom, 1);

  spcObj = ::ksSpcObjectEnd();
  //если объект спецификации создан, дадим возможность пользователю на него посмотреть и отредактировать. Функцию нужно запускать вне Cursor и Placement
  if (spcObj)
    if (::ksEditWindowSpcObject(spcObj))
      return spcObj;
 }
 return 0;
}
```

б)

Для стандартных изделий в стиле спецификации (см. "graphic.lyt") колонка "Наименование" имеет тип значения "запись". Это значит, что наименование формируется по шаблону обозначения для данного стандартного изделия (например, болта). Шаблоны создаются на базе типов атрибутов и хранятся в библиотеке типов атрибутов, которая определена в стиле спецификации (например, "spc.lat").

При простановке болта из библиотеки из базы данных заполняется некоторая структура данных BOLT, которая несет информацию для построения геометрии болта и формирования обозначения.

```
 struct BOLT {
   float dr;/*диаметр резьбы*/
   float p;/*шаг резьбы */
   float s;/*размер под ключ*/
   float h;/*высота головки */
   float D;/* диаметр описанной окружности*/
   float d3;/*диаметр отв. в стержне*/
   float h2;/*высота подголовка*/
   float  L;/*длина стержня*/
   float l1;/*расстояние до отверстия в стержне*/
   float b;/*длина резьбовой части*/
   float d2;/*диаметр под головкой*/
   float z4;/*величина фаски конца*/
    short     k;// 1 - длина резьбы ниже ломаной (резьба только на b) ; 2 - выше ломаной
(резьба до головки и на в)
   unsigned short f;/*битовые маски*/
   short klass;/*класс точности*/
   unsigned short gost;/*номер госта*/
 };

#define   SPC_NAME      5     //наименование

//---------------------------------------------------------------------------------------------
//  создадим объект спецификации для раздела "Стандартные изделия"
//---------------------------------------------------------------------------------------------
reference EditSpcObj (BOLT &tmp,  reference geom) {

 reference spcObj = 0;
 //если редактируем макро объект , то войдем в режим редактирования объекта
 if (::EditMacroMode()) {
   //найдем объкт спецификации по геометрии
   spcObj = ::ksGetSpcObjForGeom("graphic.lyt" , //имя библиотеки типов
                                 1,        //   номер типа спецификации
                                 0,        //для макроредактирования
                                 1, 1);
   //войдем в режим редактирования
   if (!::ksSpcObjectEdit(spcObj))
     spcObj  = 0;
```

```
 }

 if(spcObj ||  ksSpcObjectCreate("graphic.lyt" , //имя библиотеки типов
                                 1,        //   номер типа спецификации
                              25,  0,   //номер раздела и подраздела определены в стиле спецификации
                                   308437807397.0 , 0)) //тип атрибута определен в библиотеке типов
атрибутов spc.lat

  uint uBuf;
  //исполнение
  if(!(tmp.f & ISPOLN)) //если исполнения нет
    ::ksSpcVisible(SPC_NAME, 2, 0); //выключим исполнение
  else {
    uBuf = (uint) 2;
    ::ksSpcVisible(SPC_NAME, 2, 1);
    ::ksSpcChangeValue(SPC_NAME , 2, &uBuf, UINT_ATTR_TYPE );
  }

  //изменим диаметр
  uBuf = (uint) tmp.dr;
  ::ksSpcChangeValue(SPC_NAME , 4, &uBuf, UINT_ATTR_TYPE );

  //отследим мелкий шаг
  if(!(tmp.f & PITCH)){//выключить шаг и его разделитель
    ::ksSpcVisible(SPC_NAME, 5, 0);
    ::ksSpcVisible(SPC_NAME, 6, 0);   //шаг
  }
  else {
    ::ksSpcVisible(SPC_NAME, 5, 1);
    ::ksSpcVisible(SPC_NAME, 6, 1);   //шаг
    ::ksSpcChangeValue(SPC_NAME , 6, &tmp.p, FLOAT_ATTR_TYPE);
  }

  // выключим поле допуска
  ::ksSpcVisible(SPC_NAME, 7, 0);

  //изменим длину
  uBuf = (uint) tmp.L;
```

```
  ::ksSpcChangeValue(SPC_NAME , 9, &uBuf, UINT_ATTR_TYPE );

  // выключим класс прочности
  ::ksSpcVisible(SPC_NAME, 10, 0);

  // выключим материал
  ::ksSpcVisible(SPC_NAME, 11, 0);

  // выключим покрытие
  ::ksSpcVisible(SPC_NAME, 12, 0);

  //изменим ГОСТ
  uBuf = (uint) tmp.gost;
  ::ksSpcChangeValue(SPC_NAME , 14, &uBuf, UINT_ATTR_TYPE );

  //подключим геометрию
  if (geom)
    ::ksSpcIncludeReference(geom, 1);

  spcObj = ::ksSpcObjectEnd();
  //если объект спецификации создан, дадим возможность пользователю на него посмотреть и отредактировать. Функцию нужно запускать вне Cursor и Placement
  if (spcObj)
    if (::ksEditWindowSpcObject(spcObj))
      return spcObj;
 }
 return 0;
}
```

в) пример создания шаблона обозначения для стандартного изделия "болт":

Шаблон обозначения можно создать визуальными средствами, предоставленными КОМПАС-ГРАФИК для создания типа атрибута. Этот способ не требует непосредственного программирования, но приводит к затратам времени на создание шаблона, а при потере данных в библиотеке требует повторного создания утраченного шаблона.

Можно создать шаблон обозначения средствами apptool из библиотеки.

Полное обозначение болта :

Болт 1 М12 х 1.25 -6g х 60.58.35Х.16 ГОСТ 7808 - 70

Представим это обозначение в виде компонент :

НомерИмя компонентыУмолчательное значениеТипОчередность сортировки

1 Имя элем.Болтстрока1
2 Исполнение1uint3
3 РезьбаМстрока0 (не сортируется)
4 Диаметр12uint 4 или 3 (сортирует подряд)
5 Разделительхстрока0 (не сортируется)
6 Шаг1.25float 5 или 3 (сортирует подряд)
7 Поле допуска-6gстрока6 или 3 (сортирует подряд)
8 Разделительхстрока0 (не сортируется)
9 Длина60uint7 или 3 (сортирует подряд)
10 Кл. прочности.58строка0 (не сортируется)
11 Материал.35Хстрока0 (не сортируется)
12 Покрытие.16строка0 (не сортируется)
13 ГОСТГОСТстрока0 (не сортируется)
14 Номер7808uint2
15 Разделитель-строка0 (не сортируется)
16 Год70строка0 (не сортируется)

```
//ключи определены в стиле спецификации
#define   ST_KEY1      100    //ключи для стандапртных изделий
#define   ST_KEY2_GOST  5      //для стандартных изделий
#define   ST_KEY3       1
#define   ATTR_TYPE_LIB_NAME_KEY3     "d:\\gr\\spc.lat"  имя библиотеки типов атрибутов


//---------------------------------------------------------------------------
//  создадим  тип атрибута болта
//---------------------------------------------------------------------------
void TypeAttrBolt() {

//Инициализация данных для создания шаблона

//заполним структуру типа атрибута
```

```
 ksAttributeType attrType;
 strcpy(attrType.header,"Болт");   // заголовок-комментарий типа
 attrType.rowsCount        = 1;    // кол-во строк в таблице
 attrType.flagVisible      = 1;    // видимый, невидимый   в таблице
 strcpy(attrType.password,"");     // пароль, если не пустая строка  - защищает от несанкционированного изменения типа
 attrType.columns        = CreateArray(ATTR_COLUMN_ARR,0); // динамический массив компонент записи
 attrType.key1 = ST_KEY1;
 attrType.key2 = ST_KEY2_GOST;
 attrType.key3 = ST_KEY3;
 attrType.key4 = 0;
 ColumnInfo parStruct1;

  //массив полей записи

  //  колонка 1 "Имя элем."
   strcpy(parStruct1.header, "Имя элем.");// заголовок-комментарий столбца
   parStruct1.type = STRING_ATTR_TYPE;    // тип данных в столбце - см.ниже
   parStruct1.key = 1;              // дополнительный признак,  очередность сортировки
   strcpy(parStruct1.def,"Болт");         // значение по умолчанию
    parStruct1.flagEnum =0;                  // флаг, включающий режим, когда значение поля атрибута будет заполняться из массива перечисленных значений (1-режим включен, 0-отключен
   parStruct1.fieldEnum = 0;         // массив неопределенной длины перечислений (строки)
    parStruct1.columns = 0;                 // массив неопределенной длины информации о колонках для записи

   //добавим компоненту в массив компонент
   AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

  //  колонка 2 "Исполнение"
   strcpy(parStruct1.header, "Исполнение"); // заголовок-комментарий столбца
   parStruct1.type = UINT_ATTR_TYPE;       // тип данных в столбце - см. "ltdefine.h"
   parStruct1.key = 3;              // дополнительный признак,  очередность сортировки
   strcpy(parStruct1.def,"1");            // значение по умолчанию

   //добавим компоненту в массив компонент
   AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));
```

```
// колонка 3 "Резьба"
 strcpy(parStruct1.header, "Резьба"); // заголовок-комментарий столбца
 parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 0;                  //   очередность сортировки
 strcpy(parStruct1.def,"M");          // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 4 "Диаметр"
 strcpy(parStruct1.header, "Диаметр"); // заголовок-комментарий столбца
 parStruct1.type = UINT_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 3;                  //   очередность сортировки
 strcpy(parStruct1.def,"12");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 5 "разделитель"
 strcpy(parStruct1.header, ""); // заголовок-комментарий столбца
 parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 0;                  //   очередность сортировки
 strcpy(parStruct1.def,"x");          // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 6 "Шаг"
 strcpy(parStruct1.header, "Шаг"); // заголовок-комментарий столбца
 parStruct1.type = FLOAT_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 3;                  //   очередность сортировки
 strcpy(parStruct1.def,"1.25");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 7 "Поле допуска"
```

```
  strcpy(parStruct1.header, "Поле допуска"); // заголовок-комментарий столбца
  parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
  parStruct1.key = 3;                  //   очередность сортировки
  strcpy(parStruct1.def,"-6g");          // значение по умолчанию

  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 8 "разделитель"   strcpy(parStruct1.header, ""); // заголовок-комментарий столбца
  parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
  parStruct1.key = 0;                  //   очередность сортировки
  strcpy(parStruct1.def,"x");          // значение по умолчанию

  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 9 "Длина"
  strcpy(parStruct1.header, "Длина"); // заголовок-комментарий столбца
  parStruct1.type = UINT_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
  parStruct1.key = 3;                  //   очередность сортировки
  strcpy(parStruct1.def,"60");         // значение по умолчанию

  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 10 "Кл. прочности"
  strcpy(parStruct1.header, "Кл. прочности"); // заголовок-комментарий столбца
  parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
  parStruct1.key = 0;                  //   очередность сортировки
  strcpy(parStruct1.def,".58");          // значение по умолчанию

  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 11 "Материал"
  strcpy(parStruct1.header, "Материал"); // заголовок-комментарий столбца
  parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см.ниже
```

```
 parStruct1.key = 0;                //   очередность сортировки
 strcpy(parStruct1.def,".35X");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 12 "Покрытие"
 strcpy(parStruct1.header, "Покрытие"); // заголовок-комментарий столбца
 parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 0;                //   очередность сортировки
 strcpy(parStruct1.def,".16");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// колонка 13 "ГОСТ"
 strcpy(parStruct1.header, "ГОСТ"); // заголовок-комментарий столбца
 parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 0;                //   очередность сортировки
 strcpy(parStruct1.def,"ГОСТ");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 14 "Номер"
 strcpy(parStruct1.header, "Номер"); // заголовок-комментарий столбца
 parStruct1.type = UINT_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 2;                //  очередность сортировки
 strcpy(parStruct1.def,"7808");         // значение по умолчанию

 //добавим компоненту в массив компонент
 AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 15 "разделитель"
 strcpy(parStruct1.header, ""); // заголовок-комментарий столбца
 parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
 parStruct1.key = 0;                //   очередность сортировки
 strcpy(parStruct1.def,"-");         // значение по умолчанию
```

```
  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

 // колонка 16 "Год"
  strcpy(parStruct1.header, "Год"); // заголовок-комментарий столбца
  parStruct1.type = STRING_ATTR_TYPE;  // тип данных в столбце - см. "ltdefine.h"
  parStruct1.key = 0;                 //   очередность сортировки
  strcpy(parStruct1.def,"70");         // значение по умолчанию

  //добавим компоненту в массив компонент
  AddArrayItem(attrType.columns, -1, &parStruct1, sizeof(parStruct1));

  // создать  шаблон  обозначения

 /*double  numbType =*/  ksCreateAttrType(&attrType,   // информация о типе атрибута
                          ATTR_TYPE_LIB_NAME_KEY3); // имя библиотеки типов атрибутов

 MessageBoxResult();  // проверяем результат работы нашей функции

 //удалим  массив колонок
 DeleteArray(attrType.columns);

}
```

г) пример простановки линии-выноски к объекту спецификации :

```
//фрагмент ресурса

STRINGTABLE {
 STR225 "Линию-выноску проставлять?"
 STR226 "Укажите позиционную линию-выноску"
 STR227 "!Создать новую линию-выноску !Подключить существующую"
 STR228 "Ошибка! Объект - не позиционная линия-выноска!"
}

MENU_POS_LEADER MENU
{
```

```
 MENUITEM "Подключить существующую", 2
 MENUITEM "Создать новую линию-выноску ", 1
}

extern TModule *module;

//---------------------------------------------------------------------------------------------
// Отрисовать позиционную линию-выноску
// Уже проверено , что объект спецификации есть
// Функцию нужно запускать вне Cursor и Placement
//---------------------------------------------------------------------------------------------
void DrawPosLeader(reference _spcObj) {
  RequestInfo info;
  bool flag = false;
  reference posLeater = 0;
  double x1, y1;
  char buf[128];

  do {
    memset(&info, 0, sizeof(info));
    info.commands = (char*)MENU_POS_LEADER;

    module->LoadString(STR226, buf, 128); //"Укажите позиционную линию-выноску"
    info.prompt = buf;
    int j1 = Cursor(&info, &x1, &y1, 0);
    switch (j1) {
      case 1: //Создать новую линию-выноску
        posLeater = ::ksCreateViewObject(POSLEADER_OBJ);
        flag = false;
        break;
      case 2: //Подключить существующую
        module->LoadString(STR226, buf, 128); //"Укажите линию-выноску."
        memset(&info, 0, sizeof(info));
        info.commands = buf;
        if(Cursor(&info, &x1, &y1, 0)) {
        posLeater = ::FindObj(x1, y1, 100);     // величина стороны окошка-ловушки с центром
x,y
          if (!(posLeater && ::GetObjParam(posLeater, 0, 0, 0) == POSLEADER_OBJ)) {
```

```
            module->LoadString(STR228, buf, 128); //"Ошибка! Объект-не позиционная линия-выноска!"
            Error(buf);
            posLeater = 0;
            flag = true;
          }
          else
            flag = false;
          break;
        }
        else
          flag = false;
        break;
      case -1:
       posLeater = ::FindObj(x1, y1, 100);     // величина стороны окошка-ловушки с центром x,y
        if (!(posLeater && ::GetObjParam(posLeater, 0, 0, 0) == POSLEADER_OBJ)) {
          module->LoadString(STR228, buf, 128); //"Ошибка! Объект-не позиционная линия-выноска!"
          Error(buf);
          posLeater = 0;
          flag = true;
        }
        else
          flag = false;
      break;
    }
  }  while(flag);

  //линия-выноска есть, подключим ее к объекту спецификации
  if (posLeater ) {
    //войдем в режим редактирования объекта спецификации
    if (::ksSpcObjectEdit(_spcObj)) {
      //подключим линию-выноску
      ::ksSpcIncludeReference(posLeater, true);
      //закроем объект спецификации
      ::ksSpcObjectEnd();
    }
```

```
  }
}
```

ksGetWorkWindowColor - пример использования

```
unsigned long color = ksGetWorkWindowColor ();
char buf [128];
sprintf (buf, "Цвет фона рабочего окна = %ld ", color);
Message (buf);
```

ksGetSysOptions, ksSetSysOptions - пример использования

```
if (ksGetCurrentDocument(1)) {
   SnapOptrions par;
   char mes[700];
   memset(&par, 0, sizeof(SnapOptrions));
   //получим параметры привязок
   ksGetSysOptions(SNAP_OPTIONS, &par, sizeof(SnapOptrions));
   sprintf(mes          , "Привязка 'Ближайшая точка '-%s\n", par.nearestPoint   ? "включена"
: "отключена");
  sprintf(mes+strlen(mes), "Привязка 'Середина      '-%s\n", par.nearestMiddle  ? "включена"
: "отключена");
   sprintf(mes+strlen(mes), "Привязка 'Пересечение    '-%s\n", par.intersect     ? "включена"
: "отключена");
   sprintf(mes+strlen(mes), "Привязка 'Касание        '-%s\n", par.tangentToCurve ? "включе-
на" : "отключена");
    sprintf(mes+strlen(mes), "Привязка 'Нормаль        '-%s\n", par.normalToCurve  ? "вклю-
чена" : "отключена");
    sprintf(mes+strlen(mes), "Привязка 'По сетке       '-%s\n", par.grid           ? "включена" :
"отключена");
  sprintf(mes+strlen(mes), "Привязка 'Выравнивание   '-%s\n", par.xyAlign       ? "включена"
: "отключена");
  sprintf(mes+strlen(mes), "Привязка 'Угловая привязка'-%s\n", par.angSnap        ? "вклю-
чена" : "отключена");
  sprintf(mes+strlen(mes), "Привязка 'Точка на кривой '-%s\n", par.pointOnCurve   ? "вклю-
чена" : "отключена");
        sprintf(mes+strlen(mes), "Общие настройки : динамически %sотслеживать\n",
par.commonOpt & SN_DYNAMICALLY   ? "" : "не ");
    sprintf(mes+strlen(mes), "                : %sотображать текст\n"       , par.commonOpt &
SN_ASSISTANT     ? "" : "не ");
```

```
    sprintf(mes+strlen(mes), "                    : %sучитывать фоновые слои и виды\n",
par.commonOpt & SN_BACKGROUND_LAYER ? "" : "не ");
    sprintf(mes+strlen(mes), "                    : %sподавить привязки\n", par.commonOpt &
SN_SUSPENDED ? "" : "не ");
  sprintf(mes+strlen(mes), " угловой шаг %f", par.angleStep);
  sprintf(mes+strlen(mes), " локальная привязка = %d", par.localSnap);
  Message(mes);
  //очистим структуру привязок
  memset(&par, 0, sizeof(SnapOptrions));
  //включим привязку Касание
  par.tangentToCurve = 1;
  //включим привязку Точка на кривой
  par.pointOnCurve = 1;
  //текст отображать
  par.commonOpt  = par.commonOpt | SN_ASSISTANT;
  par.angleStep = 10;
  //локальная привязка
  par.localSnap = SN_GRID;
  // заменим привязки
  ksSetSysOptions(SNAP_OPTIONS, &par, sizeof(SnapOptrions));
 }
```

AddStyle - пример использования

```
{
unsigned short StyleID;
CurveStyleParam   CrvStyle;

StyleID = AddStyle (CURVE_STYLE, &CrvStyle,
                    sizeof (CurveStyleParam), 0);
}
```

Пример создания стиля кривой, содержащего фрагменты (кривой "с картинками")

```
LibStyle  par;   //структура параметров стиля из библиотеки
 strcpy(par.fileName, "d:\\0\\1.lcs") ;
 par.styleNumber = 1;
 //получим некий стиль кривой, на базе которого сделаем свой стиль
```

```
unsigned short tl = AddStyle(CURVE_STYLE_EX,  &par, sizeof(par), 1);

CurveStyleParam curPar;
//берем параметры стиля
GetStyleParam(CURVE_STYLE_EX, tl, &curPar, sizeof(curPar));

//если стиль с картинками продолжаем
if (curPar.curveType == 2) {
  CurvePatternEx par;
  int count = GetArrayCount(curPar.pattern);
  //убеждаемся, что в стиле есть прерывистые участки
  if (count){
    //берем первый участок
    GetArrayItem(curPar.pattern, // указатель на массив
                 0,    // индекс в массиве (нумерация начинается с 0)
                 &par,// указатель на структуру элемента
                 sizeof(par));  // размер структуры элемента
    //устанавливаем признак, что картинку возьмем из фрагмента
    par.pictureType = 1;
    //удаляем  массивы полилиний прежней картинки
    DeleteArray(par.picture.polygon);
    DeleteArray(par.picture.fill);
    //подставляем фрагмент
    strcpy(par.frwName, "d:\\0\\1.frw");
    //заменяем первый участок в стиле
    SetArrayItem(curPar.pattern, // указатель на массив
                 0,    // индекс в массиве (нумерация начинается с 0)
                 &par,// указатель на структуру элемента
                 sizeof(par));  // размер структуры элемента

  }
}
//создаем новый стиль, который будет иметь картинку, взятую из фрагмента
unsigned short tl1 = AddStyle(CURVE_STYLE_EX,  &curPar, sizeof(curPar), 0);
//создадим окружность с этим стилем
Circle(100,100, 50, tl1);
```

GetStyleParam - пример использования

```
{
CurveStyleParam   CrvStyle;

GetStyleParam(CURVE_STYLE, 1,
                    &CrvStyle, sizeof(CurveStyleParam));
}
```

ksGetLibraryStylesArray - пример использования

```
//получим массив оформлений для графических документов из библиотеки graphic.lyt
reference styleArr = ksGetLibraryStylesArray ("d:\\Kompas56\\Sys\\graphic.lyt",
GRAPHIC_LAYOUT_STYLE_LIBRARY);// библиотека оформлений
if (styleArr)
{
char buf[128];

//определим количество стилей и отобразим на экране
int count = GetArrayCount(styleArr);
sprintf(buf, "count = %d", count);
Message(buf);

//в цикле получим информацию о каждом стиле и отобразим на экране
for (uint i = 0; i < count; i++)
{
LibraryStyleParam par;
GetArrayItem (styleArr,// указатель на массив
i,// индекс в массиве (нумерация начинается с 0)
&par,// указатель на структуру элемента
sizeof (LibraryStyleParam));// размер структуры элемента

sprintf(buf, "ID = %d\nname=%s", par.styleId, par.styleName);
Message(buf);
}
}
```

ksIsStyleInDocument, ksDeleteStyleFromDocument - пример использования

```
LibStyle  par;   //структура параметров стиля кривой
strcpy(par.fileName, "c:\\1.lcs") ;
par.styleNumber = 1;
par.typeAllocation = 1; //ссылка на библиотеку стилей
//проверяем, есть ли стиль в текущем документе
if (!ksIsStyleInDocument (CURVE_STYLE,  &par, sizeof (par), 1)) {
Message ("Стиля в документе нет");
unsigned short tl = AddStyle (CURVE_STYLE,  &par, sizeof (par), 1);
LineSeg (20, 20, 70, 20, tl);

CurveStyleParam par1;
int t = GetStyleParam (CURVE_STYLE, tl, &par1, sizeof (par1));
if (t) {
Message (par1.name);
//стиль уже должен быть
if (ksIsStyleInDocument (CURVE_STYLE,  &par, sizeof (par), 1))
Message ("Стиль в документе есть");
//удалим стиль из документа
ksDeleteStyleFromDocument (CURVE_STYLE,  &par, sizeof (par), 1);
}
   else
     Error ("ошибка");
 }
```

ksGetSystemVersion - пример использования

Получить версию системы

Пример :

Для версии 5.4 Release 2 Build 1

```
iMajor  = 5
iMinor  = 4
iRelease = 2
iBuild  = 1
```

ksGetSystemProfileString - пример использования

```
char buf[255];
if (ksGetSystemProfileString("Directories", "SYS", buf, 255))
Message(buf);
 else
   Error("Ключ не найден");
```

ksSystemPath - Пример использования

```
{
int BitCount;
char *Buff;

BitCount = ksSystemPath(NULL, 0, sptLIBS_FILES);
ksSystemPath(&Buff, BitCount, sptLIBS_FILES);
}
```

EnableTaskAccess, PumpWaitingMessages - пример использования

```
void EnableTaskAccess_Example (void)
{
EnableTaskAccess(0); //запретили доступ к задаче
for(int i=0; i<10000; i)
{
LineSeg(10, 10i, 20, 10i, 1);
if(!(i%100))
{
//посылаем необрабатываемое сообщение своему приложению
//для выполнения процесса в фоновом режиме

::PostAppMessage(::GetCurrentTask(), 0, 0, 0);

// через каждые 100 отрезков обрабатываем очередь сообщений
// при этом Windows получает возможность выполнить свои
// действия (например переключиться на другую задачу)

PumpWaitingMessages();
}
```

```
}
EnableTaskAccess(1); //разрешили доступ к задаче
}; /* EnableTaskAccess_Example */
```

IsEnableTaskAccess - пример использования

```
//проверяем доступ к задаче
bool flagEnableTaskAccess = IsEnableTaskAccess();
Message (flagEnableTaskAccess ? "Доступ разрешен" : "Доступ запрещен");
if (flagEnableTaskAccess)
{
//запрещаем доступ
EnableTaskAccess (0);
//проверяем доступ к задаче
flagEnableTaskAccess = IsEnableTaskAccess();
Message (flagEnableTaskAccess ? "Доступ разрешен" : "Доступ запрещен");
EnableTaskAccess (1);// разрешить доступ
}
```

ksSetCriticalProcess - пример использования

```
reference pObj;
 RequestInfo info;
 double x, y;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите объект";
 int j =1;
 while(j) {
  if(YesNo("Включить критический режим?")==1)
   ksSetCriticalProcess ();
  j = Cursor(&info, &x ,&y, 0);
  if(j && ExistObj(pObj = FindObj(x, y, 1e6))){
   LightObj(pObj, 1);
   Message("Выделили объект");
   LightObj(pObj, 0);
  }
  int rez = ReturnResult();
  Message(rez == etLibraryClose ? "принудительное завершение выполнения библиотеки"
```

```
                : "Это не принудительное завершение");
  if (rez != etLibraryClose)
   j = YesNo("Продолжать?")==1;
 }
```

Cursor- Пример использования

```
#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

int type, flag=0;

//функция обратной связи для выполнения цикла в Cursor
int far __export pascal CallBackC (int com, double *x, double *y, RequestInfo *info,
                                   void *phantom, int/* dynamic */) {
 Phantom *valueub = (Phantom *)phantom;
  switch (com) {
    case 1:
    case 2:
      type = com;
    break;
    case -1: //поставить в модель
      MoveObj(rub->type1.gr, *x, *y);
      StoryTmpGroup(rub->type1.gr); //поставить временную группу в вид
      ClearGroup(rub->type1.gr);
      break;
  }

//группа для фантома должна быть временная и обновляться при изменении вида
//отрисовки
  if (rub->type1.gr)
    DeleteObj(rub->type1.gr);
  rub->type1.gr = NewGroup(1); // временная группа
    if((flag==1 && com==1)||(flag==2 && com==2))
    type = 3;
//обновляется не только изображение но и меню для запроса
    switch (type ) {
```

```
      case 1:
        Circle (0, 0, 20, 1);
        info->commands =  !Квадрат !Треугольник ;
        flag = 1;
        break;
      case 2:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 0, 20, 1);
        LineSeg(0, 20,  -10, 0, 1);
        info->commands =  !Окружность !Квадрат;
        flag = 2;
        break;
      case 3:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 10, 20, 1);
        LineSeg(10, 20,  -10, 20, 1);
        LineSeg(-10, 20,  -10, 0, 1);
        info->commands =  !Окружность !Треугольник;
        flag = 0;
        break;
    }
  EndGroup();

  return 1;
}

void Cursor_Example (void) {
  type = 1;
  int j = 1;
  struct Phantom  rub;
  rub.type1.xBase = 0;
  rub.type1.yBase = 0;

  rub.type1.scale = 1;
  rub.phType       = 1;

  rub.type1.ang  = 0;
  double x, y;
```

```
  rub.type1.gr = NewGroup(1); // временная группа
    Circle (0, 0, 20, 1);
  EndGroup();
  RequestInfo  info;
  memset(&info, 0, sizeof(info));
  info.commands = "!Квадрат !Треугольник";
  info.callBack = CallBackC;//указываем адрес функции обратной связи для Cursor,
Cursor(&info , &x, &y, &rub);

}; /* Cursor*/

struct RequestInfo info;
memset(&info,0,sizeof(RequestInfo));     //Структура информации

info.prompt="Укажите начальную точку";
Cursor(&info,&xBase,&yBase,0);//Начальная точка
struct Phantom f;
memset(&f,0,sizeof(Phantom));     //Фантом
f.type2.xBase = xBase;
f.type2.yBase = yBase;
info.prompt="Укажите точку на окружности";
f.phType=7;  // тип "резиновой" нити - окружность
double x,y;
//если был запущен другой процесс то не запускается второй Cursor
if (!ksIsActiveProcessRunnig () && Cursor(&info,&x,&y,&f))
  Circle(xBase, yBase, DistancePntPnt (xBase, yBase, x,y),1);
else
  Message("Запущен другой процесс");

//Функция обратной связи Cursor для построения пользовательского фантома (в данном
случае "резиновый" отрезок)
int far __export pascal CallBackEskLinsvzLin(int com ,double *x,double *y,RequestInfo*,void
*phantom,int dynamic) {
  Phantom *valueub = (Phantom *)phantom;
  if (!dynamic) {     //Фиксация точки
    if (com == -1) {
      StoreTmpGroup(rub->type6.gr);
      ClearGroup(rub->type6.gr);
```

```
    }
    DeleteObj(rub->type6.gr);

    return 0;
  }
  else {
    if (fabs(xBase -*x) > 0.001 || fabs(yBase -*y) > 0.001) {
      DeleteObj(rub->type6.gr);
      rub->type6.gr = NewGroup(1);
        LineSeg(xBase, yBase, *x, *y, 1);
      EndGroup();
    }
    return 1;
  }
}

struct RequestInfo info;
memset(&info,0,sizeof(RequestInfo));     //Структура информации

info.prompt="Укажите начальную точку";
Cursor(&info,&xBase,&yBase,0);  //Начальная точка

info.dynamic=1;
info.callBack=CallBackEskLinsvzLin;

struct Phantom f;
memset(&f,0,sizeof(Phantom));     //Фантом
f.type2.xBase = xBase;
f.type2.yBase = yBase;
f.phType=6;  //пользовательский фантом
info.prompt="Укажите точку на линии";

f.type6.gr = NewGroup(1);
  LineSeg(xBase, yBase, xBase+0.1, yBase,1);
EndGroup();
double x,y;
Cursor(&info,&x,&y,&f);
```

Placement - Пример использования

```
#ifndef  __MATH_H
#include <math.h>
#endif

#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

int type, flag=0;

// Функция обратной связи, для выполнения цикла в Placement
int far __export pascal CallBackP (int com, double *x, double *y, double *angl,
                                   RequestInfo *info, void *phantom, int /* dynamic*/) {
  Phantom *valueub = (Phantom *)phantom;
  switch (com) {
    case 1:
    case 2:
      type = com;
    break;
    case -1: //поставить в модель
      MoveObj(rub->type1.gr, *x, *y);
      if(fabs(*angl) > 0.001)
        RotateObj(rub->type1.gr, *x, *y, *angl);
      StoryTmpGroup(rub->type1.gr); //поставить временную группу в вид
      ClearGroup(rub->type1.gr);
      break;
  }

//группа для фантома должна быть временная и обновляться при изменении вида
// отрисовки
 if (rub->type1.gr)
    DeleteObj(rub->type1.gr);
  rub->type1.gr = NewGroup(1); // временная группа
    if((flag==1 && com==1)||(flag==2 && com==2))
    type = 3;
```

```
//обновляется не только изображение но и меню для запроса
 switch (type ) {
     case 1:
       Circle (0, 0, 20, 1);
       info->commands =  !Квадрат !Треугольник ;
       flag = 1;
       break;
     case 2:
       LineSeg(-10, 0, 10, 0, 1);
       LineSeg(10, 0, 0, 20, 1);
       LineSeg(0, 20,  -10, 0, 1);
       info->commands =  !Окружность !Квадрат;
       flag = 2;
       break;
     case 3:
       LineSeg(-10, 0, 10, 0, 1);
       LineSeg(10, 0, 10, 20, 1);
       LineSeg(10, 20,  -10, 20, 1);
       LineSeg(-10, 20,  -10, 0, 1);
       info->commands =  !Окружность !Треугольник;
       flag = 0;
       break;
   }
  EndGroup();

  return 1;
}

void Placement_Example (void) {
  type = 1;
  int j = 1;
  struct Phantom  rub;
  rub.type1.xBase = 0;
  rub.type1.yBase = 0;

  rub.type1.scale = 1;
  rub.phType       = 1;
```

```
  rub.type1.ang  = 0;
  double x, y;
  rub.type1.gr = NewGroup(1); // временная группа
    Circle (0, 0, 20, 1);
  EndGroup();
RequestInfo  info;
  memset(&info, 0, sizeof(info));
  info.commands = "!Квадрат !Треугольник";
  info.callBack = CallBackP;//указываем адрес функции обратной связи для Placement
  Placement(&info, &x, &y, &rub.type1.ang, &rub);

}; /* Placement_Example*/
```

ksGetCursorPosition - Пример использования

```
//получить координаты курсора
 typeCursorPos = YesNo("С учетом привязок?") == 1 ? 1 : 0;
 int j ;
 double x, y;
 do {
   ksGetCursorPosition(&x, &y, typeCursorPos);   // координаты курсора в миллиметрах,
                                  // 0 - без учета привязок 1 - с учетом привязок,

   char buf[255];
   sprintf(buf, " x = %0.2f y = %0.2f; Продолжать? ", x, y);
   j = YesNo(buf);
 } while(j == 1);
```

ksGetCursorLimit - Пример использования

```
double x, y;                          // координаты точки
RequestInfo info;                     // параметры запроса
memset(&info, 0, sizeof(info));       // очищаем параметры запроса
info.prompt = "Укажите объект";       // строка подсказки
double lim = ::ksGetCursorLimit();    // радиус ловушки курсора
while (::Cursor(&info, &x ,&y, 0)) {  // запуск процесса указания точки
reference obj = ::FindObj(x, y, lim);  // ближайщий объект
```

```
if (::ExistObj(obj))                  // если объект найден и существует
::LightObj(obj, 1);                   // подсвечиваем объект
}
```

GetValidator - Пример использования

```
// Описание класса обработки диалога, в котором использованы два TEdit,
// чтобы ввести проверку интервала вводимых значений.
// К TEdit подключается валидатор

#define DIALOG_1101
#define IDC_EDIT11001
#define IDC_EDIT21002

//-----------------------------------
// ресурс диалога
DIALOG_1 DIALOG 114, 71, 103, 107
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION | WS_SYSMENU
CAPTION "Диалог"
FONT 8, "MS Sans Serif"
{
DEFPUSHBUTTON "OK", IDOK, 27, 80, 50, 14
EDITTEXT IDC_EDIT1, 63, 24, 21, 12
LTEXT "Исполнение", -1, 15, 27, 43, 8
EDITTEXT IDC_EDIT2, 63, 48, 32, 12
LTEXT "Длина", -1, 17, 50, 29, 8
}

//-----------------------------------
class TestDialog : public TDialog
{
float len;
int ispoln;
TEdit *hLen;    //ввод длины: значение кладется в len
TEdit *hIspoln; //ввод исполнения: значение кладется в ispoln
public:
TestDialog (TWindow *ptParent);
void SetupWindow();
```

```
protected :
void EvLen();
void EvIspoln();
void CmOk(); // IDOK
DECLARE_RESPONSE_TABLE(TestDialog);
};

DEFINE_RESPONSE_TABLE1(TestDialog , TDialog)
EV_EN_KILLFOCUS  (IDC_EDIT2,  EvLen),
EV_EN_KILLFOCUS  (IDC_EDIT1, EvIspoln),
EV_COMMAND(IDOK, CmOk),
END_RESPONSE_TABLE;

//конструктор диалога
//-----------------------------------
TestDialog::TestDialog(TWindow *ptParent):
TDialog(ptParent , DIALOG_1, module),
len(20),
ispoln (1)
{
hLen     = new TEdit(this, IDC_EDIT2, 10);
//интервал для float: тип 4;   min=0 max=100
float minF=0, maxF=100;
//валидатор контролирует значения типа float
hLen->SetValidator ((TValidator *) GetValidator(&minF, &maxF, 4));

hIspoln  = new TEdit (this, IDC_EDIT1, 10);
//интервал для int: тип 2;   min=1 max=5
//валидатор контролирует значения типа int
int minI =1, maxI=5;
hIspoln->SetValidator ((TValidator *) GetValidator (&minI, &maxI, 2));

};

//-----------------------------------
void TestDialog::SetupWindow()
{
char s[128];
```

```
TDialog::SetupWindow();
sprintf (s, "%.1f", len);
hLen->SetText(s);
sprintf (s, "%d", ispoln);
hIspoln->SetText(s);
}

//-----------------------------------
void TestDialog::EvLen()
{
if (hLen->IsValid(FALSE))//мы в интервале
{
char buf[20];
char *c;
GetDlgItemText (IDC_EDIT2, buf, 10);
len = strtod (buf, &c);
}
}

//-----------------------------------
void TestDialog::EvIspoln()
{
if (hIspoln->IsValid(FALSE))//мы в интервале
{
char buf[20];
char *c;
GetDlgItemText (IDC_EDIT1, buf, 10);
ispoln = (int) strtol (buf, &c, 10);
}
}

//-----------------------------------
void TestDialog::CmOk()
{
if (!hLen->IsValid(TRUE))
{
hLen->SetFocus();
return;
```

```
}
if (!hIspoln->IsValid(TRUE))
{
hIspoln->SetFocus();
return;
}
TDialog::CmOk();
char buf[128];
sprintf (buf, "len=%.3f ispoln=%d",len,ispoln);
Message (buf);
}

//---------------------------------------------------------------------------
// Функция вызова диалога с использованием валидаторов
//---------------------------------------------------------------------------
void TestValidator()
{
TestDialog *pDialog;
pDialog = new TestDialog (GetWindowPtr((HWND)GetHWindow()));
pDialog->Execute();
delete pDialog;
}
```

ksPhantomShowHide - Пример использования

```
{
#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

int type, flag=0;

//функция обратной связи для выполнения цикла в Cursor
int far __export pascal CallBackC (int com, double *x, double *y, RequestInfo *info,
                                   void *phantom, int/* dynamic */) {
 Phantom *valueub = (Phantom *)phantom;

switch (com) {
```

```
    case 1:
    case 2:
      type = com;
    break;
    case -1: //поставить в модель
      MoveObj(rub->type1.gr, *x, *y);

      ksPhantomShowHide (0);

      StoreTmpGroup(rub->type1.gr); //поставить временную группу в вид
      ClearGroup(rub->type1.gr);
      break;
  }

//группа для фантома должна быть временная и обновляться при изменении вида
//отрисовки
  if (rub->type1.gr)
    DeleteObj(rub->type1.gr);
  rub->type1.gr = NewGroup(1); // временная группа
    if((flag==1 && com==1)||(flag==2 && com==2))
    type = 3;
//обновляется не только изображение но и меню для запроса
    switch (type ) {
      case 1:
        Circle (0, 0, 20, 1);
        info->commands =  !Квадрат !Треугольник ;
        flag = 1;
        break;
      case 2:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 0, 20, 1);
        LineSeg(0, 20,  -10, 0, 1);
        info->commands =  !Окружность !Квадрат;
        flag = 2;
        break;
      case 3:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 10, 20, 1);
```

```
        LineSeg(10, 20,  -10, 20, 1);
        LineSeg(-10, 20,  -10, 0, 1);
        info->commands =  !Окружность !Треугольник;
        flag = 0;
        break;
     }
  EndGroup();

  return 1;
}

void Cursor_Example (void) {
  type = 1;
  int j = 1;
  struct Phantom  rub;
  rub.type1.xBase = 0;
  rub.type1.yBase = 0;

  rub.type1.scale = 1;
  rub.phType       = 1;

  rub.type1.ang  = 0;
  double x, y;
  rub.type1.gr = NewGroup(1); // временная группа
    Circle (0, 0, 20, 1);
  EndGroup();
  RequestInfo  info;
  memset(&info, 0, sizeof(info));
  info.commands = "!Квадрат !Треугольник";
  info.callBack = CallBackC;//указываем адрес  функции обратной связи для Cursor   ,
Cursor(&info , &x, &y, &rub);

}; /* Cursor*/
}
```

```
{
#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

int type, flag=0;

//функция обратной связи для выполнения цикла в Cursor
int far __export pascal CallBackC (int com, double *x, double *y, RequestInfo *info,
                                   void *phantom, int/* dynamic */) {
 Phantom *valueub = (Phantom *)phantom;

rub->type1.ang  = 90;
ksChangeObjectInLibRequest (&info, &rub);

switch (com) {
   case 1:
   case 2:
     type = com;
   break;
   case -1: //поставить в модель
if (ksIsCursorOrPlacementDocument () == 1)
{
     MoveObj(rub->type1.gr, *x, *y);
     StoreTmpGroup(rub->type1.gr); //поставить временную группу в вид
     ClearGroup(rub->type1.gr);
}
     break;
  }

//группа для фантома должна быть временная и обновляться при изменении вида
//отрисовки
  if (rub->type1.gr)
    DeleteObj(rub->type1.gr);
  rub->type1.gr = NewGroup(1); // временная группа
```

```
    if((flag==1 && com==1)||(flag==2 && com==2))
    type = 3;
//обновляется не только изображение но и меню для запроса
    switch (type ) {
      case 1:
        Circle (0, 0, 20, 1);
        info->commands =  !Квадрат !Треугольник ;
        flag = 1;
        break;
      case 2:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 0, 20, 1);
        LineSeg(0, 20,  -10, 0, 1);
        info->commands =  !Окружность !Квадрат;
        flag = 2;
        break;
      case 3:
        LineSeg(-10, 0, 10, 0, 1);
        LineSeg(10, 0, 10, 20, 1);
        LineSeg(10, 20,  -10, 20, 1);
        LineSeg(-10, 20,  -10, 0, 1);
        info->commands =  !Окружность !Треугольник;
        flag = 0;
        break;
    }
  EndGroup();

  return 1;
}

void Cursor_Example (void) {
  type = 1;
  int j = 1;
  struct Phantom  rub;
  rub.type1.xBase = 0;
  rub.type1.yBase = 0;

  rub.type1.scale = 1;
```

```
  rub.phType       = 1;

  rub.type1.ang  = 0;
  double x, y;
  rub.type1.gr = NewGroup(1); // временная группа
    Circle (0, 0, 20, 1);
  EndGroup();
  RequestInfo  info;
  memset(&info, 0, sizeof(info));
  info.commands = "!Квадрат !Треугольник";
  info.callBack = CallBackC;//указываем адрес функции обратной связи для Cursor,
Cursor(&info , &x, &y, &rub);

}; /* Cursor*/
}
```

Пример использования функций ввода параметров

```
void MessageEsc(void) {
Message(Отказ от ввода);
}

void Read_Example (void) {
int I;  long l;  double r;
char s[80], buf[80];

if (ReadInt(Ввод целого числа, 5, 0, 999, &i)) {
sprintf(buf, Результат= %4d, i);
Message(buf);
}
else MessageEsc();

if (ReadLong(Ввод длинного целого , 77, 0, 999, &l)) {
sprintf(buf, Результат= %4d, l);
Message(buf);
}
else MessageEsc();
```

```
if (ReadDouble(Ввод действительного числа, 5, 0, 999, &r)) {
sprintf(buf, Результат= %5.1f, r);
Message(buf);
}
else MessageEsc();

if (ReadString(Ввод строки, s))
Message(s);
else MessageEsc();

}; /* Read_Example */
```

Пример использования функций выдачи сообщений

```
void Message_Example ( void ) {

if ( YesNo( «Выполнять действие» )
Message( «Пришло подтверждение» );
else Message( «Отказ» );

Pause( «Нажмите любую клавишу» );
Error( «Сообщение об ошибке» );

}; /* Message_Example */
```

ReturnResult, StrResult, MessageBoxResult - пример использования

```
void ReturnResult _Example ( void ) {
char s[80];

LineSeg ( 0, 0, 0, 0, 1 ); /* совпадающие узлы отрезка */
if ( ReturnResult ( ) ) {
/* выдать сообщение в специальном окне */
MessageBoxResult( );
};

Circle ( 10, 10, 0, 1 ); /* нулевой радиус окружности */
if ( ReturnResult ( ) ) {
```

```
StrResult( s, 80 );
Error( s );
};

}; /* ReturnResult */
```

DrawBitmap, DrawSlide - пример использования

```
#ifndef __OWL_WINDOW_H
#include <owl\window.h>
#endif

void DrawBitmap_Example ( void ) {

// Пример файла ресурсов
//
//Описание слайда

100 RCDATA {
 SC, 0,
 LS, 2, 1,
 LN, 12, 59, 179, 59,
 END_SLIDE
}
200 BITMAP 1.bmp

extern TModule *module ;

TWindow *st = new TWindow ( GetWindowPtr( ( HWND )GetHWindow( ) ), 0 );
st->Attr.X =300;
st->Attr.Y =30;
st->Attr.W =170;
st->Attr.H =160;

st->Create( );  //создаем окно для отрисовки слайда
DrawBitmap ( ( void* ) st->HWindow, 100 );
DrawSlide ( ( void* ) st->HWindow, 200 );
```

```
Message( Вывод слайда );
delete st;

}; /* DrawBitmap_Example */
```

WriteSlide - пример использования

```
#ifndef __OWL_WINDOW_H
#include <owl\window.h>
#endif

void WriteSlide_Example ( void ) {

char name[128];
double x, y;
int slideID;

 // ввод имени файла для записи слайда

 if ( SaveFile(*.rc, name ) ){
   if ( Cursor( Укажите точку привязки слайда,&x,&y,NULL,NULL ) ) {
    if ( ReadInt( Введите идентификатор слайда, 100, 0, 32000, &slideID ) ) {
     if ( !WriteSlide ( name, slideID, x, y ) )
      Error( Группа селектирования пуста );
     ClearGroup( 0 );
    }
   }
 }

}; /* DrawBitmap_Example */
```

ksSlideBackground - пример использования

```
{
ksSlideBackground ( 15 );
}
```

ksDrawSlideFromFile - Пример использования

```
//фрагмент из файла .rc

1000 DIALOG 84, 66, 204, 133
EXSTYLE WS_EX_DLGMODALFRAME
STYLE DS_MODALFRAME | WS_POPUP | WS_VISIBLE | WS_CAPTION | WS_SYSMENU
CAPTION ""
FONT 8, "MS Sans Serif"
{
 CONTROL "OK", IDOK, "BUTTON", BS_DEFPUSHBUTTON | WS_CHILD | WS_VISIBLE |
WS_TABSTOP, 8, 111, 50, 14
 CONTROL "Cancel", IDCANCEL, "BUTTON", BS_PUSHBUTTON | WS_CHILD |
WS_VISIBLE | WS_TABSTOP, 72, 111, 50, 14
 CONTROL "", IDD_SHOW, "static", SS_BLACKFRAME | WS_CHILD | WS_VISIBLE |
WS_BORDER, 7, 3, 105, 97
}


class TShowStatic;


//Описание класса, обслуживающего диалог
class TModDialog :public TDialog
{
public:
 UINT IdSlide;
TShowStatic* Show;
TModDialog ( TWindow *ptParent );
 UINT GetIdSlide( ) { return IdSlide; };
};

//Описание класса обслуживающего элемент вывода слайда
class TShowStatic : public TStatic {
public:
 TShowStatic ( TModDialog * ptParent, int resourceId );
protected :
void       EvPaint( );
```

```
DECLARE_RESPONSE_TABLE( TShowStatic );
};

DEFINE_RESPONSE_TABLE1( TShowStatic,TStatic )
EV_WM_PAINT,
END_RESPONSE_TABLE;
//----------------------------------------------------------------------------
-----
//
//----------------------------------------------------------------------------
-----
TShowStatic :: TShowStatic ( TModDialog *ptParent, int resourceId ):
 TStatic( ptParent, resourceId ) {
SetBkgndColor( GetSysColor( COLOR_APPWORKSPACE ) );
}

//----------------------------------------------------------------------------
-----
//
//----------------------------------------------------------------------------
-----
void TShowStatic ::EvPaint( )
{
 TStatic::EvPaint( );
 TModDialog * d = TYPESAFE_DOWNCAST( Parent, TModDialog );
 if ( d )
//  DrawSlide ( ( void * ) HWindow, d->GetIdSlide( ) );
//  DrawBitmap ( ( void * ) HWindow, d->GetIdSlide( ) );
//в файле "c:\\libtest\\1.rc" - находится отлаживаемый слайд
  ksDrawSlideFromFile ( ( void * ) HWindow, "c:\\libtest\\1.rc" );

}
//----------------------------------------------------------------------------
-----
//
//----------------------------------------------------------------------------
-----
TModDialog :: TModDialog ( TWindow *ptParent )
```

```
 : TDialog( ptParent, 1000, module ){
 IdSlide = 100;
Show=new TShowStatic( this,IDD_SHOW );
}

//---------------------------------------------------------------------------
-----
// Функция, вызывающая диалог с отрисовкой слайда или битмапа
//---------------------------------------------------------------------------
-----
void TestShowDialog( ) {
TModDialog *pDialog;
pDialog = new TModDialog( GetWindowPtr( ( HWND )GetHWindow( ) ) );
pDialog->Execute( );
delete pDialog;
}

//файл 1.rc
100 RCDATA {
 GB, 168, 158,
 SC, 1,
 LS, 0, 1,
 LN, 69, 57, 69, 26,
 LN, 69, 26, 74, 18,
 LN, 74, 18, 101, 18,
 LN, 101, 18, 105, 26,
 LN, 105, 26, 105, 57,
 LN, 74, 37, 101, 37,
 AR1, 114, 57, 45, 74, 37, 69, 57,
 AR1, 61, 57, 44, 105, 57, 101, 37,
 AR1, 80, 28, 11, 69, 28, 74, 37,
 AR1, 94, 28, 11, 101, 37, 105, 28,
 LN, 69, 57, 69, 88,
 LN, 69, 88, 74, 96,
 LN, 74, 96, 101, 96,
 LN, 101, 96, 105, 88,
 LN, 105, 88, 105, 57,
 LN, 74, 77, 101, 77,
```

```
 AR1, 114, 57, 45, 69, 57, 74, 77,
 AR1, 62, 57, 44, 103, 77, 105, 57,
 AR1, 80, 86, 11, 74, 77, 69, 86,
 AR1, 95, 86, 11, 106, 86, 100, 76,
 SC, 0,
 LS, 2, 1,
 LN, 62, 57, 112, 57,
 SC, 0,
 TS, 0,
 MA, 45, 112,
 TX, "ГОСТ 15524-70\0"
 MA, 37, 127,
 TX, "Класс точности А\0"
 MA, 37, 142,
 TX, " гайки высокие\0"

 END_SLIDE


}
```

ksDrawKompasGroup - пример использования

```
//текущим документом должен быть графический документ
 if ( ksGetCurrentDocument( 1 ) ) {
  TWindow *st = new TWindow ( GetWindowPtr( ( HWND )GetHWindow( ) ), 0 );
  st->Attr.X =300;
  st->Attr.Y =30;
  st->Attr.W =170;
  st->Attr.H =160;

  st->Create( );  //создали окно, в котором хотим отрисовать группу
  //создаем группу с изображением прямоугольника
  reference gr = NewGroup( 1 );
   Mtr( 20, 15, 45, 1 );
   LineSeg( -10, 0, 10, 0, 1 );
   LineSeg( 10, 0, 10, 20, 1 );
   LineSeg( 10, 20, -10, 20, 1 );
```

```
 LineSeg( -10, 20, -10, 0, 1 );
 DeleteMtr( );
EndGroup( );
//отрисовываем группу в окне st
ksDrawKompasGroup ( ( void * )st->HWindow,       // несущее окно
            gr );       //группа
//для красоты рисуем рамочку по периметру окна
TClientDC dc(*st );
HPEN hPen = ( HPEN ) ::SelectObject( dc, ::CreatePen( PS_SOLID,2,RGB( 0,0,255 ) ) );
dc.MoveTo( 0, 0 );
dc.LineTo( 170, 0 );
dc.LineTo( 170, 160 );
dc.LineTo( 0, 160 );
dc.LineTo( 0, 0 );
::DeleteObject( ::SelectObject( dc, hPen ) );

Message( "смотри" );
delete st;
}
else
Error ( "Графический документ не активен" );
```

ksInitFilePreviewFunc - пример использования

```
//---------------------------------------------------------------------------
// функция обратной связи для отображения в окне предварительного просмотра
// файлов не документов КОМПАС
// ---
int far __export pascal MyPreviewFunc( HWND HWindow, // дескриптор окна просмотра
 char * fileName ){ // файл, который нужно показать в окне просмотра
 if ( fileName ) {
  char *c = strrchr( fileName, '.' );
  if ( c && strcmp( c, ".cdw" ) && strcmp( c, ".frw" ) && strcmp( c, ".spw" ) && strcmp( c, ".kdw"
) ) {
   //если не документ КОМПАС отобразим имя файла
   c = strrchr( fileName, '\\' );
   if ( c ) {
```

```
    TRect rect;
    ::GetClientRect( HWindow, &rect );

    TWindow *staticW = new TWindow( HWindow );

    TDC *dc = new TClientDC(*staticW );
    dc->TextOut( 40, ( rect.bottom - rect.top )*0.5-10, c1, strlen( c1 ) );

    delete dc;
    delete staticW;
    return 1;
   }
  }
 }
 return 0;
}

//------------------------------------------------------------------------------
//
// ---
extern "C" void far __export __pascal LibraryEntry( unsigned int com ){
 switch ( com ) {
  case 1: {
   //установим адрес нашей функции просмотра
   ksInitFilePreviewFunc( MyPreviewFunc );
   char fileName[250];
   //выберем файл
   if( ( ksChoiceFile( 0,"Все файлы (*.* )|*.*|", fileName, 250, 1 ) ) != 0 )
    Message( fileName );
   // обнулим адрес
   ksInitFilePreviewFunc( 0 );
   // выберем файл
   if( ( ksChoiceFile( 0,"Все файлы (*.* )|*.*|", fileName, 250, 1 ) ) != 0 )
    Message( fileName );
   break;
  }
 }
}
```

GetHWindow - пример использования

```
void GetHWindow_Example ( void ) {

/* выдача сообщения в главном окне */

MessageBox( GetHWindow( ), Текст заголовка окна,
Текст сообщения,
MB_ICONINFORMATION );

}; /* GetHWindow_Example */
```

ksDrawKompasDocument - пример использования

```
TWindow *st = new TWindow (GetWindowPtr( ( HWND )GetHWindow( ) ), 0 );
 st->Attr.X =300;
 st->Attr.Y =30;
 st->Attr.W =170;
 st->Attr.H =160;

 st->Create( );  //создали окно, в котором хотим отрисовать слайд

 ksDrawKompasDocument( ( void* ) st->HWindow,   // несущее окно
          "d:\\0\\fg1.frw" ); // полное имя файла документа

 Message( "Слайд КОМПАС" );
 delete st;
```

База данных с доступом через интерфейс ODBC

ODBC - Open Database Connectivity представляет собой открытый интерфейс для подключения к базам данных различных форматов, например, MS Excel, Access.

Чтобы обеспечить связь с базой данных, необходимо подключить драйвер базы данных нужного типа, используя Администратор источников данных ODBC.

База данных текстового формата

База данных (БД) текстового формата представляет собой текстовый файл, содержащий таблицу значений.

Строки таблицы являются записями БД. В качестве разделителей значений в строке можно использовать пробел или , (запятую). Данные в столбцах должны иметь одинаковый тип.

В системе КОМПАС такие базы данных могут быть использованы, например, для хранения значений свойств параметризованных объектов. По умолчанию файлам БД тестового формата системы КОМПАС присваивается расширение loa.

Синтаксис БД текстового формата.

Файл БД содержит комментарии и таблицу значений. Комментариями являются следующие элементы текста:

- фрагмент, заключенный между символами /* и */.

- строки, начинающиеся с //.

Остальные строки образуют таблицу текстовой базы данных, например:

```
/* a диаметр оси
   b длина оси
   c ширина проточки
*/
// a   b  c
  25  55  1.5
  30  55  2
  45  80  2.5
```

Пример использования функций работы с базами данных

```
//пример текстового файла 1.loa с тремя параметрами - шириной, высотой и типом линии прямоугольника
/*
   10.0          100          1
   15.0           80          2
   20.0          120          1
*/

#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

#ifndef __LIBDB_H
#include <libdb.h>
#endif
```

```
static void ReadFirstStatement(reference bd, reference r1, void *b) {
// вспомогательная функция чтения записей

  char buf[40];
  int i;
  do { // считываем все записи выборки
    i= ReadRecord(bd, r1, b);
    if (i) {
      sprintf(buf, "a=%4.1f b=%4.1f t= %d", b->a, b->b, b->t);
      Message(buf);
    }
  } while(i);
  Message("конец выборки");
} /* ReadFirstStatement */

void ExampleDB (void) {
  int i;
  char buf[40];
  struct {
   double a;
   double b;
   int    t;
  } b;

// создать блок заголовка базы данных
  reference bd = CreateDB ("TXT_DB");
// связать с файлом  базы данных 1.loa
  ConnectDB(bd, "1.loa");

  // задаем отношение из двух действительных и одного целого поля
  reference r1 = Relation(bd);
   RDouble ("a");
   RDouble ("b");
   RInt    ("");
  EndRelation();

  // определяем запрос для считывания всех полей(колонок) записи
```

```
  DoStatement(bd, r1, ""); // равносильно DoStatement(bd, r1, "1 2 3");

  // считываем все записи
  ReadFirstStatement(bd, r1, &b);

  // выдать вторую запись
  Condition(bd, r1, "Index1000 = 2");
  ReadFirstStatement(bd, r1, &b);

  // по условию выдать первую и третью запись(см. файл 1.loa)
  Condition(bd, r1, "a=10 || b>100");
  ReadFirstStatement(bd, r1, &b);

  //задаем  отношение для считывания только второго поля записи
  reference r2 = Relation(bd);
    RChar("b", 20, 0);
  EndRelation();

// Освободить запрос базы данных
FreeStatement(bd,  r1, «»);

  DoStatement(bd, r2, "2");
  do {
    i= ReadRecord(bd, r2, buf);
    if (i) {
      Message(buf);
    }
  } while(i);
  Message("конец выборки");

  DisconnectDB(db);
  // удаляем блок заголовка базы данных
  DeleteDB(bd);
}; /* ExampleDB */
```

GetTableName, GetColumnName - пример использования

```
char buf[128],nameOBDC[128];//имя источника данных
```

```
//создать объект, обслуживающий базу данных
reference bd = CreateDB ("ODBC_DB");
if (ConnectDB (bd, nameOBDC)) {
if (GetTableName (bd, buf, 128, 'F')) {
do {
Message(buf);
if (GetColumnName (bd, buf, buf, 128, 'F')) {
do {
Message(buf);
} while (GetColumnName (bd, buf, buf, 128, 'N'));
}
} while (GetTableName (bd, buf, 128, 'N')) ;
}
}
```

OpenTextFile, CloseTextFile, ReadStrFromTextFile- пример использования

```
////////////////////////////////////////////////////////////////////////
//пример текстового файла 1.loa
  10.0          100         1
  15.0           80         2
  20.0          120         1

////////////////////////////////////////////////////////////////////////
//пример текстового файла запросов 11.txt  привязанный к файлу 1.loa
1:TXT_DB
2:1.loa
3:1 2 3
4:a=%.1f
////////////////////////////////////////////////////////////////////////
//пример текстового файла  запросов 12.txt  привязанный к БД ACCECC с именем
// test в ODBC.INI. ODBCSDK должен быть установлен на ПК
1:ODBC_DB
2:test
3:select a,b,t from table1
4:where a=%.1f
////////////////////////////////////////////////////////////////////////
```

```
#ifndef __LIBTOOL_H
#include <libtool.h>
#endif

#ifndef __LIBDB_H
#include <libdb.h>
#endif

void DrawRect (void) {
  char buf[40], s[128];
  struct {
   double a;
   double b;

   int t;
  } r;

  reference  f = OpenTextFile("11.txt");

  if (EditMacroMode())   //режим редактирования
    GetMacroParam(&r, sizeof(r));
  else {  //создание нового макроэлемента
    r.a = 20;
    r.b = 10;
  }
  // считали тип БД  - "TXT_DB"
  if (!ReadStrFromTextFile(f,buf,1)) {Error("Ошибка"); return; }
  reference bd = CreateDB (buf);
  // считали имя БД  - "1.loa"
  if (!ReadStrFromTextFile(f,buf,2)) {Error("Ошибка"); return; }
  ConnectDB(bd, buf);

  //задаем отношение
  reference r1 = Relation(bd);
   RDouble ("a");
   RDouble ("b");
   RInt    ("");
  EndRelation();
```

```
 //считывать все колонки "1 2 3"
 if (!ReadStrFromTextFile(f,buf,3)) {Error("Ошибка"); return; }
 DoStatement(bd, r1, buf);
 ReadDouble("задайте ширину",r.a,0,100, &r.a);
 //считать условие   "a=%.1f"
 if (!ReadStrFromTextFile(f,buf,4)) {Error("Ошибка"); return; }
 sprintf (s, buf,r.a); // условие например a=10.0
 Condition(bd, r1, s);
 if (ReadRecord(bd, r1, &r))  {
   reference m = Macro();
     LineSeg(0, 0, r.a, 0, r.t);
     LineSeg(r.a, 0, r.a, r.b, r.t);

     LineSeg(r.a, r.b, 0, r.b, r.t);
     LineSeg(0, r.b, 0, 0, r.t);
   EndObj();
  //записать параметры в макроэлемент
   SetMacroParam( m, &r, sizeof(r), NULL, NULL, -1);
 }
 else
   Error("прямоугольник с такими параметрами не найден в БД");
 DeleteDB(bd);
 CloseTextFile(f);

}; /* OpenTextFile */
```

Пример использования функций работы с динамическими массивами

```
struct User_Data// определение структуры пользовательского массива
{
int I;
char *c;
float f;
};

//функция удаляет элемент User_Data пользовательского массива
void WINAPI DelUserFunc (void *val)
```

```
{
User_Data  *tmp = (User_Data *)val;
Message(tmp->c);
delete [] tmp->c;
delete tmp;
}
//функция добавляет элемент User_Data в пользовательский массив
int CreateUserItem (reference arr, int i, char *c, float f)
{
User_Data* tmp = new User_Data;
tmp->i = I;
tmp->c = new char [strlen(c) +1];
strcpy (tmp->c, c);
tmp->f = f;
return  AddArrayItem (arr, -1, tmp, 0);
}

//функция выдает содержимое элемента User_Data
void ShowUserItem (User_Data & tmp)
{
char buf[128];
sprintf (buf, " i=%d f=%f\n c=%s ", tmp.i, tmp.f, tmp.c);
Message (buf);
}

void Array_Example (void)
{
User_Data *tmp;

//создадим пользовательский массив неопределенной длины
reference userArr = CreateArray (USER_ARR, DelUserFunc);

if (!CreateUserItem (userArr, 1, "январь", 31.))
MessageBoxResult();
if (!CreateUserItem (userArr, 2, "февраль", 29.))
MessageBoxResult();
if (!CreateUserItem (userArr, 3, "март", 31.))
MessageBoxResult();
```

```
//просмотрим массив
for (int i=0; i< GetArrayCount(userArr); i++)
{
// получим указатель на элемент массива
GetUserArrayItem (userArr, i, &(void*)tmp);
ShowUserItem (*tmp);
}
ClearArray (userArr);//очистить массив
Message ("удаляем массив");
DeleteArray (userArr);
};
```

Значения параметра pObj

NULL или указатель на интерфейс документа ksDocument3D или IDocument3D	- выполняется операция с атрибутом у документа,
Указатель на группу объектов дерева ksFeatureCollection или IFeatureCollection	- выполняется операция с групповым атрибутом,
Указатель на объект дерева ksFeature или IFeature	- выполняется операция с атрибутом определенного объекта,
Указатель на COM-интерфейс из диапазона: o3d_planeXOY...o3d_planeYOZ,o3d_sketch, o3d_axis2Planes...o3d_lastEntityElement-1,o3d_entity	- выполняется операция с атрибутом у соответствующего ему элемента дерева.

CreateAttrType - Пример использования

```
void CreateAttrType_Example (void)
{
//создадим тип атрибута - строка
AttributeType attrType;//структура типа атрибута
char nameFile[128];

reference pCol;//указатель на массив колонок
ColumnInfo parStruct;//структура для колонки

//создадим массив колонок типа атрибута
```

```
pCol = CreateArray (ATTR_COLUMN_ARR,0);

//опишем единственную колонку типа STRING_ATTR_TYPE
strcpy (parStruct.header, "строка");// заголовок-комментарий столбца
parStruct.type = STRING_ATTR_TYPE;// тип данных в столбце - см.ниже

// дополнительный признак, который позволит отличить две переменные с одинаковым типом
parStruct.key = 0;
strcpy(parStruct.def,"линия");// значение по умолчанию
parStruct.flagEnum =0;// режим заполнения
parStruct.fieldEnum = 0;// массив неопределенной длины перечислений (строки)
parStruct.columns = 0;// массив неопределенной длины информации
// о колонках для записи

AddArrayItem(pCol, -1, &parStruct, sizeof(parStruct)); //добавили колонку

//заполним структуру типа атрибута
strcpy(attrType.header,"тип строка");// заголовок-комментарий типа
attrType.rowsCount = 1;// кол-во строк в таблице
attrType.flagVisible = 1;// видимый/невидимый
strcpy(attrType.password,"");// пароль
attrType.columns = pCol;// массив неопределенной длины информации о колонках

//запросить имя библиотеки
if(!ChoiceFile("lta", NULL, nameFile, 128))
nameFile[0]='\0'; //будем создавать в документе

//создать тип атрибута
double numbType = CreateAttrType (&attrType,// информация о типе атрибута
nameFile);// имя библиотеки типов атрибутов
if (numbType > 1)
{
char buf[128];
sprintf(buf, "numbType=%f ",numbType);
Message(buf);
}
else MessageBoxResult();// ошибка
```

```
//удалим массив колонок
DeleteArray(pCol);
};
```

ksCreateAttrType - Пример использования

```
//---------------------------------------------------------------------------
// создадим тип атрибута болта
//---------------------------------------------------------------------------
void TypeAttrBolt()
{
//Инициализация данных для создания шаблона

//заполним структуру типа атрибута
ksAttributeType attrType;
strcpy(attrType.header,"Болт");// заголовок-комментарий типа
attrType.rowsCount = 1;// кол-во строк в таблице
attrType.flagVisible = 1;// видимый, невидимый  в таблице
strcpy (attrType.password,"");// пароль, если не пустая строка
//- защищает от несанкционированного изменения типа
attrType.columns = CreateArray (ATTR_COLUMN_ARR, 0);// динамический массив
//компонент записи
attrType.key1 = ST_KEY1;
attrType.key2 = ST_KEY2_GOST;
attrType.key3 = ST_KEY3;
attrType.key4 = 0;
ColumnInfo parStruct1;

//массив полей записи

// колонка 1 "Имя элем."
strcpy (parStruct1.header, "Имя элем.");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см.ниже
parStruct1.key = 1;// дополнительный признак, очередность сортировки
strcpy (parStruct1.def,"Болт");// значение по умолчанию
parStruct1.flagEnum =0;// флаг, включающий режим,
// когда значение поля атрибута
```

```
// будет заполнятся из массива
// перечисленных значений 1 и 0 отключен
parStruct1.fieldEnum = 0;// массив неопределенной длины
//перечислений (строки)
parStruct1.columns = 0;// массив неопределенной длины
// информации о колонках для записи

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 2 "Исполнение"
strcpy(parStruct1.header, "Исполнение");// заголовок-комментарий столбца
parStruct1.type = UINT_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 3;// дополнительный признак, очередность сортировки
strcpy (parStruct1.def,"1");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 3 "Резьба"
strcpy (parStruct1.header, "Резьба");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;//  очередность сортировки
strcpy (parStruct1.def,"M");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 4 "Диаметр"
strcpy (parStruct1.header, "Диаметр");// заголовок-комментарий столбца
parStruct1.type = UINT_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 3;//  очередность сортировки
strcpy (parStruct1.def,"12");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 5 "разделитель"
```

```
strcpy (parStruct1.header, ""); // заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,"x");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 6 "Шаг"
strcpy (parStruct1.header, "Шаг");// заголовок-комментарий столбца
parStruct1.type = FLOAT_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 3;// очередность сортировки
strcpy(parStruct1.def,"1.25");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 7 "Поле допуска"
strcpy (parStruct1.header, "Поле допуска");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 3;// очередность сортировки
strcpy (parStruct1.def,"-6g");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 8 "разделитель"
strcpy (parStruct1.header, "");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,"x");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 9 "Длина"
strcpy (parStruct1.header, "Длина");// заголовок-комментарий столбца
parStruct1.type = UINT_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
```

```
parStruct1.key = 3;// очередность сортировки
strcpy(parStruct1.def,"60");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 10 "Кл. прочности"
strcpy (parStruct1.header, "Кл. прочности");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,".58");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 11 "Материал"
strcpy (parStruct1.header, "Материал");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,".35X");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 12 "Покрытие"
strcpy (parStruct1.header, "Покрытие");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,".16");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 13 "ГОСТ"
strcpy (parStruct1.header, "ГОСТ");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy(parStruct1.def,"ГОСТ");// значение по умолчанию
```

```
//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 14 "Номер"
strcpy (parStruct1.header, "Номер");// заголовок-комментарий столбца
parStruct1.type = UINT_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 2;// очередность сортировки
strcpy (parStruct1.def,"7808");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 15 "разделитель"
strcpy (parStruct1.header, "");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,"-");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof (parStruct1));

// колонка 16 "Год"
strcpy (parStruct1.header, "Год");// заголовок-комментарий столбца
parStruct1.type = STRING_ATTR_TYPE;// тип данных в столбце - см. "ltdefine.h"
parStruct1.key = 0;// очередность сортировки
strcpy (parStruct1.def,"70");// значение по умолчанию

//добавим компоненту в массив компонент
AddArrayItem (attrType.columns, -1, &parStruct1, sizeof(parStruct1));

// создать шаблон обозначения

ksCreateAttrType (&attrType,// информация о типе атрибута
ATTR_TYPE_LIB_NAME_KEY3);// имя библиотеки типов атрибутов

MessageBoxResult();// проверяем результат работы нашей функции
```

```
//удалим массив колонок
DeleteArray (attrType.columns);
}
```

DeleteAttrType - Пример использования

```
void DeleteAttrType_Example (void)
{
double numb;
int j;
char nameFile[128];
char password[11];

//запросить имя библиотеки
if (!ChoiceFile("tip", NULL, nameFile, 128))
nameFile[0]='\0'; //будем искать в документе
do
{
j = ReadDouble (" номер типа атрибута", 1000.,0, 1e12,&numb);
if (j)
{
j = ReadString (" пароль типа атрибута", password, 10);
if (j)
{
if (!DeleteAttrType (numb, nameFile, password))
MessageBoxResult();// проверяем результат работы функции
}
}
}
while (j);
};
```

GetAttrType - Пример использования

```
{

//создадим тип атрибута - строка
```

```
 AttributeType attrType, NewAttrType;  //структура типа атрибута
 char nameFile[128];

 reference pCol;      //указатель на массив колонок
 ColumnInfo parStruct; //структура для колонки

 //создали массив колонок типа атрибута
 pCol = CreateArray(ATTR_COLUMN_ARR,0);

  //опишем единственную колонку типа STRING_ATTR_TYPE
  strcpy(parStruct.header, "строка");  // заголовок-комментарий столбца
  parStruct.type = STRING_ATTR_TYPE; // тип данных в столбце - см.ниже

// дополнительный признак, который позволит отличить две переменные с одинаковым типом
  parStruct.key = 0;
  strcpy(parStruct.def,"линия"); // значение по умолчанию
  parStruct.flagEnum =0;   // режим заполнения
  parStruct.fieldEnum = 0;  // массив неопределенной длины перечислений (строки)
   parStruct.columns = 0;    / массив неопределенной длины информации о колонках для записи

  AddArrayItem(pCol, -1, &parStruct, sizeof(parStruct)); //добавили колонку

//заполним структуру типа атрибута
 strcpy(attrType.header,"тип строка");  // заголовок-комментарий типа
 attrType.rowsCount    = 1;  // кол-во строк в таблице
 attrType.flagVisible   = 1;   // видимый/невидимый
 strcpy(attrType.password,"");  // пароль
 attrType.columns     = pCol; // массив неопределенной длины информации о колонках

//запросить имя библиотеки
 if(!ChoiceFile("lta", NULL, nameFile, 128))
  nameFile[0]='\0'; //будем создавать в документе

 //создать тип атрибута
 double numbType = CreateAttrType(&attrType,  // информация о типе атрибута
                 nameFile); // имя библиотеки типов атрибутов
```

```
 if (numbType > 1) {
  char buf[128];
  sprintf(buf, "numbType=%f ",numbType);
  Message(buf);
 }
 else MessageBoxResult(); // ошибка

 GetAttrType(numbType, nameFile, &NewAttrType);

 //удалим массив колонок
 DeleteArray(pCol);
}
```

ksGetAttrType, ChoiceAttrTypes - Пример использования

```
static void ShowColumns (reference pCol, BOOL fl);
void ShowCol (ColumnInfo &par,int iCol, BOOL fl);

//--------------------------------------------------------------------------
// показать колонку
//--------------------------------------------------------------------------
void ShowCol (ColumnInfo &par,int iCol, BOOL fl)
{
char buf[128];
char s[10];
if (fl)
strcpy(s, "структура");
else
s[0]='\0';

//выдадим поля колонки не указатели
sprintf (buf,"%s i=%d header=%s type=%d def=%s flagEnum=%d",
s,iCol,par.header,
par.type, par.def, par.flagEnum);
Message(buf);
if (par.type == RECORD_ATTR_TYPE)//структура
ShowColumns(par.columns,TRUE);
```

```
else
{
if (par.flagEnum)//выдадим массив перечислений
{
int n1 = GetArrayCount (par.fieldEnum);
Message ("массив перечислений");
for (int i1=0; i1<n1; i1)
{
if (!GetArrayItem (par.fieldEnum , i1, &buf, sizeof(buf)))
MessageBoxResult();// проверяем результат работы
// нашей функции
else
Message(buf);
}
}
}
}

//---------------------------------------------------------------------------
// показать колонки
//---------------------------------------------------------------------------
static void ShowColumns (reference pCol, BOOL fl)
{
ColumnInfo par;
par.columns = CreateArray (ATTR_COLUMN_ARR,0);
par.fieldEnum = CreateArray (CHAR_STR_ARR,0);
int n = GetArrayCount (pCol);

for (int i = 0; i < n; i)
{
if (!GetArrayItem(pCol, i, &par, sizeof(par)))
MessageBoxResult();// проверяем результат работы нашей функции
else
ShowCol(par,i, fl);
}
DeleteArray (par.columns);
DeleteArray (par.fieldEnum);
}
```

```
//---------------------------------------------------------------------------
// получить тип атрибута
//---------------------------------------------------------------------------
void ShowTypeAttr ()
{
double numb;

char buf[128], nameFile[128];
ksAttributeType attrType;
attrType.columns = CreateArray (ATTR_COLUMN_ARR,0);

//запросить имя библиотеки
if (!ChoiceFile ("*.lat",NULL, nameFile,128))
nameFile[0]='\0';//будем искать в документе
do
{
numb = ChoiceAttrTypes (nameFile);
if (numb)
{
if (!ksGetAttrType (numb, nameFile, &attrType))
MessageBoxResult();// проверяем результат работы
// нашей функции
else
{
sprintf (buf,"key1 = %d key2 =%d key3 = %d key4 =&d",
attrType.key1,attrType.key2,attrType.key3,attrType.key4);
Message (buf);
sprintf (buf,"header=%s rowsCount=%d flagVisible=%d password=%s",
attrType.header,attrType.rowsCount,
attrType.flagVisible,attrType.password);
Message (buf);
ShowColumns (attrType.columns, FALSE);//пользовательская
// функция
}
}
}
while(numb);
```

```
//удалим массив колонок
DeleteArray(attrType.columns);
}
```

SetAttrType - Пример использования

```
{

//создадим тип атрибута - строка

 AttributeType attrType, NewAttrType;  //структура типа атрибута
 char nameFile[128];

 reference pCol;      //указатель на массив колонок
 ColumnInfo parStruct; //структура для колонки

 //создали массив колонок типа атрибута
 pCol = CreateArray(ATTR_COLUMN_ARR,0);

  //опишем единственную колонку типа STRING_ATTR_TYPE
  strcpy(parStruct.header, "строка");  // заголовок-комментарий столбца
  parStruct.type = STRING_ATTR_TYPE; // тип данных в столбце - см.ниже

// дополнительный признак, который позволит отличить две переменные с одинаковым типом
  parStruct.key = 0;
  strcpy(parStruct.def,"линия"); // значение по умолчанию
  parStruct.flagEnum =0;   // режим заполнения
  parStruct.fieldEnum = 0;  // массив неопределенной длины перечислений (строки)
   parStruct.columns = 0;    / массив неопределенной длины информации о колонках для записи

  AddArrayItem(pCol, -1, &parStruct, sizeof(parStruct)); //добавили колонку

//заполним структуру типа атрибута
 strcpy(attrType.header,"тип строка");  // заголовок-комментарий типа
 attrType.rowsCount    = 1;  // кол-во строк в таблице
```

```
 attrType.flagVisible   = 1;   // видимый/невидимый
 strcpy(attrType.password,"");  // пароль
 attrType.columns     = pCol; // массив неопределенной длины информации о колонках

//запросить имя библиотеки
 if(!ChoiceFile("lta", NULL, nameFile, 128))
  nameFile[0]='\0'; //будем создавать в документе

 //создать тип атрибута
 double numbType = CreateAttrType(&attrType,  // информация о типе атрибута
                nameFile); // имя библиотеки типов атрибутов
 if (numbType > 1) {
  char buf[128];
  sprintf(buf, "numbType=%f ",numbType);
  Message(buf);
 }
 else MessageBoxResult(); // ошибка

//заполним структуру нового типа атрибута
 strcpy(attrType.header,"тип строка");  // заголовок-комментарий типа
 attrType.rowsCount    = 1;  // кол-во строк в таблице
 attrType.flagVisible   = 1;   // видимый/невидимый
 strcpy(attrType.password,"pass");  // пароль
 attrType.columns     = pCol; // массив неопределенной длины информации о колонках

 SetAttrType (numbType, nameFile, &attrType, «pass»);

 //удалим массив колонок
 DeleteArray(pCol);
}
```

ksSetAttrType - Пример использования

```
double numb;
int j;
char nameFile[128];
char password[11];
```

```
ksAttributeType attrType;
attrType.columns = CreateArray (ATTR_COLUMN_ARR, 0);

//поменяем местами первую колонку и последнюю

ColumnInfo par1;//структура для первой колонки
par1.columns = CreateArray (ATTR_COLUMN_ARR,0);
par1.fieldEnum = CreateArray (CHAR_STR_ARR,0);

ColumnInfo parN;//структура для последней колонки
parN.columns = CreateArray (ATTR_COLUMN_ARR,0);
parN.fieldEnum = CreateArray (CHAR_STR_ARR,0);

//запросить имя библиотеки
if (!ChoiceFile("*.lat",NULL, nameFile,128))
nameFile[0]='\0'; //будем искать в документе
do
{
j = ReadDouble ("Ввести номер типа атрибута", 1000.,0, 1e12,&numb);
if (j)
{
j = ReadString ("Ввести пароль типа атрибута", password, 10);
if (j)
{
//считаем тип атрибута
if (!ksGetAttrType(numb, nameFile, &attrType))
MessageBoxResult();// проверяем результат работы
//нашей функции
else
{
strcpy (attrType.password,password);
// число колонок
int n = GetArrayCount (attrType.columns);
//считаем первую колонку
GetArrayItem (attrType.columns, 0, &par1, sizeof (par1));
//считаем последнюю колонку
GetArrayItem(attrType.columns, n-1, &parN, sizeof (parN));
//заменим первую колонку
```

```
SetArrayItem (attrType.columns, 0, &parN, sizeof (parN));
//заменим последнюю колонку
SetArrayItem (attrType.columns, n-1, &par1, sizeof (par1));

//заменим тип атрибута на новый
double numbType = ksSetAttrType (numb, nameFile,
&attrType, password);
if (numbType > 1)
{
char buf[128];
sprintf (buf, "numbType=%f ",numbType);
Message (buf);
}
else
MessageBoxResult();// неудачное завершение -
// выдадим результат работы нашей функции
}
}
}
}
while(j);
//удалим все созданные массивы
DeleteArray (par1.columns);
DeleteArray (par1.fieldEnum);

DeleteArray(parN.columns);
DeleteArray(parN.fieldEnum);

DeleteArray(attrType.columns);
```

CreateAttr - Пример использования

```
void CreateAttr_Example (void) {

//  создадим атрибут для типа атрибута с колонками
//  DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE, LINT_ATTR_TYPE
//  Предполагается, что данный тип уже существует
```

```
ksAttribute  attrPar;
//структура данных для оформления атрибута
struct {
double a;
char c[MAX_TEXT_LENGTH];
long b;
} bufS;
// заполним поля атрибута
bufS.a = 987654321.0;
bufS.b = 99999l;
strcpy(bufS.c,"Исполнение 1");

//заполним параметры структуры атрибута
strcpy(attrPar.comment,"элемент");     //комментарий атрибута
attrPar.key1 = 1;   // рекоменд. как код разработчика
attrPar.key2 = 10;  // рекоменд. как код атрибута
attrPar.key3 = 100; // рекоменд. как код разработчика
attrPar.key4 = 0;   // системный код атрибута
                    // значения от 0 до 1000 зарезервированы АСКОН
attrPar.flagVisible = 0;  //массив, определяющий для каждой колонки
                    // атрибут видимость-невидимость
                    // 0 -видимое поле 1- невидимое поле
attrPar.values = &bufS;  // массив значений ячеек таблицы атрибутов
                    // сначала все значения для 1-ой строки,
                    // затем все значения для 2-ой строки и т.д.
attrPar.valSize = sizeof(bufS);
strcpy(attrPar.password,"111"); //пароль

double x, y;
reference pObj;

RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = " Укажите объект ";

int j = Cursor(&info, &x ,&y, 0, 0);
if (j) {
 if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
```

```
LightObj(pObj, 1);
//запросить имя библиотеки
char nameFile[128];
if(!ChoiceFile("*.tip",NULL, nameFile, 128))
nameFile[0]='\0'; //будем искать в документе
double numb;
j = ReadDouble("Ввести номер типа атрибута", 1000.,0, 1e12,&numb);
if(j) {
reference attr= ksCreateAttr(pObj, &attrPar, numb, nameFile);
if (!attr) MessageBoxResult(); // неудачное завершение
}
LightObj(pObj, 0);
 }
}

}; /* CreateAttr */
```

ksCreateAttr - Пример использования

```
//Создадим атрибут типа "таблица неопределенной длины",
//тип содержит три колонки: DOUBLE_ATTR_TYPE(double),
// STRING_ATTR_TYPE(строка),LINT_ATTR_TYPE(long int).
//Атрибут создадим для внутреннего макроэлемента.
//Макроэлемент на момент создания атрибута временный, т.е. не в модели чертежа.
//Отредактируем атрибут в диалоге редактирования атрибута - добавим строку, удалим.

ksAttribute  attrPar;
memset(&attrPar, 0, sizeof(attrPar));
struct TypeStr {
double a;
char c [MAX_TEXT_LENGTH];
long b;
} ;
 static  TypeStr arrBuf [3] = { {187654321.0, "1 строка", 19999l},
              {287654321.0, "2 строка", 29999l},
              {387654321.0, "3 строка", 39999l} };
```

```
//массив, определяющий для каждой колонки видимость-невидимость
//(вторая колонка выключена)
 unsigned char fV[3]= {1,0,1};
 attrPar.key1 = 1;            // рекоменд. как код разработчика
 attrPar.key2 = 10;             // рекоменд. как код атрибута
 attrPar.key3 = 100;            // рекоменд. как код разработчика
 attrPar.key4 = 0;            // системный код атрибута
                  // значения от 0 до 1000 зарезервированы
                  // за АО "АСКОН"
 attrPar.flagVisible = fV;         //массив, определяющий для каждой колонки
                  // атрибута видимость-невидимость
                  //0 -видимое поле 1- невидимое поле
 attrPar.values = &arrBuf ;       // массив значений ячеек таблицы атрибутов
                // сначала все значения для 1-ой строки,
      // затем все значения для 2-ой строки и т.д.
 attrPar.valSize = sizeof(TypeStr)*3;
 strcpy(attrPar.password,"111");//пароль
 reference pObj;
 char nameFile[128];
 if(!ChoiceFile("*.lat",NULL, nameFile, 128))
  nameFile[0]='\0'; //будем искать в документе
//номер типа атрибута
//(тип создан ранее,
//если nameFile пустое - ищем тип атрибута в документе)
 double numb;
 int j = ReadDouble ("Ввести номер типа атрибута", 1000.,0, 1e12,&numb);
 if(j) {
  //временная группа
  reference g = NewGroup(1);
   //создадим пустой macro
   Macro (0);
   reference m = EndObj();

   pObj = LineSeg(10, 10, 20, 20, 1);
   //добавим отрезок в macro
   ksAddObjectToMacro (m, //указатель на макроэлемент
           pObj ); //указатель на добавляемый объект
//создадим для отрезка, который принадлежит macro атрибут
```

```
  reference attr= ksCreateAttr(pObj, &attrPar, numb, nameFile);
if (!attr)
  MessageBoxResult(); // неудачное завершение - выдадим результат работы нашей функции
  else {
   //вызываем диалог для редактирования атрибута
   ViewEditAttr ( attr, 2, "111"); // пароль атрибута для просмотра 0

   TypeStr arrBuf1 = { 222222222222.0, "новая строка", 4444444l} ;
   //добавим в конец атрибута еще одну строку
   ksAddAttrRow (attr,       // указатель атрибута
         -1,       // номер строки, после которой вставлять, -1 -в конец
         fV,  // указатель на массив флагов видимости ячеек строки
         &arrBuf1,       // указатель на память пользователя, откуда копируются данные
         sizeof(TypeStr),    // размер выделенной памяти под value
         "111");      // пароль атрибута

   //удалим первую строку
   ksDeleteAttrRow (attr,     // указатель атрибута
          0, "111");  // номер строки
   //узнаем количество колонок и строк в атрибуте
   unsigned int rowsCount, columnsCount;
   GetAttrTabInfo (attr,       // указатель атрибута
         &rowsCount,   // количество строк
         &columnsCount);// количество столбцов

   char buf[250];
   sprintf(buf, "rowsCount = %d columnsCount = %d",rowsCount,columnsCount);
   //выведем количество колонок и строк в атрибуте на экран
   Message (buf);
  }
 //закроем временную группу
 EndGroup();
 //поставим все объекты временной группы в модель чертежа
 StoreTmpGroup(g);
}
```

DeleteAttr - Пример использования

```
{
//  создадим атрибут для типа атрибута с колонками
//  DOUBLE_ATTR_TYPE, STRING_ATTR_TYPE, LINT_ATTR_TYPE
//  Предполагается, что данный тип уже существует

Attribute  attrPar;
//структура данных для оформления атрибута
struct {
 double a;
 char c[MAX_TEXT_LENGTH];
 long b;
} bufS;

// заполним поля атрибута
bufS.a = 987654321.0;
bufS.b = 99999l;
strcpy(bufS.c,"Исполнение 1");

//заполним параметры структуры атрибута
strcpy(attrPar.comment,"элемент");     //комментарий атрибута
attrPar.key1 = 1;   // рекоменд. как код разработчика
attrPar.key2 = 10;  // рекоменд. как код атрибута
attrPar.key3 = 100; // рекоменд. как код разработчика
attrPar.key4 = 0;   // системный код атрибута
                    // значения от 0 до 1000 зарезервированы АСКОН
attrPar.flagVisible = 0;  //массив, определяющий для каждой колонки
                    // атрибут видимость-невидимость
                    // 0 -видимое поле 1- невидимое поле
attrPar.values = &bufS;  // массив значений ячеек таблицы атрибутов
                     // сначала все значения для 1-ой строки,
                     // затем все значения для 2-ой строки и т.д.
attrPar.valSize = sizeof(bufS);
strcpy(attrPar.password,"111"); //пароль

double x, y;
reference pObj;
```

```
RequestInfo info;
memset(&info, 0, sizeof(info));
info.promt = " Укажите объект ";

int j = Cursor(&info, &x ,&y, 0, 0);
if (j) {
 if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){

LightObj(pObj, 1);
//запросить имя библиотеки
char nameFile[128];
if(!ChoiceFile("*.tip",NULL, nameFile))
nameFile[0]='\0'; //будем искать в документе
double numb;
j = ReadDouble("Ввести номер типа атрибута", 1000.,0, 1e12,&numb);
if(j) {
reference attr= CreateAttr(pObj, &attrPar, numb, nameFile);
if (!attr) MessageBoxResult(); // неудачное завершение
}
LightObj(pObj, 0);
 }
}
DeleteAttr (pObj, attr, «111»);

};
```

GetAttrValue, SetAttrValue, GetAttrRow, SetAttrRow - Пример использования

```
void GetAttrvalue_Example (void) {

// если у объекта есть атрибут с ключом key1 = 100 и паролем "str",
// то заменим содержимое атрибута
 double x, y;
 reference pObj;
 int j;
 char buf[MAX_TEXT_LENGTH];
```

```
 char s[MAX_TEXT_LENGTH];


RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите объект";

j = Cursor(&info, &x ,&y, 0, 0);
if (j) {
 if (ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
  LightObj(pObj, 1);
  //создадим итератор для поиска по атрибутам объекта с ключами 1,10,100
  reference iter = CreateAttrIterator(pObj, 0,0,0,0,0);
  //встали на первый атрибут
  reference attr = MoveAttrIterator(iter, 'F', 0);
  if (attr) {
   // получили данные для первой строки из таблицы атрибута
   // если тип атрибута - STRING_ATTR_TYPE,то таблица атрибута будет состоять
   // из 1 строки и 1 колонки, ячейка и строка в таблице атрибутов будет
   // совпадать, поэтому в данном случае функции
   // GetAttrRow и GetAttrValue будут работать одинаково

   GetAttrRow (attr, 0, 0, &buf, sizeof(buf));
   sprintf(s, "содержимое атрибута - %s",buf);
   Message(s);

   strcpy(buf, "дуга");
   // изменили содержимое строки в таблице атрибута
   // функции SetAttrRow и SetAttrValue в данном случае будут работать одинаково
   SetAttrRow(attr, 0, 0, &buf, sizeof(buf), "str");
  }
  else Message("атрибут не найден");
  LightObj(pObj, 0);
 }
}

}; /* GetAttrValue */
```

ksDeleteAttrRow, ksAddAttrRow - Пример использования

```
//создадим атрибут для табличного типа неопределенной длины, тип содержит три колонки
//DOUBLE_ATTR_TYPE(double),STRING_ATTR_TYPE(строка),LINT_ATTR_TYPE(long int).
// Атрибут создадим для внутреннего объекта macro.
// Macro на момент создания атрибута временный, т.е. не в модели чертежа.
// Отредактируем атрибут - в диалоге редактирования атрибута, добавим строку, удалим

Attribute attrPar;
struct TypeStr
{
double a;
char c[MAX_TEXT_LENGTH];
long b;
} ;
static TypeStr arrBuf[3] =
{
{187654321.0, "1 строка", 19999l},
{287654321.0, "2 строка", 29999l},
{387654321.0, "3 строка", 39999l}
};

//массив, определяющий для каждой колонки видимость-невидимость
// (вторая колонка выключена)
unsigned char fV[3]= {1,0,1};
attrPar.key1 = 1;// рекоменд. как код разработчика
attrPar.key2 = 10;// рекоменд. как код атрибута
attrPar.key3 = 100;// рекоменд. как код разработчика
attrPar.key4 = 0;// системный код атрибута
// значения от 0 до 1000 зарезервированы
// за АО "АСКОН"
attrPar.flagVisible = fV;// массив, определяющий для каждой колонки атрибута
// видимость-невидимость
// 0 -видимое поле 1- невидимое поле
attrPar.values = &arrBuf ;// массив значений ячеек таблицы атрибутов
// сначала все значения для 1-ой строки,
// затем все значения для 2-ой строки и т.д.
attrPar.valSize = sizeof(TypeStr)*3;
```

```
strcpy (attrPar.password,"111");// пароль, если не пустая строка
// - защищает от несанкционированного
// изменения информации в атрибуте

reference pObj;
char nameFile[128];
if (!ChoiceFile ("*.lat",NULL, nameFile, 128))
nameFile[0]='\0';//будем искать в документе
//номер типа атрибута (тип создан ранее, если nameFile пустое - ищем тип атрибута в документе)
double numb;
int j = ReadDouble ("Ввести номер типа атрибута", 1000.,0, 1e12,&numb);
if (j)
{
//временная группа
reference g = NewGroup (1);
//создадим пустой macro
Macro (0);
reference m = EndObj ();

pObj = LineSeg (10, 10, 20, 20, 1);
//добавим отрезок в macro
ksAddObjectToMacro (m,//указатель на макроэлемент
pObj);//указатель на добавляемый объект
//создадим для отрезка, который принадлежит macro, атрибут
reference attr= CreateAttr (pObj, &attrPar, numb, nameFile);
if (!attr)
MessageBoxResult();// неудачное завершение - выдадим результат работы
// нашей функции
else
{
//вызываем диалог для редактирования атрибута
ViewEditAttr (attr, 2, "111");// пароль атрибута для просмотра 0

TypeStr arrBuf1 = { 222222222222.0, "NEW строка", 4444444l} ;
//добавим в конец атрибута еще одну строку
ksAddAttrRow (attr,// указатель атрибута
-1,// вставлять строку в конец
```

```
fV,// указатель на массив флагов
// видимости ячеек строки
&arrBuf1,// указатель на память пользователя,
// откуда копируются данные
sizeof (TypeStr),// размер выделенной памяти под value
"111");// пароль атрибута

//удалим первую строку
ksDeleteAttrRow (attr,// указатель атрибута
0, "111");  // номер строки и пароль

//узнаем количество колонок и строк в атрибуте
unsigned int rowsCount, columnsCount;
GetAttrTabInfo (attr,// указатель атрибута
&rowsCount,// количество строк
&columnsCount);// количество столбцов

char buf[250];
sprintf (buf, "rowsCount = %d columnsCount = %d",rowsCount,columnsCount);

//выведем количество колонок и строк в атрибуте на экран
Message (buf);
}
//закроем временную группу
EndGroup();
//поставим все объекты временной группы в модель чертежа
StoreTmpGroup(g);
}
```

GetSizeAttrValue - Пример использования

```
void GetAttrvalue_Example (void) {

// если у объекта есть атрибут с ключом key1 = 100 и паролем "str",
// то заменим содержимое атрибута
 double x, y;
 reference pObj;
 int j;
```

```
 char buf[MAX_TEXT_LENGTH];
 char s[MAX_TEXT_LENGTH];

 RequestInfo info;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите объект";

 j = Cursor(&info, &x ,&y, 0, 0);
 if (j) {
  if (ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
   LightObj(pObj, 1);
   //создадим итератор для поиска по атрибутам объекта с ключами 1,10,100
   reference iter = CreateAttrIterator(pObj, 0,0,0,0,0);
   //встали на первый атрибут
   reference attr = MoveAttrIterator(iter, 'F', 0);
   if (attr) {
    // получили данные для первой строки из таблицы атрибута
    // если тип атрибута - STRING_ATTR_TYPE,то таблица атрибута будет состоять
    // из 1 строки и 1 колонки, ячейка и строка в таблице атрибутов будет
    // совпадать, поэтому в данном случае функции
    // GetAttrRow и GetAttrValue будут работать одинаково

    GetAttrRow (attr, 0, 0, &buf, sizeof(buf));

 // GetSizeAttrRow и GetSizeAttrValue будут работать одинаково
    int len1 = GetSizeAttrValue (attr, 0);
 int count = 1; // количество колонок
  int len2 = ksGetSizeAttrValue (attr, 0, &count );
  int len3 = ksGetSizeAttrRow (attr, &count );    sprintf(s, "содержимое атрибута - %s",buf);
    Message(s);

    strcpy(buf, "дуга");
    // изменили содержимое строки в таблице атрибута
    // функции SetAttrRow и SetAttrValue в данном случае будут работать одинаково
    SetAttrRow(attr, 0, 0, &buf, sizeof(buf), "str");
   }
   else Message("атрибут не найден");
   LightObj(pObj, 0);
```

```
 }
 }

}; /* GetAttrValue */
```

GetSizeAttrRow - Пример использования

```
void GetAttrvalue_Example (void) {

// если у объекта есть атрибут с ключом key1 = 100 и паролем "str",
// то заменим содержимое атрибута
 double x, y;
 reference pObj;
 int j;
 char buf[MAX_TEXT_LENGTH];
 char s[MAX_TEXT_LENGTH];
 RequestInfo info;
 memset(&info, 0, sizeof(info));
 info.prompt = "Укажите объект";

 j = Cursor(&info, &x ,&y, 0, 0);
 if (j) {
  if (ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
   LightObj(pObj, 1);
   //создадим итератор для поиска по атрибутам объекта с ключами 1,10,100
   reference iter = CreateAttrIterator(pObj, 0,0,0,0,0);
   //встали на первый атрибут
   reference attr = MoveAttrIterator(iter, 'F', 0);
   if (attr) {
    // получили данные для первой строки из таблицы атрибута
    // если тип атрибута - STRING_ATTR_TYPE,то таблица атрибута будет состоять
    // из 1 строки и 1 колонки, ячейка и строка в таблице атрибутов будет
    // совпадать, поэтому в данном случае функции
    // GetAttrRow и GetAttrValue будут работать одинаково

    GetAttrRow (attr, 0, 0, &buf, sizeof(buf));

// GetSizeAttrRow и GetSizeAttrValue будут работать одинаково
```

```
    int len1 = GetSizeAttrValue (attr, 0);
int count = 1; // количество колонок
 int len2 = ksGetSizeAttrValue (attr, 0, &count );
 int len3 = ksGetSizeAttrRow (attr, &count );
int rowsCount, columnsCount;
int GetAttrTabInfo ( pObj, &rowsCount, &columnsCount );
sprintf( s, "Количество колонок - %i", columnsCount );
Message( s );
sprintf( s, "Количество строк - %i", rowsCount );
Message( s );
    sprintf(s, "содержимое атрибута - %s",buf);
    Message(s);

    strcpy(buf, "дуга");
    // изменили содержимое строки в таблице атрибута
    // функции SetAttrRow и SetAttrValue в данном случае будут работать одинаково
    SetAttrRow(attr, 0, 0, &buf, sizeof(buf), "str");
   }
   else Message("атрибут не найден");
   LightObj(pObj, 0);
  }
 }

}; /* GetAttrValue */
```

GetAttrKeysInfo - Пример использования

```
void ReadObjAttr ()
{
struct
{
double d;
char s[MAX_TEXT_LENGTH];
long l;
}
sBuf;
double x, y;
reference pObj;
```

```
int j;
char buf[127];
RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите объект";
do
{
j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj(x, y, MAXDOUBLE)))
{
LightObj(pObj, 1);
//создадим итератор для хождения по атрибутам объекта

reference iter = CreateAttrIterator (pObj, 0,0,0,0,0);
//встали на первый атрибут
reference attr = MoveAttrIterator (iter, 'F', 0);
if (attr)
{
Message("тип и ключи атрибута");
unsigned int k1,k2,k3,k4;
double numb;
GetAttrKeysInfo (attr,&k1,&k2,&k3,&k4,&numb);
sprintf (buf, "k1=%d k2=%d k3=%d k4=%d
numb=%f ", k1, k2, k3, k4, numb);
Message (buf);

Message ("количество колонок");
unsigned int col;
GetAttrTabInfo (attr, 0, &col);
sprintf (buf, "column=%d",col);
Message (buf);

Message("информация о колонках");
ColumnInfo par;//структура для первой колонки
par.columns = CreateArray (ATTR_COLUMN_ARR);
par.fieldEnum = CreateArray (CHAR_STR_ARR);
```

```
for (int i=0; i < col; i)
{
GetAttrColumnInfo (attr, i, &par);
ShowCol (par,i, FALSE);
}
DeleteArray (par.columns);
DeleteArray (par.fieldEnum);

Message("строка атрибута");
GetAttrRow (attr, 0, 0, &sBuf, sizeof (sBuf));
sprintf (buf, "d=%f s=%s l=%ld",sBuf.d, sBuf.s, sBuf.l);
Message (buf);

Message("заменим строку атрибута");
sBuf.d = numb;
strcpy (sBuf.s,"1234567\nasdfgh\nzxcvb");
sBuf.l = 88888l;
SetAttrRow (attr, 0, 0, &sBuf, sizeof (sBuf), "111");
memset (&sBuf,0, sizeof(sBuf));
GetAttrRow (attr, 0, 0, &sBuf, sizeof (sBuf));
sprintf (buf, "d=%f s=%s l=%ld",sBuf.d, sBuf.s, sBuf.l);
Message (buf);
}
else
Message("атрибут не найден");
LightObj(pObj, 0);
}
}
}
while (j);
}
```

GetAttrColumnInfo - Пример использования

```
void ReadObjAttr() {
 struct {
  double d;
  char s[MAX_TEXT_LENGTH];
```

```
 long l;
} sBuf;
double x, y;
reference pObj;
int j;
char buf[127];
RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите объект";

do {
 j = Cursor(&info, &x ,&y, 0);
 if (j) {
  if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
   LightObj(pObj, 1);
   //создадим итератор для хождения по атрибутам объекта

   reference iter = CreateAttrIterator(pObj, 0,0,0,0,0);
   //встали на первый атрибут
   reference attr = MoveAttrIterator(iter, 'F', 0);
   if (attr) {

    Message("тип и ключи атрибута");
    unsigned int k1,k2,k3,k4;
    double numb;
    GetAttrKeysInfo(attr,&k1,&k2,&k3,&k4,&numb);
    sprintf (buf, "k1=%d k2=%d k3=%d k4=%d numb=%f ",k1,k2,k3,k4,numb);
    Message(buf);

    Message("количество колонок");
    unsigned int col;
    GetAttrTabInfo(attr, 0, &col);
    sprintf (buf, "column=%d",col);
    Message(buf);

    Message("информация о колонках");
    ColumnInfo par; //структура для первой колонки
    par.columns = CreateArray(ATTR_COLUMN_ARR);
```

```
    par.fieldEnum = CreateArray(CHAR_STR_ARR);
    for (int i=0; i < col; i) {
     GetAttrColumnInfo (attr, i, &par);
     ShowCol(par,i, FALSE);
    }
    DeleteArray(par.columns);
    DeleteArray(par.fieldEnum);

    Message("строка атрибута");
    GetAttrRow (attr, 0, 0, &sBuf, sizeof(sBuf));
    sprintf (buf, "d=%f s=%s l=%ld",sBuf.d, sBuf.s, sBuf.l);
    Message(buf);

    Message("заменим строку атрибута");
    sBuf.d = numb;
    strcpy(sBuf.s,"1234567\nasdfgh\nzxcvb");
    sBuf.l = 88888l;
    SetAttrRow(attr, 0, 0, &sBuf, sizeof(sBuf), "111");
    memset(&sBuf,0, sizeof(sBuf));
    GetAttrRow (attr, 0, 0, &sBuf, sizeof(sBuf));
    sprintf (buf, "d=%f s=%s l=%ld",sBuf.d, sBuf.s, sBuf.l);
    Message(buf);


   }
   else
    Message("атрибут не найден");
   LightObj(pObj, 0);
  }
 }
} while (j);
}
```

ViewEditAttr - Пример использования

```
void ViewEditAttr_Example (void) {

 // поиск объектов по значению атрибута
```

```
double x, y;
//найдем объект
int j;
reference pObj;

RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите объект";

do {
 j = Cursor(&info, &x ,&y, 0, 0);
 if (j) {
  if(ExistObj(pObj = FindObj(x, y, MAXDOUBLE))){
  //создадим итератор для движения по атрибутам с ключом 10
   reference iter = CreateAttrIterator(pObj, 10, 0,0,0,0);
   LightObj(pObj, 1); //подсветили объект
   // позиционировались на первый атрибут
   reference pAttr = MoveAttrIterator( iter, 'F', 0);
   if (pAttr) {
    do {
     // выдать параметры атрибута
     ViewEditAttr (pAttr, 1, 0);
     // на следующий атрибут
     pAttr = MoveAttrIterator( iter, 'N', 0);
    } while(pAttr);
   }
   LightObj(pObj, 0); //выключим подсветку
   DeleteIterator(iter);
  }
 }
}while(j);

}; /* ViewEditAttr */
```

ViewEditAttrType - Пример использования

```
void ViewEditAttrType_Example (void) {
```

```
//выбираем библиотеку типов , задаем номер, пароль типа и редактируем тип

 char nameFile[128];
 char password[10];
 int j;
 if(!ChoiceFile("*.tip", nameFile))
  nameFile[0]='\0'; //будем искать в документе
 double numb;
 j = ReadDouble("номер типа атрибута", 1000.,0, 1e12,&numb);
 if (j) {
  j = ReadString("пароль типа атрибута", password, 10);
  if (j) {
   //отредактировать атрибут с уникальным номером numb
   ViewEditAttrType (nameFile, 2, numb, password);
  }
 }

}; /* ViewEditAttrType */
```

CreateAttrIterator - Пример использования

```
double x, y;
reference pObj;
char password[11];
int j;
RequestInfo info;
memset (&info, 0, sizeof(info));
info.prompt = "Укажите объект";
do
{
j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj (x, y, 1e6)))
{
LightObj (pObj, 1);
//создадим итератор для хождения по атрибутам объекта
```

```
reference iter = CreateAttrIterator (pObj, 0,0,0,0,0);
//встали на первый атрибут
reference attr = MoveAttrIterator (iter, 'F', 0);
if (attr)
{
j = ReadString ("Ввести пароль типа атрибута", password, 10);
if (j)
{
//удалить атрибут
if (!DeleteAttr (pObj, attr, password))
MessageBoxResult();// неудачное завершение -
// выдадим результат работы
// нашей функции
}
}
else
Message("атрибут не найден");
LightObj (pObj, 0);
}
}
}
while (j);
```

ChoiceAttr - Пример использования

```
double x, y;
reference pObj;
int j;
RequestInfo info;
memset(&info, 0, sizeof(info));
info.prompt = "Укажите объект";

do
{
j = Cursor (&info, &x ,&y, 0);
if (j)
{
if (ExistObj(pObj = FindObj (x, y, 1e6)))
```

```
{
LightObj (pObj, 1);
ChoiceAttr (pObj);
LightObj (pObj, 0);
}
}
}
while (j);
```

ksGetLibraryAttrTypesArray - Пример использования

```
//получим массив типов атрибутов в spc.lat
 reference typeArr = ksGetLibraryAttrTypesArray(("d:\\0\\Spc.lat"));  // полное имя библиотеки стилей

 if (typeArr) {
  char buf[128];
  //определим количество типов и отобразим на экране
  int count = GetArrayCount(typeArr);
  sprintf(buf, "count = %d", count);
  Message(buf);
  //в цикле получим информацию о каждом типе и отобразим на экране
  for (uint i = 0; i < count; i) {
   LibraryAttrTypeParam par;
   GetArrayItem(typeArr, // указатель на массив
         i,    // индекс в массиве (нумерация начинается с 0)
         &par,   // указатель на структуру элемента
         sizeof(LibraryAttrTypeParam)); // размер структуры элемента

   sprintf(buf, "ID = %f\nname=%s", par.typeId, par.name);
   Message(buf);
  }
 }
```

Caption - пример использования

```
KGAX1.Caption = "Заголовок";                    (* )
Caption = KGAX1.Caption;                        (* )
KGAX1.SetCaption("Заголовок");                  (**)
```

```
Caption = KGAX1.GetCaption();                    (**)
```

Text - пример использования

```
KGAX1.Text = "Заголовок";                        (* )
Text = KGAX1.Text;                               (* )
KGAX1.SetText ("Заголовок");                     (**)
Text = KGAX1.GetText();                          (**)
```

DocumentType - пример использования

```
KGAX1.DocumentType =  vt_SheetStandart;          (* )
Type = KGAX1.DocumentType;                       (* )
KGAX1.SetText (vt_SheetStandart);                (**)
Type = KGAX1.GetDocumentType();                  (**)
```

DocumentFileName - пример использования

```
KGAX1.DocumenFileName = "C:\\1.frw";                    (* )
fileName = KGAX1.DocumenFileName;                       (* )
KGAX1.SetDocumenFileName ("C:\\1.frw");                 (**)
fileName = KGAX1.GetDocumenFileName();                  (**)
```

Document3DDrawMode - пример использования

```
KGAX1.DocumenFileName = "C:\\1.frw";                    (* )
fileName = KGAX1.DocumenFileName;                       (* )
KGAX1.SetDocumenFileName ("C:\\1.frw");                 (**)
fileName = KGAX1.GetDocumenFileName();                  (**)
```

Document3DWireframeShadedMode - пример использования

```
KGAX1.Document3DWireframeShadedMode = TRUE;             (* )
Mode = KGAX1.Document3DWireframeShadedMode;             (* )
KGAX1.SetDocument3DWireframeShadedMode (TRUE);          (**)
KGAX1.GetDocument3DWireframeShadedMode();               (**)
```

Пример описания ресурсов

```
#define END_OF_STRING_RESOURCE_TABLE 0
```

```
#define END_OF_RESOURCE_TABLE       0xffff

#define ID_PROP_ELEMENT_PARAM   20000 // Группа параметры элемента
#define PROP_DOUBLE_LIST        20001 // Список вещественных значений
#define PROP_INT_LIST           20002 // Список целых значений
#define PROP_STRING_LIST        20003 // Список строк
#define ID_PROP_VIEW            20004 // Вид

STRINGTABLE DISCARDABLE
BEGIN
  ID_PROP_ELEMENT_PARAM   "Параметры элемента"
  PROP_DOUBLE_LIST        "Список вещественных значений"
  PROP_INT_LIST           "Список целых значений"
  PROP_STRING_LIST        "Список строк"
END
```

Для фиксированного списка вещественных значений:

```
PROP_DOUBLE_LIST RCDATA
{
  L"1.0\0"
  L"2.0\0"
  L"3.0\0"
  L"4.0\0"
  L"5.0\0"
  END_OF_STRING_RESOURCE_TABLE
}
```

Для фиксированного списка целых значений:

```
PROP_INT_LIST RCDATA
{
  L"1\0"
  L"2\0"
  L"3\0"
  L"4\0"
  L"5\0"
  END_OF_STRING_RESOURCE_TABLE
}
```

Для фиксированного списка строк:

```
PROP_STRING_LIST RCDATA
{
  L"Первая строка\0"
  L"Вторая строка\0"
  L"Третья строка\0"
  END_OF_STRING_RESOURCE_TABLE
}
```

Пример описания ресурсов для списка битмапов ksOPControlListBmp:

```
#define IDB_ID_VIEW                 1101 // ID_VIEW    - Вид
#define IDB_ID_VIDSEC                1102 // ID_VIDSEC  - Вид разрез
#define IDB_ID_TOPVID                1103 // ID_TOP     - Вид сверху
#define IDB_ID_SIDEVID               1111 // ID_SIDE    - Вид сбоку

IDB_ID_VIEW       ICON DISCARDABLE "res\\G_VIEW.ico"
IDB_ID_TOPVID     ICON DISCARDABLE "res\\G_TOP.ico"
IDB_ID_SIDEVID    ICON DISCARDABLE "res\\G_LEFT.ico"
IDB_ID_VIDSEC     ICON DISCARDABLE "res\\G_SEC.ico"

STRINGTABLE DISCARDABLE
BEGIN
  ID_PROP_VIEW          "Вид"
  IDB_ID_VIEW           "Вид"
  IDB_ID_TOPVID         "Вид сверху"
  IDB_ID_SIDEVID        "Вид сбоку"
  IDB_ID_VIDSEC         "Вид\\разрез"
END

// Список значений
ID_PROP_VIEW RCDATA
{
  IDB_ID_VIEW
  IDB_ID_TOPVID
  IDB_ID_SIDEVID
  IDB_ID_VIDSEC
```

```
  END_OF_RESOURCE_TABLE
}
```

Пример передачи в список массива

```
long varsCount = 100;

// Массив для наполнения комбобокса
SAFEARRAYBOUND sabNewArray;
sabNewArray.cElements = varsCount;
sabNewArray.lLbound = 0;
SAFEARRAY * pSafe = ::SafeArrayCreate( VT_R8, 1, &sabNewArray );

if( pSafe )
{
  for ( long j = 0; j < varsCount; j++ )
  {
    double  val = j * 0.5;
    ::SafeArrayPutElement( pSafe, &j, &val);
  }

  _variant_t varArr;
  varArr.vt = VT_ARRAY | VT_R8;
  varArr.parray = pSafe;
  // Передадим массив значений в контрол
  dt->Add( varArr );
---
```

Пример использования утилиты

```
//---------------------------------------------------------------------------
// Интерфейс связи с вспомогательным контроллером
// ---
_COM_SMARTPTR_TYPEDEF(IWow32Util, __uuidof(IWow32Util));
IWow32UtilPtr wow32Util;

//---------------------------------------------------------------------------
// Соединение в вспомогательному Win32 приложению (exe) для получения доступа
```

```
// к 32 разрядным драйверам для которых не найдены 64 разрядные аналоги
// ---
IWow32UtilPtr GetWow32Util()
{
if ( !(bool)wow32Util )
{
try
{
wow32Util.CreateInstance( _T("Wow32Util.Application") );
}
catch ( _com_error & )
{

}
}
return wow32Util;
}

//---------------------------------------------------------------------------
// Создание интерфейса во внешнем 32 разрядном приложении
//---
IDispatchPtr CreateWow32Interface( LPCTSTR ClassID )
{
IDispatchPtr res;
IWow32UtilPtr util( GetWow32Util() );
if ( util )
{
try
{
_bstr_t id( ClassID );
util->CreateWow32Interface( id, &res );
}
catch (_com_error&)
{
//AR todo ComErrorMessage
}
}
return res;
```

```
}

//--------------------------------------------------------------------------
// Создание интерфейса во внешнем 32 разрядном приложении
//---
#pragma comment( lib, "rpcrt4.lib" )
_LIBFUNC(IDispatchPtr) CreateWow32Interface( const CLSID& rclsid )
{
_TUCHAR * str = NULL;
if ( RPC_S_OK != UuidToString((UUID *)&rclsid, &str) || !str )
return NULL;
_bstr_t classId( "{" );
classId += str;
classId += _T("}");

IDispatchPtr res( CreateWow32Interface(classId) );

RpcStringFree( &str );
return res;
}

#define JET_OLED_PROVIDER _T("Provider=Microsoft.Jet.OLEDB.4.0")
#define ACE_OLED_PROVIDER _T("Provider=Microsoft.ACE.OLEDB.12.0")
static bool usesJET_OLED_PROVIDER = false;
static bool usesWOW32_PROVIDER = false;
//--------------------------------------------------------------------------
/// Создать подключение к БД
/**
param fullFileName - Полный путь + имя файла
param forWrite - Для записи
*/
// ---
ado::_ConnectionPtr ExcelDataBaseADO::OpenConnection( LPCTSTR fullFileName, bool
forWrite )
{
ado::_ConnectionPtr pConnection;

try
```

```
{
// Создадим соединение
#ifdef _WIN64
if ( usesWOW32_PROVIDER )
pConnection = CreateWow32Interface( __uuidof( ado::Connection ) );
else
#endif
pConnection.CreateInstance(__uuidof( ado::Connection ));

// Открываем соединение с БД( если файл excel не существует то он создасться)
if ( pConnection )
{
CString connName = usesJET_OLED_PROVIDER ? JET_OLED_PROVIDER :
ACE_OLED_PROVIDER;
connName += _T(";Data Source=");
connName += fullFileName;
connName += _T(";Extended Properties="Excel 8.0;HDR=NO;");

if ( !forWrite )
connName += _T("IMEX=1;");

connName += _T("""");

pConnection->Open( (LPCTSTR)connName, _T(""), _T(""), 0 );
}
}
#ifdef _DEBUG
catch ( _com_error & er )
{
if ( usesJET_OLED_PROVIDER && usesWOW32_PROVIDER )
::ComErrorMessage( er );

pConnection = NULL;
}
#else
catch ( _com_error& )
{
pConnection = NULL;
```

```
}
#endif
if ( !(bool)pConnection && !usesJET_OLED_PROVIDER )
{
usesJET_OLED_PROVIDER = true;
pConnection = OpenConnection( fullFileName, forWrite );
usesJET_OLED_PROVIDER = false;

#ifdef _WIN64
if ( !(bool)pConnection && !usesWOW32_PROVIDER )
{
usesWOW32_PROVIDER = true;
pConnection = OpenConnection( fullFileName, forWrite );
// usesWOW32_PROVIDER = false;
}
#endif

}
return pConnection;
}
```

Получение интерфейса Recordset также требуется делать через CreateWow32Interface

```
#ifdef _WIN64
if ( usesWOW32_PROVIDER )
rset = CreateWow32Interface(_T("ADODB.Recordset"));
else
#endif
rset.CreateInstance(_T("ADODB.Recordset"));
```

STEPS 1

Step2, математика

- Пересечение прямых.
- Пересечение кривых.
- Пересечение отрезка и дуги.
- Касательная из точки.
- Касательная под углом.
- Поворот точки.
- Симметрия точки.
- Сопрягающие окружности к двум прямым.
- Перпендикуляр.

Step2a - массив неопределенной длины

- Массив строк
- Массив математических точек.
- Массив строк объекта "текст".
- Массив колонок типа атрибута.
- Массив полилиний.
- Массив габаритных прямоугольников.
- Массив структур пользователя.
- Массив экземпляров класса пользователя.
- Массив параметров узла дерева библиотеки.

Step3 - объекты

- Создание документа.
- Виды.
- Слои.
- Группы.
- Именованная группа.
- Отрезки.
- Дуги.
- Линии.
- Окружности.
- Точки.
- Bezier-сплайны.
- Штриховка.
- Текст.

Step3a

- Контур.
- Технические требования.
- Стрелка вида.
- Работа со штампом.
- Таблица.
- Эквидистанта.
- Эллипс.
- Полилиния.
- Nurbs.
- Допуск формы.
- Одинаковая шероховатость.
- Вставка фрагмента внешней ссылкой.
- Вставка локального фрагмента.

Step4

- Работа с БД.
- Ввод длинного целого.
- Выбор имени файла.
- Пример с обработкой очереди сообщений.
- Относительный путь к файлу.
- Работа с системными папками.
- Записать слайд.

Step4_1

Процессы Cursor и Placement.

Step4_2

Отрисовать слайд.

Step5

- Трансформация объекта по матрице.
- Копирование объекта.
- Симметрия объекта.
- Редактирование допуска формы.
- Редактирование таблицы.
- Взять тексты граф и редактировать штамп.
- Получить текст технических требований.
- Редактирование технических требований.

- Вставка фрагмента.
- Работа с библиотекой фрагментов.
- Вставка фрагмента россыпью.
- Редактирование таблиц.
- Редактирование допуска формы.
- Просмотр допуска формы.
- Просмотр таблицы.

Step6

Запомнить указатель.

Step7

- Хождение по виду, макроэлементу, документам, видам, именованным и рабочим группам, слоям, группе.
- Хождение по элементам документа с определенным атрибутом, по атрибутам объекта.

S tep8

- Создать тип атрибута.
- Удалить тип атрибута.
- Получить тип атрибута.
- Заменить тип атрибута.
- Создать атрибут определенного типа.
- Удалить атрибут документа, удалить атрибут.
- Считать атрибут.
- Просмотреть атрибут.
- Просмотреть библиотеку.
- Просмотреть тип.
- Просмотреть атрибут.
- Создать атрибут документа, создать атрибут 3D документа.
- Просмотреть атрибут документа.
- Удалить атрибут 3D документа.
- Просмотреть атрибут документа, просмотреть атрибут объектов 3D документа.

Step9

- Для всех размеров.
- Линейный размер.
- Угловой размер.
- Шероховатость.
- Линия-выноска.

- Позиционная линия-выноска.
- Клеймение.
- Маркирование.
- Обозначение базы.
- Линия разреза/сечения,
- Диаметральный размер.
- Радиальный размер.
- Радиальный размер с изломом.
- Стрелка вида.

Step10

- Создание типа атрибута болта.
- Создание объекта спецификации "деталь", объекта спецификации для раздела "Стандартные изделия".
- Конвертировать спецификацию во фрагмент.
- Просмотреть спецификацию.

Step11

Пример запроса командного окна.

SlideWrk

Записать слайд.

Разбить слайд.

Отрисовать слайд.

Step3d1

Для операций:

- Базовая операция выдавливания.
- Операции вращения.
- Операции по сечениям.
- Создание фаски и скругления.
- Операции оболочка, уклон, сечение плоскостью, сечение эскизом.

Step3d2

Для операций:

- Итератор по документам
- Использование массива элементов.
- Взять/изменить имя компоненты.
- Фиксирование и установка стандартного объекта.

- Получить и заменить параметры цвета компоненты.
- Взять и поменять внешние переменные компоненты.
- Получить и изменить место расположения детали в сборке.
- Получить интерфейс ksEntity объекта создаваемого системой по умолчанию и поменять параметры.
- Создать эскиз.
- Сформировать массив объектов (здесь эскизов) и вернуть его интерфейс.
- Установить и получить параметры пользователя в компоненте.

Step3d3

- Конструктивная ось операции.
- Конструктивная ось по двум точкам.
- Конструктивная ось, проходящая через ребро.
- Смещенная плоскость, ось по двум плоскостям, плоскость под углом к другой плоскости .
- Плоскость через три вершины.

Step12

Пример создания пользовательской панели свойств

EventCOM

- События, подписка и отписка.
- События приложения.
- События документа.
- События графического документа.
- События документа-модели.
- События объектов графического документа.
- События объектов документа-модели.
- События документа-спецификации.
- События объектов документа-спецификации.
- События редактирования описаний спецификации.
- События основной надписи.
- События окна документа.
- События менеджера выбора объектов.

EventAuto

- События приложения.
- События графического документа.
- События документа-модели.

- События документа.
- События окна документа.
- События объектов графического документа.
- События объектов документа-модели.
- События менеджера выбора объектов.
- События документа-спецификации.
- События объектов документа-спецификации.
- События редактирования описаний спецификации.
- События основной надписи.

Cube

ITessellation, ksFaceDefinition::GetTessellation, ITessellationks::GetFacetPoints, ITessellationks::GetFacetNormals, ksPart::TransformPoint, ksDocumentFrameNotify::BeginPaintGL, ksDocumentFrameNotify::ClosePaintGL, ksGLObject

Gayka1

COM

HotPointDescription, ksColouring, IPropertyMultiButton, IPropertyMultiButton::AddButton, IPropertyList, IPropertySeparator, IPropertyControls::Add.

IApplication::CreateProcessParam, IProcessParam::specToolbar, IProcessParam::Caption, IProcessParam::AutoReduce, IProcessParam::DefaultControlFix, IProcessParam::PropertyTabs, IPropertySlideBox, IPropertyGrid, IPropertyTab, IPropertyTabs::Add, IPropertyCheckBox, IPropertyUserControl, ksOpenHelpFile.

ILibHPObject, ILibHPObject::LibHotPnt_GetMenu, ILibHPObject::LibHotPnt_Prepare, ILibHPObject::LibHotPnt_Complete, ILibHPObject::LibHotPnt_Get, ILibHPObject::LibHotPnt_Set, ILibHPObject::LibHotPnt_GetCursorText, ILibHPObject::LibHotPnt_ExecuteCommand. ILibPropertyObject.

Step1_API7_2D - строительные обозначения

- Марка/позиционное обозначение.
- Марка/позиционное обозначение с линией-выноской.
- Марка/позиционное обозначение на линии.
- Выносная надпись к многослойным конструкциям.
- Фигурная скобка.
- Прямая координационная ось.
- Круговая координационная ось.
- Номер узла.
- Обозначение узла.
- Обозначение узла в сечении.

Автоматизация (Automation)

Технология Automation реализована в API системы КОМПАС через интерфейсы IDispatch.

С использованием интерфейсов такого типа можно получить доступ к системе как для работы с графическими документами, так и с трехмерными моделями. При этом может быть реализован весь функционал систем КОМПАС-ГРАФИК и КОМПАС-3D.

Dispath-интерфейсы могут быть использованы при программировании в большинстве современных сред: Visual Basic, Visual C++, C+Builder, Delphi.

Функции оформления библиотек

DisplayLibraryName - Получить имя библиотеки

DisplayLibraryNameW - Получить имя библиотеки (Unicode)

LibInterfaceNotifyEntry- Головная функция библиотеки, подписка на обработку событий

LibIsOnApplication7 - Задать тип версии API, используемого библиотекой

LibObjInterfaceEntry - Получить имя библиотеки (Unicode)

LibraryBmpBeginID - Получить для указанного размера иконок начало диапазона идентификаторов иконок команд библиотеки

LIBRARYBMPSIZE - Задать размер окна вывода растрового слайда

LIBRARYENTRYDEMOEX - Головная функция библиотеки (демо режим)

LIBRARYENTRYDEMO - Головная функция библиотеки (демо режим)

LIBRARYENTRY - Головная функция библиотеки

LIBRARYHELPFILE - Определить имя файла справочной системы, подключаемого к библиотеке

LIBRARYHELPFILEW - Определить имя файла справочной системы, подключаемого к библиотеке (Unicode)

LibraryHintTipsBeginID - Задать идентификатор начала диапазона ресурсов для подсказок к командам

LIBRARYID - Задать идентификатор ресурсов

LIBRARYNAME - Задать имя библиотеки

LIBRARYNAMEW - Задать имя библиотеки (Unicode)

LibToolBarId - Получить идентификаторы инструментальных и компактных панелей

Пример использования функции LibToolBarId

```
//---------------------------------------------------------------------------
// Получить идентификаторы панелей
// barType == 0 - компактная панель 1 - простая панель
// index - индекс панели
// ---
extern "C" int WINAPI __export LibToolBarId( int barType, int index ){
```

```
 if ( !barType )
 {
   // Компактная панель "Крепежные элементы"
   return !index ? COMPACT_BAR_CONSTR : -1;
 }
#ifndef __LIGHT_VERSION__
 else
 {
  switch( index )
   {
     case 0: return BAR_CONSTRUCT; // Панель "Конструктивные элементы"
     case 1: return BAR_PROFILE;   // Панель "Профили"
   }
 }
#endif
 return -1;
}
```

Кнопки типа Fly-Out

1. Для создания кнопки типа Fly-Out необходимо описать RCDATA состава выпадающей панели. Идентификатор команды, указанной в инструментальной панели, должен быть увеличен на 1000 (COMMAND_ID 1000).
2. В RCDATA должна быть та же команда, иначе она станет недоступной после загрузки панели.
3. После загрузки основной панели на Fly-Out кнопке активной будет первая кнопка из выпадающей панели.

Пример использования функций оформления библиотеки

Пример:

```
#define LIB_ID 1000
// Определить имя библиотеки /* LIBRARYNAME */
char*?WINAPI LIBRARYNAME()
 return "Простая библиотека";
}
// Задать идентификатор ресурсов
/* LibraryID */
int?WINAPI LIBRARYID()
{
 return LIB_ID; /*Идентификатор библиотеки равен 1000*/
}
```

```
// Задать размер окна вывода BITMAP-слайда?
/* LibraryBmpSize */
long WINAPI LIBRARYBMPSIZE()
{
 return MAKELONG( 170, 160 );
}

// Определить имя файла справочной системы
/* LibraryHelpFile */
char WINAPI LIBRARYHELPFILE()
{
 return "Konstr.hlp";
}
// Головная функция библиотеки
void WINAPI LIBRARYENTRY( UINT Comm )
{
switch ( Comm )
{
case 1:
 DrawBolt7798;
break;
case 2:
 DrawShayba18123;
break;

case 3:
 DrawGayka6393;
 break;

case 4:
 DrawGayka5915;
 break;
case 5:
 DrawGayka15521;?
 break;
case 6:
 DrawShtift10774;?
 break;
}
};

/* LibraryEntry*/
// Получить идентификаторы панелей
int WINAPI LibToolBarId (int barType, int index)
```

```
{
  // Компактные панели
  if(!barType)
  {
    return !index ? COMPACT_BAR1 : -1;
  }
  // Инструментальные панели
  else
  {
    switch (index)
    {
      case 0  : return BAR_1;
      case 1  : return BAR_2;
      case 2  : return BAR_3;
    }
  }
  return -1;
}
/*LibToolBarId*/
/*
```

// Пример файла ресурсов (rc):

// С каждой строкой меню будет выводиться соответствующий по номеру команды BITMAP - слайд, иллюстрирующий выбранную операцию.

//Размер слайда может быть произвольным. Он определяется функцией LIBRARYBMPSIZE.

//битмапы для отображения в менеджере библиотек (идентификатор равен номеру команды)

// Определение BITMAP-слайдов для библиотечных функций:

1 BITMAP "B7798.bmp"

2 BITMAP "Sh18123.bmp"

3 BITMAP "G6393.bmp"

4 BITMAP "G5915.bmp"

5 BITMAP "G15521.bmp"

6 BITMAP "Sh10774.bmp"

// Определение в файле ресурсов меню операций.

```
// Примечание: Идентификаторы команд меню должны быть в интервале от 1 до 900.
// Команда выбранной строчки меню будет передана в функцию LibraryEntry.
KONSTR_1 MENU
{
 MENUITEM "Болт ГОСТ 7798-70", 1
  MENUITEM "Шайба ГОСТ 18123-72", 2
  POPUP "ГАЙКИ"
  {
    MENUITEM "Гайка ГОСТ 6393-73", 3
    POPUP "ГАЙКИ ШЕСТИГРАННЫЕ"
    {
      MENUITEM "Гайка ГОСТ 5915-70", 4
      MENUITEM "Гайка ГОСТ 15521-50", 5
    }
  }
  MENUITEM "Штифт ГОСТ 10774-80", 6
}
// Определение размеров окна вывода BITMAP-слайда.
// Примечание: Идентично вызову ф-ии LIBRARYBMPSIZE, но приоритет вызова функции выше.
KONSTR_1 RCDATA
{
  170   // Размер по горизонтали
  160   // Размер по вертикали
}
// Определение имени библиотеки:
// Примечание: Идентично вызову ф-ии LIBRARYNAME, но приоритет вызова функции выше.
STRINGTABLE
{
  KONSTR_1 "Конструкторская библиотека"
}
// Определение имени иконки для минимизированного окна библиотеки:
KONSTR_1 ICON "konstr.ico"
// При подключении библиотеки будет создана компактная панель "Конструкторская библиотека", в которую входят
// инструментальные панели "Болты" и "Шайбы". Также отдельно будет создана инструментальная панель "Гайки".
```

```
// Команда "Штифт ГОСТ 10774-80" не входит ни в одну из панелей.
// Определение имени панелей:
STRINGTABLE {
  BAR_1 "Болты"
  BAR_2 "Шайбы"
  BAR_3 "Гайки"
  COMPACT_BAR1 "Конструкторская библиотека"
}
#define END_OF_RESOURCE_TABLE              0xffff
// Определение состава панелей:
COMPACT_BAR1 RCDATA
{
  BAR_1
  BAR_2
  END_OF_RESOURCE_TABLE
}
BAR_1 RCDATA
{
  1 // Болт ГОСТ 7798-70
  END_OF_RESOURCE_TABLE
}
BAR_2 RCDATA
{
  2 // Шайба ГОСТ 18123-72
  END_OF_RESOURCE_TABLE
}
BAR_3 RCDATA
{
  3 // Гайка ГОСТ 6393-73
  4 // Гайка ГОСТ 5915-70
  5 // Гайка ГОСТ 15521-50
  END_OF_RESOURCE_TABLE
}
// Определение иконок инструментальных панелей:
BAR_1 ICON DISCARDABLE "bar1.ico"
BAR_2 ICON DISCARDABLE "bar2.ico"
BAR_3 ICON DISCARDABLE "bar3.ico"
*/
```

//битмапы для отображения в панелях и в настройках интерфейса (идентификатор равен номеру команды + 1000) размер битмапа должен быть 22x22 пиксела

1001 BITMAP "B7798.bmp"

1002 BITMAP "Sh18123.bmp"

1003 BITMAP "G6393.bmp"

1004 BITMAP "G5915.bmp"

1005 BITMAP "G15521.bmp"

1006 BITMAP "Sh10774.bmp"

Пример использования функции LibraryBmpBeginID

```
---------------------------------------
// Получить начало диапазона иконок для команд библиотеки
// ---
unsigned int WINAPI LibraryBmpBeginID( unsigned int bmpSizeType )
{
  int res = 0;
  switch ( bmpSizeType )
  {
    case ksBmp1616: res = 1000; break;
    case ksBmp2424: res = 2000; break;
  }
  return res;
}
```

Функции

DisplayLibraryName - Получить имя библиотеки, отображаемое на экране

ExternalGetImage - Получить слайд для команды библиотеки с указанным номером

ExternalGetMenu - Получить меню библиотеки

ExternalGetResourceModule - Задать модуль с ресурсами библиотеки.

ExternalGetToolBarId - Задать идентификаторы панелей управления библиотеки

ExternalMenuItem - Получить строку меню для создания меню в виде строк

ExternalRunCommand - Головная функция библиотеки

GetHelpFile - Получить имя файла справочной системы, подключаемого к библиотеке

GetImageHeight - Задать высоту окна вывода слайда

GetImageWidth - Задать ширину окна вывода слайда

GetLibraryName - Получить имя библиотеки

IsOnApplication7 - Получить тип версии API используемого библиотекой.

Пример описания ресурсов панелей и кнопки типа Fly-Out

```
// Определение имени:
STRINGTABLE {
  COMPACT_BAR1 "Компактная панель"
}
#define END_OF_RESOURCE_TABLE              0xffff
// Определение состава:
COMPACT_BAR1 RCDATA
{
  BAR_1 // Инструментальная панель, входящая в компактную
  END_OF_RESOURCE_TABLE
}
```

Необходимые ресурсы для инструментальной панели:

```
// Определение имени панелей:
STRINGTABLE {
  BAR_1 "Инструментальная панель"
}
// Определение состава панели:
BAR_1 RCDATA
{
  1 // Идентификатор команды библиотеки
  END_OF_RESOURCE_TABLE
}
```

// Определение иконок инструментальной панели при включении ее в компактную панель:

```
BAR_1 ICON DISCARDABLE "bar1.ico"
// Определение состава панели:
BAR_1 RCDATA
{
  COMMAND_1              // Идентификатор команды библиотеки (Кнопка типа Fly-Out)
  ...                    // Остальные команды
  END_OF_RESOURCE_TABLE
}
// Определение кнопки типа Fly-Out
COMMAND_1 1000  RCDATA  // Кнопка с идентификатором COMMAND_1 на панели BAR_1 будет Fly-Out кнопкой
```

```
{
 COMMAND_1          // Тот же идентификатор желательно ставить первым, чтобы после загрузки
                    // эта команда становилась текущей
 ...                // Остальные команды
 END_OF_RESOURCE_TABLE
```

Параметры библиотек типа Converter

Тип	Уровень	Имя параметра	Пример значения параметра	Обязатель-ный	Описание
раздел	0	Convertors		+	Список конверторов.
раздел	1	ConverType	LibConverter	+	Тип конвертора.
параметр строка	2	ProgID	"ProjectEDrawings.EDrawings"	~1	ProgID (Для ActiveX библиотек).
параметр строка	2	Path	"S:\C++\Visualc\MyConverter.rtw"	~1	Путь к файлу конвертора (Для обычных dll или rtw).
параметр строка	2	Name	"Пример конвертора"	+	Имя конвертора.
параметр DWORD	2	Connect	1	-	Признак подключения библиотеки: 1 - библиотека подключается (по необходимости), 0 - библиотека в openfile и savefile не используется.
раздел	2	FileType	m3d	+	Тип документа системы КОМПАС (расширение имени файла).
параметр строка	3	Filter	"Файлы моделей (*.mym3d)|*.mym3d|"	+	Фильтр для выбора файла.
параметр DWORD	3	OpenCommandID	1	~2	Номер команды для открытия файла.
параметр DWORD	3	SaveCommandID	2	~2	Номер команды для сохранения файла.

параметр DWORD	3	OpenConverterParameters	1	-	Признак наличия диалога редактирования параметров конвертации для заданной команды открытия файла. Если параметр задан, в диалоге выбора файла для открытия появится кнопка **Параметры**, которая позволяет открыть диалог параметров конвертора. Вызов диалога осуществляется функцией IKompasConverter::VisualEditConvertParam.
параметр DWORD	3	SaveConverterParameters	1	-	Признак наличия диалога редактирования параметров конвертации для заданной команды сохранения файла. Если параметр задан, в диалоге выбора файла для открытия появится кнопка **Параметры**, которая позволяет открыть диалог параметров конвертора. Вызов диалога осуществляется функцией IKompasConverter::VisualEditConvertParam.

Значения параметра Обязательный:

\+ - обязательный параметр;

\- - необязательный параметр;

~1 - обязательно нужен один из параметров или ProgID или Path. Одновременно оба ключа задавать не нужно, первым проверяется наличие ключа ProgID.

~2 - OpenCommandID и SaveCommandID не являются обязательными, но нужен хотя бы один ключ. Можно задать оба ключа.

Пример файла реестра Windows для регистрации библиотеки (*.reg).

```
Windows Registry Editor Version 5.00
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter]
@=dword:00000000
"Path"="S:\\C++\\Visualc\\MyConverter.rtw"
"Name"="Пример конвертора"
"Connect"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\m3d]
@=dword:00000000
"Filter"="Файлы моделей(*.mym3d)|*.mym3d|"
"OpenCommandID"=dword:00000007
"SaveCommandID"=dword:00000001
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\a3d]
@=dword:00000000
"Filter"="Файлы сборок(*.mya3d)|*.mya3d|"
"OpenCommandID"=dword:00000008
"SaveCommandID"=dword:00000002
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\frw]
@=dword:00000000
"Filter"="Файлы фрагментов(*.myfrw)|*.myfrw|"
"OpenCommandID"=dword:00000009
"SaveCommandID"=dword:00000003
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\cdw]
@=dword:00000000
"Filter"="Файлы чертежей(*.mycdw)|*.mycdw|"
"OpenCommandID"=dword:0000000a
"SaveCommandID"=dword:00000004
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001
```

```
[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\spw]
@=dword:00000000
"Filter"="Файлы спецификаций(*.myspw)|*.myspw|"
"OpenCommandID"=dword:0000000b
"SaveCommandID"=dword:00000005
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\kdw]
@=dword:00000000
"Filter"="Файлы текстовых документов(*.mykdw)|*.mykdw|"
"OpenCommandID"=dword:0000000c
"SaveCommandID"=dword:00000006
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter]
@=dword:00000000
"Path"="S:\\C++\\Visualc\\MyConverter.rtw"
"Name"="Пример конвертора"
"Connect"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\m3d]
@=dword:00000000
"Filter"="Файлы моделей(*.mym3d)|*.mym3d|"
"OpenCommandID"=dword:00000007
"SaveCommandID"=dword:00000001
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001
```

Пример регистрации библиотеки-добавления

Приведен фрагмент файла экспорта реестра (reg - файла). В данном примере testAddins - название папки библиотеки-добавления. Регистрационная информация добавляется в предопределенный раздел реестра AddIns

```
Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\AddIns\testAddins]
```

"AutoConnect"=dword:00000000

"Path"="c:\\КОМПСАС\\libs\\newApiRtw\\teststep2.rtw"

"FriendlyName"="Test-Addins"

Значения флага AutoConnect = 0 не подключать 1 -подключать библиотеку на старте.

Пример регистрации библиотеки-конвертора

Приведен фрагмент файла экспорта реестра (reg - файла). В данном примере LibConverter - название папки конвертора. Регистрационная информация добавляется в предопределенный раздел реестра Converters.

Windows Registry Editor Version 5.00

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter]

@=dword:00000000

"Path"="S:\\C++\\Visualc\\MyConverter.rtw"

"Name"="Пример конвертора"

"Connect"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\m3d]

@=dword:00000000

"Filter"="Файлы моделей(*.mym3d)|*.mym3d|"

"OpenCommandID"=dword:00000007

"SaveCommandID"=dword:00000001

"OpenConverterParameters"=dword:00000001

"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\a3d]

@=dword:00000000

"Filter"="Файлы сборок(*.mya3d)|*.mya3d|"

"OpenCommandID"=dword:00000008

"SaveCommandID"=dword:00000002

"OpenConverterParameters"=dword:00000001

"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\frw]

@=dword:00000000

"Filter"="Файлы фрагментов(*.myfrw)|*.myfrw|"

"OpenCommandID"=dword:00000009

```
"SaveCommandID"=dword:00000003
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\cdw]
@=dword:00000000
"Filter"="Файлы чертежей(*.mycdw)|*.mycdw|"
"OpenCommandID"=dword:0000000a
"SaveCommandID"=dword:00000004
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\spw]
@=dword:00000000
"Filter"="Файлы спецификаций(*.myspw)|*.myspw|"
"OpenCommandID"=dword:0000000b
"SaveCommandID"=dword:00000005
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001

[HKEY_LOCAL_MACHINE\SOFTWARE\ASCON\KOMPAS-3D\Converters\LibConverter\kdw]
@=dword:00000000
"Filter"="Файлы текстовых документов(*.mykdw)|*.mykdw|"
"OpenCommandID"=dword:0000000c
"SaveCommandID"=dword:00000006
"OpenConverterParameters"=dword:00000001
"SaveConverterParameters"=dword:00000001
```

Пример использования функции LibCommandState

```
//------------------------------------------------------------------------------
// Состояние команды
// ---
int WINAPI LibCommandState( unsigned int comm, int * enable, int * checked )
{
  if ( enable )
  {
    int type = ksGetDocumentType( 0 );
```

```
        *enable = type == lt_DocSheetStandart || type == lt_DocFragment || type == lt_DocSheetUser; // Команда доступна в 2D документе
  }
  return 0;
}
```

Пример использования функции GetImageHeight

```
Public Function GetImageHeight() As Long
GetImageHeight = 100
End Function
```

Пример использования функции GetImageWidth

```
Public Function GetImageWidth() As Long
GetImageWidth = 100
End Function
```

Пример использования функции GetLibraryName

```
Public Function GetLibraryName() As String
GetLibraryName = "Самая простая библиотека"
End Function
```

Пример использования функции ExternalMenuItem

```
Public Function ExternalMenuItem(ByVal number As Integer, itemType As Integer, command As Integer) As String
itemType = 3 '"ENDMENU"'
ExternalMenuItem = ""
command = -1
Select Case number
Case 1
itemType = 1 'MENUITEM'
ExternalMenuItem = "Создать деталь"
command = 1
Case 2
itemType = 2 'POPUP'
ExternalMenuItem = "Выпадающее меню_1"
command = -1
Case 3
itemType = 1 'MENUITEM'
```

```
ExternalMenuItem = "Деталь с фасками"
command = 2
Case 4
itemType = 0 'SEPARATOR'
ExternalMenuItem = ""
command = -1
Case 5
itemType = 1 'MENUITEM'
ExternalMenuItem = "Команда_3"
command = 3
Case 6
itemType = 3 'ENDMENU'
ExternalMenuItem = ""
command = -1
Case 7
itemType = 2 '"POPUP"'
ExternalMenuItem = "Выпадающее меню_2"
command = -1
Case 8
itemType = 1 '"MENUITEM"'
ExternalMenuItem = "Команда_4"
command = 4
Case 9
itemType = 3 '"ENDMENU"'
ExternalMenuItem = ""
command = -1
End Select
End Function
```

Пример использования функции ExternalGetImage

```
Public Function ExternalGetImage(ByVal command As Integer, enableDelete As Integer) As OLE_HANDLE
Select Case command
Case 1
ExternalGetImage = Image1.Picture.Handle
Case 2
ExternalGetImage = Image2.Picture.Handle
Case 3
```

```
ExternalGetImage = Image3.Picture.Handle
Case 4
ExternalGetImage = Image4.Picture.Handle
Case Else
ExternalGetImage = Image5.Picture.Handle
End Select
enableDelete = FALSE
End Function
```

Пример использования функции ExternalRunCommand

```
Public Sub ExternalRunCommand(ByVal command As Integer, ByVal mode As Integer, ByVal
Kompas As Object)
Select Case command
Case 1
Dim document As Object'КОМПАС-документ
Set document = Kompas.Document3D'3D-документ
If Not document is Nothing Then
Kompas.ksMessage "Получен 3D-документ"
End If
Set document = Nothing'освободить документ
Case Else
MsgBox "Команда " + Str(command) + " не выполнена"
End Select
End Sub
```

Ошибка!

Отсутствует приложение, сопоставленное типу файла данного примера.

Ошибка!

Чтобы открыть список исправлений, необходимо установить Microsoft Excel.

Типы настроек для документов. Соответствие входных и выходных параметров

Типы настроек		Интерфейсы и структуры параметров	
		Интерфейсы	Структуры параметров
DIMENTION_OPTIONS	- настройки размера,	ksDimensionsOptions	DimensionsOptions

ARROWFILLING_OPTIONS	- признак зачернения стрелок,	ksLtVariant Может принимать значение, 0 - не зачернять или 1 - зачернять	int Может принимать значение, 0 - не зачернять или 1 - зачернять
SHEET_OPTIONS	- параметры листа; осталась для совместимости со старыми библиотеками.	ksSheetOptions	SheetPar
SHEET_OPTIONS_EX	- параметры листа активного документа, от текущего листа.	ksSheetOptions	SheetOptions
LENGTHUNITS_OPTIONS	- настройки единиц измерений	ksLtVariant Может принимать значение размерности длины, ST_MIX_SM, ST_MIX_M.	short Может принимать значение размерности длины, ST_MIX_SM, ST_MIX_M.
SNAP_OPTIONS_EX	- настройки привязок для активного документа	ksSnapOptions	SnapOptions
OVERLAP_OBJECT_OPTIONS	- настройки перекрывающихся объектов	ksOverlapObjectOptions	OverlapObjectOptions

Типы системных настроек.Соответствие входных и выходных параметров

Типы настроек		Интерфейсы и структуры параметров	
		Интерфейсы	Структуры параметров
SNAP_OPTIONS	- настройки привязок	ksSnapOptions	SnapOptions
SNAP_OPTIONS_EX	- настройки привязок для активного документа	ksSnapOptions	SnapOptions
SHEET_OPTIONS_EX	- параметры листа активного документа, от текущего листа.	ksSheetOptions	SheetPar
VIEWCOLOR_OPTIONS	- настройки цвета фона рабочего поля графических документов	ksViewColorParam	ViewColorParam
TEXTEDIT_VIEWCOLOR_OPTIONS	- настройки цвета фона текстовых документов	ksViewColorParam	ViewColorParam

MODEL_VIEWCOLOR_OPTIONS	- настройки цвета фона документов моделей	ksViewColorParam	ViewColorParam
LENGTHUNITS_OPTIONS	- настройки единиц измерений	ksLtVariant Может принимать значение размерности длины, ST_MIX_SM, ST_MIX_M.	short Может принимать значение размерности длины, ST_MIX_SM, ST_MIX_M.
OVERLAP_OBJECT_OPTIONS	- настройки перекрывающихся объектов	ksOverlapObjectOptions	OverlapObjectOptions
DIMENTION_OPTIONS_EX ARROWFILLING_OPTIONS	- настройки размера, - признак зачернения стрелок	ksDimensionsOptions ksLtVariant Может принимать значение, 0 - не зачернять или 1 - зачернять	DimensionsOptions int Может принимать значение, 0 - не зачернять или 1 - зачернять

Справочный файл плотностей

При расчетах вы можете не вводить значение плотности материала вручную, а выбрать его из справочного файла.

Этот файл называется graphic.dns и хранится в подпапке \sys главной папки КОМПАС-3D.

Файл graphic.dns записан в текстовом формате Windows (кодировка ANSI) и имеет простой синтаксис, что позволяет легко вносить в него исправления и дополнения.

Разделы в файле открываются и закрываются символами { и } соответственно. Название раздела должно находиться на той же строке, что и открывающая скобка. Разделы могут быть вложенными (то есть вы можете формировать древовидную структуру). Пробелы не являются значащими символами и не влияют на последующие действия отображение строк в справочном диалоге на экране.

Ниже приводится фрагмент файла graphic.dns, в котором описывается раздел Металлы, содержащий два подраздела - Алюминиевые сплавы и Бронзы.

```
{Металлы

{ Алюминиевые сплавы
  АД, АД1                 = 2.71
  Д1                      = 2.80
}
{ Бронзы
  Бр.АЖ9-4                = 7.50
  Бр.АЖМц10-3-1.5         = 7.50
  Бр.КМц3-1               = 8.40
}
```

Источник событий для подписки API 7

Для API интерфейсов версии 7 в Automation источником событий для подписки являются объекты:- IKompasDocument

В COM источником для подписки является указатель на документ.

Источник событий для подписки API 5

В Automation источником событий для подписки являются объекты:

- ksDocument2D (чертеж, фрагмент),
- ksSpcDocument (спецификация),
- ksDocumentTxt (текстовый документ),
- ksDocument3D (документ-модель).

В COM источником для подписки является указатель на документ.

Интерфейсы возвращаемые функцией ksDocument3D::GetInterface

Идентификатор объекта	Название объекта	Интерфейс
o3d_coordinate3dCollection	Интерфейс коллекции координат	ksCoordinate3dCollection
o3d_faceCollection	Интерфейс коллекции граней	ksFaceCollection

WM_KEYDOWN

Событие WM_KEYDOWN посылается в окно с клавиатурным фокусом, если была нажата несистемная клавиша.

Синтаксис:

WM_KEYDOWN nVirtKey = (int) wParam; KeyData = lParam;

Параметры:

wParam	- код виртуальной клавиши,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,
LParamHi	биты 0-7: scan-код клавиши, зависящий от OEM; бит 8: 1, если клавиша относится к расширенным, бит 13: 1, если при нажатии клавиши была нажата <*Alt*>, бит 14: 1, если клавиша была нажата до возникновения события, бит 15: 1, если после события клавиша отпущена, 0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Умолчательное действие:

Если нажата клавиша *<F10>*, функция DEFWINDOWPROC устанавливает внутренний флаг. Когда DEFWINDOWPROC принимает событие WM_KEYUP, она проверяет, установлен ли внутренний флаг, и если это так, посылает событие WM_SYSCOMMAND в окно верхнего уровня. Значение параметра wParam становится равным SC_KEYMENU.

Примечание:

1. Приложение должно вернуть 0, если оно обрабатывает это событие.
2. Для этого события биты 13 и 15 в lParamHi будут нулевыми.
3. Если ни одно из окон не имеет клавиатурного фокуса, то вместо событий WM_KEYDOWN, WM_CHAR и WM_KEYUP посылаются события WM_SYSKEYDOWN, WM_SYSCHAR и WM_SYSKEYUP.
4. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события WM_KEYUP может выдаваться несколько событий WM_KEYDOWN. Значение разряда 30 параметра flags позволяет определить, было ли событие WM_KEYDOWN первым или или является повторным во время удержания клавиши нажатой.
5. Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый *<ALT>* and *<CTRL>* на основной клавиатуре,
- *<INS>*, *<DEL>*, *<HOME>*, *<END>*, *<PAGE UP>*, *<PAGE DOWN>* и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,
- </> и *<ENTER>* на дополнительной цифровой клавиатуре.

Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam*.

См. также:

WM_DEADCHAR, WM_SYSCHAR и WM_SYSDEADCHAR.

WM_KEYUP

Событие WM_KEYDOWN посылается в окно с клавиатурным фокусом, если была отпущена несистемная клавиша.

Синтаксис:

WM_KEYUP nVirtKey = (int) wParam; lKeyData = lParam;

Параметры:

wParam	- код виртуальной клавиши,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,
LParamHi	биты 0-7: scan-код клавиши, зависящий от OEM;
	бит 8: 1, если клавиша относится к расширенным,
	бит 13: 1, если при нажатии клавиши была нажата *<Alt>*,
	бит 14: 1, если клавиша была нажата до возникновения события,
	бит 15: 1, если после события клавиша отпущена,
	0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Умолчательное действие:

Функция DEFWINDOWPROC посылает событие WM_SYSCOMMAND в окно верхнего уровня, если *<F10>* или *<ALT>* была отпущена. Значение параметра *wParam* становится равным SC_KEYMENU.

Примечание:

1. Приложение должно вернуть 0, если оно обрабатывает это событие.
2. Для этого события биты 13 и 15 в lParamHi будут нулевыми.
3. Если ни одно из окон не имеет клавиатурного фокуса, то вместо события WM_KEYUP посылается событие WM_SYSKEYUP.
4. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события WM_KEYUP может выдаваться несколько событий WM_KEYDOWN. Значение разряда 30 параметра flags позволяет определить, было ли событие WM_KEYDOWN первым или или является повторным во время удержания клавиши нажатой.
5. Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый *<ALT>* and *<CTRL>* на основной клавиатуре,
- *<INS>*, *<DEL>*, *<HOME>*, *<END>*, *<PAGE UP>*, *<PAGE DOWN>* и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,
- </> и *<ENTER>* на дополнительной цифровой клавиатуре.

Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam*.

WM_SYSKEYDOWN

Это событие посылается в окно с клавиатурным фокусом, когда клавиша нажимается при одновременно нажатой *<Alt>*. Также оно посылается, когда ни одно из окон не имеет клавиатурного фокуса; в этом случае WM_SYSKEYDOWN посылается в активное окно. Окно, которое принимает событие, может различить эти варианты проверкой разряда 29 (context code) параметра *lKeyData*.

Синтаксис:

WM_SYSKEYDOWN nVirtKey = (int) wParam; KeyData = lParam;

Параметры:

wParam	- код виртуальной клавиши,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,

lParamHi	биты 0-7: scan-код клавиши, зависящий от OEM; бит 8: 1, если клавиша относится к расширенным, бит 13: 1, если при нажатии клавиши была нажата <Alt>, бит 14: 1, если клавиша была нажата до возникновения события, бит 15: 1, если после события клавиша отпущена, 0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Умолчательное действие:

Функция DEFWINDOWPROC посылает событие WM_SYSCOMMAND в окно верхнего уровня, если F10 или *<Alt>* была отпущена. Значение параметра *wParam* становится равным SC_KEYMENU.

Примечание:

1. Приложение должно вернуть 0, если оно обрабатывает это событие.
2. Для этого события бит 15 в lParamHi будет нулевым.
3. Если бит 13 lParamHi (context code) параметра flags =0, это событие может проследовать для обработки функцией TRANSLATEACCELERATOR, которая обработает его как событие нажатия стандартной клавиши, вместо системной. Это позволяет использовать горячие клавиши в активном окне, даже если это окно не имеет клавиатурного фокуса.
4. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события WM_KEYUP может выдаваться несколько событий WM_KEYDOWN. Значение разряда 30 параметра flags позволяет определить, было ли событие WM_KEYDOWN первым или или является повторным во время удержания клавиши нажатой.
5. Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый *<ALT>* и *<CTRL>* на основной клавиатуре,
- *<INS>*, *<DEL>*, *<HOME>*, *<END>*, *<PAGE UP>*, *<PAGE DOWN>* и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,
- </> и *<ENTER>* на дополнительной цифровой клавиатуре.

Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam*.

6. Это событие также генерируется, если нажимается *<F10>* без нажатой *<Alt>*.

WM_SYSKEYUP

Это событие посылается в окно с клавиатурным фокусом, когда отпускается клавиша, которая была нажата при одновременно нажатой *<Alt>*. Также оно посылается, когда ни одно из окон не имеет клавиатурного фокуса; в этом случае WM_SYSKEYDOWN посылается в активное окно. Окно, которое принимает событие, может различить эти варианты проверкой разряда 29 (context code) параметра *lKeyData*.

Синтаксис::

WM_SYSKEYUP nVirtKey = (int) wParam; lKeyData = lParam;

Параметры:

wParam	- код виртуальной клавиши,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,
lParamHi	биты 0-7: scan-код клавиши, зависящий от OEM;
	бит 8: 1, если клавиша относится к расширенным,
	бит 13: 1, если при нажатии клавиши была нажата *<Alt>*,
	бит 14: 1, если клавиша была нажата до возникновения события,
	бит 15: 1, если после события клавиша отпущена,
	0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Умолчательное действие:

Функция DEFWINDOWPROC посылает событие WM_SYSCOMMAND в окно верхнего уровня, если <F10>или <Alt> была отпущена. Значение параметра wParam становится равным SC_KEYMENU.

Примечание:

1. Приложение должно вернуть 0, если оно обрабатывает это событие.
2. Для этого события бит 15 в lParamHi будет нулевым.
3. Если бит 13 lParamHi (context code) параметра flags =0, это событие может проследовать для обработки функцией TranslateAccelerator, которая обработает его как событие нажатия стандартной клавиши, вместо системной. Это позволяет использовать горячие клавиши в активном окне, даже если это окно не имеет клавиатурного фокуса.
4. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события WM_KEYUP может выдаваться несколько событий WM_KEYDOWN. Значение разряда 30 параметра flags позволяет определить, было ли событие WM_KEYDOWN первым или или является повторным во время удержания клавиши нажатой.
5. Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый *<ALT>* и *<CTRL>* на основной клавиатуре,
- *<INS>*, *<DEL>*, *<HOME>*, *<END>*, *<PAGE UP>*, *<PAGE DOWN>* и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,
- </> и *<ENTER>* на дополнительной цифровой клавиатуре.

Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam.*

6. Это событие также генерируется, если нажимается *<F10>* без нажатой *<Alt>*.

WM_SYSCHAR

Это событие посылается в окно с клавиатурным фокусом, когда событие WM_SYSKEYDOWN обрабатывается функцией TRANSLATEMESSAGE. Оно позволяет по-

лучить код символа нажатой системной клавиши, то есть клавиши, которая была нажата при нажатой *<ALT>*.

Синтаксис::

WM_SYSCHAR chCharCode = (TCHAR) wParam;KeyData = lParam;

Параметры:

wParam	- код клавиши вызова меню окна,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,
lParamHi	биты 0-7: scan-код клавиши, зависящий от OEM;
	бит 8: 1, если клавиша относится к расширенным,
	бит 13: 1, если при нажатии клавиши была нажата *<Alt>*,
	бит 14: 1, если клавиша была нажата до возникновения события,
	бит 15: 1, если после события клавиша отпущена,
	0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Умолчательное действие:

Функция DEFWINDOWPROC посылает событие WM_SYSCOMMAND в окно верхнего уровня если <F10> или <Alt> была отпущена. Значение параметра wParam становится равным SC_KEYMENU.

Примечание:

1. Приложение должно вернуть 0, если оно обрабатывает это событие.
2. Для этого события бит 15 в lParamHi будет нулевым.
3. Если бит 13 lParamHi (context code) параметра flags =0, это событие может проследовать для обработки функцией TRANSLATEACCELERATOR, которая обработает его как событие нажатия стандартной клавиши, вместо системной. Это позволяет использовать горячие клавиши в активном окне, даже если это окно не имеет клавиатурного фокуса.
4. Благодаря автоматическому повторению кода клавиши при удержанию ее нажатой, до появления события WM_KEYUP может выдаваться несколько событий WM_KEYDOWN. Значение разряда 30 параметра flags позволяет определить, было ли событие WM_KEYDOWN первым или или является повторным во время удержания клавиши нажатой.
5. Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый *<ALT>* и *<CTRL>* на основной клавиатуре,
- *<INS>*, *<DEL>*, *<HOME>*, *<END>*, *<PAGE UP>*, *<PAGE DOWN>* и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,
- </> и *<ENTER>* на дополнительной цифровой клавиатуре.

Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam*.

6. Это событие также генерируется, если нажимается *<F10>* без нажатой *<Alt>*.

7. Событие генерируется вместо WM_KEYUP, если нет окна с клавиатурным фокусом.

   См. также:

   WM_DEADCHAR, WM_CHAR и WM_SYSDEADCHAR.

WM_SYSDEADCHAR

Это событие посылается в окно с клавиатурным фокусом, когда событие WM_SYSKEYDOWN обрабатывается функцией TRANSLATEMESSAGE. Оно позволяет получить код символа нажатой пассивной системной клавиши, то есть пассивной клавиши, которая была нажата при нажатой <*ALT*>.

Синтаксис:

WM_SYSDEADCHAR chCharCode = (TCHAR) wParam; lKeyData = lParam;

Параметры:

wParam	- код клавиши,
lParamLo	- количество повторений кода клавиши за время удержания ее в нажатом состоянии,
lParamHi	биты 0-7: scan-код клавиши, зависящий от OEM; бит 8: 1, если клавиша относится к расширенным, бит 13: 1, если при нажатии клавиши была нажата <*Alt*>, бит 14: 1, если клавиша была нажата до возникновения события, бит 15: 1, если после события клавиша отпущена, 0, если после события клавиша остается нажатой.

Возвращаемое значение:

Не используется.

Примечание:

1. К пассивным клавишам относятся, например, умляуты и ударения.
2. Событие может использоваться для получения обратной связи для клавиш, нажатие которых необязательно формирует символ как таковой.

   См. также:

   WM_DEADCHAR, WM_CHAR и WM_SYSDEADCHAR.

Несистемная клавиша

Несистемной клавишей называется любая клавиша, нажимаемая без одновременного нажатия клавиши <*Alt*>.

Расширенные клавиши

Для расширенных 101 и 102 клавишных клавиатур расширенными клавишами являются следующие:

- правый <*ALT*> и <*CTRL*> на основной клавиатуре,
- <*INS*>, <*DEL*>, <*HOME*>, <*END*>, <*PAGE UP*>, <*PAGE DOWN*> и клавиши со стрелками в группах слева от дополнительной цифровой клавиатуры,

- </> и <*ENTER*> на дополнительной цифровой клавиатуре.

  Другие клавиатуры могут поддерживать бит расширенной клавиатуры при помощи параметра *lParam*.

sysKey

Параметр sysKey позволяет получить состояние клавиши <*Alt*> при нажатии клавиши.

Значения параметра:

TRUE	- была нажата клавиша <*Alt*>, что соответствует событию WM_SYSKEYDOWN,
FALSE	- не была нажата клавиша <*Alt*>, что соответствует событию WM_KEYDOWN.

Флаг клавиатуры

Параметр соответствует lParam в событии WM_KEYDOWN или WM_SYSKEYDOWN.

Значения разрядов параметра:

0 –15	- количество повторений выдаваемого кода клавиши за время ее удержания в нажатом состоянии.
29	- признак нажатия клавиши <*Alt*>. 1 - клавиша нажата, 0 - клавиша не нажата.
30	- предыдущее состояние клавиши; 1 - клавиша была нажата, 0 - клавиша не была нажата.
31	- состояние перехода; 1 - клавиша отпускается, 0 -клавиша остается нажатой.

Интерфейсы элементов модели

- Интерфейсы формообразующих операций
- Интерфейсы пространственных кривых
- Интерфейсы поверхностей Интерфейсы копирования компонентов сборкиИнтерфейсы копирования
- Интерфейсы вспомогательной геометрии

Интерфейсы дополнительных элементовИнтерфейсы поверхностей

- Интерфейс параметров кинематической поверхности ksEvolutionSurfaceDefinition или IEvolutionSurfaceDefinition.
- Интерфейс параметров поверхности выдавливания ksExtrusionSurfaceDefinition или IExtrusionSurfaceDefinition.
- Интерфейс параметров поверхности вращения ksRotatedSurfaceDefinition или IRotatedSurfaceDefinition.

- Интерфейс параметров поверхности по сечениям ksLoftSurfaceDefinition или ILoftSurfaceDefinition.
- Интерфейс параметров макроэлемента документа-модели ksMacro3DDefinition или IMacro3DDefinition.
- Интерфейс операции объединения компонентов ksUnionComponentsDefinition или IUnionComponentsDefinition.
- Интерфейс операции вычитания компонентов ksMoldCavityDefinition или IMoldCavityDefinition.

Интерфейсы пространственных кривых

- Интерфейс параметров цилиндрической спирали ksCylindricSpiralDefinition или ICylindricSpiralDefinition.
- Интерфейс параметров конической спирали ksConicSpiralDefinition или IConicSpiralDefinition.
- Интерфейс параметров сплайна ksSplineDefinition или ISplineDefinition.
- Интерфейс параметров ломаной ksPolyLineDefinition или IPolygonalLineDefinition.

Интерфейсы копирования

- Интерфейс массива удаленных индексов для операций копирования и массивов компонентов ksDeletedCopyCollection или IDeletedCopyCollection.
- Интерфейс операции копирования по сетке ksMeshCopyDefinition или IMeshCopyDefinition.
- Интерфейс операции копирования по кривой ksCurveCopyDefinition или ICurveCopyDefinition.
- Интерфейс операции копирования по окружности ksCircularCopyDefinition или ICircularCopyDefinition.
- Интерфейс зеркальной копии ksMirrorCopyDefinition или IMirrorDefinition.
- Интерфейс зеркального отражения всех элементов ksMirrorCopyAllDefinition или IMirrorAllDefinition.

Интерфейсы вспомогательной геометрии

- Интерфейс параметров объекта "Контрольная точка" ksControlPointDefinition или IControlPointDefinition.
- Интерфейс параметров объекта "Присоединительная точка" ksConjunctivePointDefinition или IConjunctivePointDefinition.

Вспомогательные плоскости:

- Интерфейс параметров плоскости, проходящей через ребро и вершину ksPlaneEdgePointDefinition или IPlaneEdgePointDefinition.
- Интерфейс параметров смещенной плоскости ksPlaneOffsetDefinition или IPlaneOffsetDefinition.
- - Интерфейс параметров плоскости, проходящей через ребро параллельно или перпендикулярно грани ksPlaneLineToPlaneDefinition или IPlaneLineToPlaneDefinition.

- Интерфейс параметров вспомогательной плоскости, проходящей через три точки ksPlane3PointsDefinition или IPlane3PointsDefinition.
- Интерфейс параметров нормальной плоскости ksPlaneNormalToSurfaceDefinition или IPlaneNormalToSurfaceDefinition.
- Интерфейс параметров плоскости, проходящей через вершину перпендикулярно ребру ksPlanePerpendicularDefinition или IPlanePerpendicularDefinition.
- Интерфейс параметров плоскости, проходящей через ребро параллельно или перпендикулярно другому ребру ksPlaneLineToEdgeDefinition или IPlaneLineToEdgeDefinition.
- Интерфейс параметров вспомогательной плоскости, построенной под углом к другой плоскости и проходящей через заданную ось или ребро ksPlaneAngleDefinition или IPlaneAngleDefinition.
- Интерфейс параметров плоскости, проходящей через вершину параллельно другой плоскости ksPlaneParallelDefinition или IPlaneParallelDefinition.
- Интерфейс параметров касательной плоскости ksPlaneTangentToSurfaceDefinition или IPlaneTangentToSurfaceDefinition.
- Интерфейс параметров конструктивной плоскости "Средняя плоскость" ksPlaneMiddleDefinition или IPlaneMiddleDefinition.

Вспомогательные оси:

- Интерфейс параметров вспомогательной оси, проходящей через прямолинейное ребро ksAxisEdgeDefinition или IAxisEdgeDefinition.
- Интерфейс параметров вспомогательной оси, проходящей через две точки ksAxis2PointsDefinition или IAxis2PointsDefinition.
- Интерфейс параметров вспомогательной оси на пересечении двух плоскостей ksAxis2PlanesDefinition или IAxis2PlanesDefinition.
- Интерфейс параметров оси формообразующего элемента ksAxisOperationsDefinition или IAxisOperationsDefinition.
- Интерфейс параметров конструктивной оси конической грани ksAxisConefaceDefinition или IAxisConefaceDefinition.

Интерфейсы копирования компонентов сборки

- Интерфейс операции копирования компонентов сборки по параллелограммной сетке ksMeshPartArrayDefinition или IMeshPartArrayDefinition.
- Интерфейс копирования компонентов сборки по концентрической сетке ksCircularPartArrayDefinition или ICircularPartArrayDefinition.
- Интерфейс операции копирования компонентов сборки по кривой ksCurvePartArrayDefinition или ICurvePartArrayDefinition.
- Интерфейс копирования компонентов сборки по образцу ksDerivativePartArrayDefinition или IDerivativePartArrayDefinition.

Интерфейсы формообразующих операций

- Интерфейс параметров основания - элемента выдавливания ksBaseExtrusionDefinition или IBaseExtrusionDefinition.

- Интерфейс параметров основания - элемента выдавливания ksBossExtrusionDefinition или IBossExtrusionDefinition.
- Интерфейс параметров вырезанного элемента выдавливания ksCutExtrusionDefinition или ICutExtrusionDefinition.
- Интерфейс параметров основания - элемента вращения ksBaseRotatedDefinition или IBaseRotatedDefinition.
- Интерфейс приклеенного элемента вращения ksBossRotatedDefinition или IBossRotatedDefinition.
- Интерфейс вырезанного элемента вращения ksCutRotatedDefinition или ICutRotatedDefinition.
- Интерфейс параметров основания - элемента по сечениям ksBaseLoftDefinition или IBaseLoftDefinition.
- Интерфейс приклеенного элемента по сечениям ksBossLoftDefinition или IBossLoftDefinition.
- Интерфейс параметров вырезанного элемента по сечениям ksCutLoftDefinition или ICutLoftDefinition.
- Интерфейс параметров основания - кинематического элемента ksBaseEvolutionDefinition или IBaseEvolutionDefinition.
- Интерфейс приклеенного кинематического элемента ksBossEvolutionDefinition или IBossEvolutionDefinition.
- Интерфейс вырезанного кинематического элемента ksCutEvolutionDefinition или ICutEvolutionDefinition.
- Интерфейс тонкостенной оболочки ksShellDefinition или IShellDefinition.
- Интерфейс ребра жесткости ksRibDefinition или IRibDefinition.
- Интерфейс уклона ksInclineDefinition или IInclineDefinition.
- Интерфейс параметров фаски ksChamferDefinition или IChamferDefinition.
- Интерфейс параметров элемента "скругление" ksFilletDefinition или IFilletDefinition.
- Интерфейс операции сечения эскизом ksCutBySketchDefinition или ICutBySketchDefinition.
- Интерфейс операции сечения плоскостью ksCutByPlaneDefinition или ICutByPlaneDefinition.

Интерфейсы дополнительных элементов

- Интерфейс описания вершины ksVertexDefinition илиIVertexDefinition.
- Интерфейс свойств ребра ksEdgeDefinition или IEdgeDefinition.
- Интерфейс свойств грани ksFaceDefinition или IFaceDefinition.
- Интерфейс параметров импортированной поверхности ksImportedSurfaceDefinition или IImportedSurfaceDefinition.
- Интерфейс условного изображения резьбы ksThreadDefinition или IThreadDefinition.
- Интерфейс параметров эскиза ksSketchDefinition или ISketchDefinition

Элементы, существующие в модели по умолчанию

o3d_planeXOY	= 1	плоскость XOY
o3d_planeXOZ	= 2	плоскость XOZ
o3d_planeYOZ	= 3	плоскость YOZ
o3d_pointCS	= 4	точка начала системы координат

Пример использования функций работы с документами

```
void Document_Example (void) {

reference pDoc;
DocumentParam   doc, doc1;
char buf[ 128 ];

lstrcpy(doc.fileName,"c:\\gr\\2.cad");
lstrcpy(doc.comment , "Чертеж" );
lstrcpy(doc.author ,"Иванов");
doc.regim=0;  //видимый режим
doc.type=1; doc.stPar.Toleranceat=3; doc.stPar.multiply=1;
doc.stPar.direct=0; doc.stPar.shtType=1;

pDoc = CreateDocument (&doc);

// создадим вид  с номером 2

ViewParam par;
int number = 2;
par.x = 10; par.y = 20; par.scale = 2; par.ang = 45;
par.color = RGB(10,20,10);
par.state = stACTIVE;  // вид будет активным
strcpy(par.name, "пользовательский вид");
CreateSheetView(&par, &number);

//создать и перейти в слой 5

Layer(5);

LineSeg(20, 10, 40, 10, 1);
LineSeg(40, 10, 40, 30, 1);
```

```
LineSeg(40, 30,  20, 30, 1);
LineSeg(20, 30,  20, 10, 1);

SaveDocument (pDoc, );  //сохранить документ
CloseDocument(pDoc);      //закрыть документ

//откроем созданный документ «c:\gr\2.cad» в видимом режиме отображения

OpenDocument( "c:\gr\2.cad", 0);

}; /* Document_Example */
```

Пример использования функций работы с документами, Unicode

```
void Document_Example (void) {

reference pDoc;
DocumentParamW doc, doc1;
char buf[ 128 ];

wcscpy( doc.fileName, _T("c:\\gr\\2.cad"));
wcscpy( doc.comment, _T("Чертеж"));
wcscpy( doc.author, _T("Иванов"));
doc.regim=0;  //видимый режим
doc.type=1; doc.stPar.Toleranceat=3; doc.stPar.multiply=1;
doc.stPar.direct=0; doc.stPar.shtType=1;

pDoc = CreateDocumentW (&doc);

// создадим вид  с номером 2

ViewParamW par;
int number = 2;
par.x = 10; par.y = 20; par.scale = 2; par.ang = 45;
par.color = RGB(10,20,10);
par.state = stACTIVE;  // вид будет активным
wcscpy(par.name,_T(" пользовательский вид"));
CreateSheetViewW(&par, &number);
```

```
//создать и перейти в слой 5

Layer(5);

LineSeg(20, 10, 40, 10, 1);
LineSeg(40, 10, 40, 30, 1);
LineSeg(40, 30,  20, 30, 1);
LineSeg(20, 30,  20, 10, 1);

SaveDocument( pDoc, 0 );  //сохранить документ
CloseDocument( pDoc );      //закрыть документ

//откроем созданный документ «c:\gr\2.cad» в видимом режиме отображения

OpenDocument( "c:\gr\2.cad", 0);

}; /* Document_Example */
```

Пример инсталляции шрифта хот-точек в Компас

```
class FontInstaller
{
  HANDLE    m_hFont  = {};
  public:
  FontInstaller( HINSTANCE module, LPWSTR lpName );
  ~FontInstaller();
};

//--------------------------------------------------------------------------------
///
// ---
FontInstaller::FontInstaller( HINSTANCE module, LPWSTR lpName )
{
  PRECONDITION( module );
  HRSRC hrc = ::FindResource( module, lpName, RT_RCDATA );
  PRECONDITION( hrc );

  DWORD dwrclen = ::SizeofResource( module, hrc );
  PRECONDITION( dwrclen );
```

```
  HGLOBAL hrcglobal = ::LoadResource( module, hrc );
  PRECONDITION( hrcglobal );

  void * pvrc = ::LockResource( hrcglobal );
  PRECONDITION( pvrc );

  DWORD fontsCount = 0;
  m_hFont = ::AddFontMemResourceEx( pvrc, dwrclen, 0, &fontsCount );

  PRECONDITION( m_hFont && fontsCount == 1 );
}

//---------------------------------------------------------------------------
///
// ---
FontInstaller::~FontInstaller()
{
  ::RemoveFontMemResourceEx( m_hFont );
}


////////////////////////////////////////////////////////////////////////////
//
// Класс устанавливает хот-точечный шрифт.
//
////////////////////////////////////////////////////////////////////////////
class HotPointsFontInstaller : public FontInstaller
{
public:
  HotPointsFontInstaller()
     : FontInstaller( module, MAKEINTRESOURCE( HOT_POINTS_FONT ) ){};
};
```

Библиотека по первому требованию должна зарегистрировать шрифт и разрегистрировать его при отключении библиотеки.

GetDocOptions - пример использования

```
DimensionsOptrions par;
memset (&par, 0, sizeof (DimensionsOptrions));
if (GetDocOptions (DIMENTION_OPTIONS, &par, sizeof(DimensionsOptrions)))
{
char buf[128];
Message("Настройки размеров текущего документа");
sprintf (buf, "Выход выносных линий за размерную=%0.1f;
Расстояние от разм.линии до текста=%0.1f;",
par.proLineExtension, par.textDistanceFromDimLine);
Message(buf);
sprintf (buf, "Расстояние от выносных линий до текста=%0.1f;
Выход размерной линии за текст=%0.1f;",
par.textDistanceFromProLine, par.dimLineExtension);
Message(buf);
sprintf (buf, "Длина стрелки для размера=%0.1f; Кол-во знаков после запятой=%d;",
par.arrowLength, par.decimalsCount);
Message(buf);
sprintf (buf, "Стиль текста=%d; точность углового размера=%s", par.style,
!par.anglePrecisionLevel ? "градусы" : par.anglePrecisionLevel ==1 ? "минуты" :
"секунды");
Message(buf);
}
```

ksPrintPreviewWindow - пример использования

```
char buf[128];

  //создать динамический массив строк
  reference docsArr = CreateArray(CHAR_STR_ARR,0);

  //наполним массив
  strcpy(buf,"d:\\0\\_222.cdw");
  AddArrayItem(docsArr, -1, buf, sizeof(buf));
  strcpy(buf,"d:\\0\\_2.cdw");
  AddArrayItem(docsArr, -1, buf, sizeof(buf));

  //в просмотровом окне должно появиться два документа, которые нужно распечатать
```

```
      if (! ksPrintPreviewWindow(docsArr,        // динамический массив указателей
CHAR_STR_ARR
                                 1)) {   // 1 - если docsArr = 0 или массив пуст,
// запросить документы у пользователя
                                       // 0 - без запроса


  // если неудачное завершение - выдадим результат работы нашей функции
  MessageBoxResult();
```

ksSheetSetupDlg - пример использования

```
DocumentParam par;
    par.type = lt_DocSheetStandart;
    par.sheet.stPar.format   = 4;
    par.sheet.stPar.multiply = 1;
    par.sheet.stPar.direct   = 0;
    if ( ::ksSheetSetupDlg(&par, ::GetHWindow()) ) {
     char buf[256];
      if ( par.type == lt_DocSheetStandart ) {
        ::sprintf( buf, "Стандартный лист : формат A%d, кратность %d, расположение %s",
par.sheet.stPar.format,
          par.sheet.stPar.multiply, par.sheet.stPar.direct ? "горизонтально" : "вертикально" );
       }
       else {
                  ::sprintf( buf, "Пользовательский формат : Длина %g, Высота %g",
par.sheet.usPar.width,
          par.sheet.usPar.height );
       }
     ::Message( buf );
    }
```

ksReDrawDocPart - пример использования

```
reference pView = OpenView ( 0 );    // Системный вид текущего документа
 RectParam rect;                  // Структура параметров прямоугольника
 rect.pBot.x = 100;               // Инициализация структуры pBot низ - лево
 rect.pBot.y = 100;               //                    pTop верх - право
 rect.pTop.x = 200;
 rect.pTop.y = 200;
 ksReDrawDocPart( &rect, pView );     // Будет перерисована область во всех граф. окнах
документа
```

SystemControlStart - Пример использования

```
void SystemControlStart_Example (void)
{
//построить заштрихованный квадрат
LineSeg (20, 30, 70, 30, 1);
LineSeg (70, 30, 70, 80, 1);
LineSeg (70, 80, 20, 80, 1);
LineSeg (20, 80, 20, 30, 1);

//штриховка появится только после возвращения управления библиотеке
SystemControlStart (0);

reference p = Hatch(0, 45, 2, 0, 0, 0);
LineSeg (20, 30, 70, 30, 1);
LineSeg (70, 30, 70, 80, 1);
LineSeg (70, 80, 20, 80, 1);
LineSeg (20, 80, 20, 30, 1);
EndObj();
};
```

SystemControlStop - Пример использования

```
class TEskWin : public TDialog
{
public:
TEskWin(TWindow *parent);
protected:
void EvDialogCommand1();
void EvClose();
DECLARE_RESPONSE_TABLE(TEskWin);
void TEskWin::EvClose() {
TDialog::EvClose();
SystemControlStop();
}

//точка входа в библиотеку
extern "C" void far __export pascal LibraryEntry (unsigned int)
{
TWindow *parent = GetWindowPtr((HWND)GetHWindow());
```

```
if (parent)
{
TEskWin *eskw = new TEskWin(parent);
eskw->Create(); //немодальный диалог
SystemControlStart("Остановить ESK");
//в это место библиотеки вернемся только после выполнения SystemControlStop();
delete eskw;
}
};
```

ksOpenHelpFile - Пример использования

```
//вызываем раздел помощи, идентификатор которого равен номеру команды
  ::ksOpenHelpFile ("profile.hlp", HELP_CONTEXT, 1);
```

ksExecuteLibraryCommand - Пример использования

```
char fileName[255];
   ::strcpy(fileName, "C:\\libtest\\libtest.rtw");    // имя файла библиотеки
   ksExecuteLibraryCommand(fileName, 3/*номер команды*/);
// выполнение команды 3 из библиотеки
```

ksSetCurrentLibrary - Пример использования

```
if (!::ksSetCurrentLibrary( ::LibraryName(), 0 ) ) { //Имя библиотеки
 Error("Нужно завершить текущую операцию");
 return;
}
```

SetMacroParam - пример использования

```
// пример построения прямоугольника с последующим его редактированием

void DrawRect ()
{
struct
{
double a;
double b;
}
r;
```

```
if (EditMacroMode())
//режим редактирования
GetMacroParam (m, &r, sizeof (r));
else
//создание нового макроэлемента
{
r.a = 20;
r.b = 10;
}
ReadDouble ("задайте ширину",r.a,0,100, &r.a);
ReadDouble ("задайте высоту",r.b,0,100, &r.b);
Macro(); /* определение макроэлемента */
// Прямоугольник
LineSeg(0, 0, r.a, 0, 0);
LineSeg(r.a, 0, r.a, r.b, 0);
LineSeg(r.a, r.b, 0, r.b, 0);
LineSeg(0, r.b, 0, 0, 0);
reference m = EndObj();

//записать параметры в макроэлемент
SetMacroParam (m, &r, sizeof(r), NULL, NULL, -1);
};
```

ksGetMacroEditParam - пример использования

```
Macro (0);
   Point (10, 10, 1);
  reference ref = EndObj ();

  SetMacroParam (ref, NULL, 0, "FileName.rtw", "LibName", 111);

  char        fileName[128];                  // буфер имени файла библиотеки
  unsigned int fileNameSize = sizeof (fileName); // размер буфера имени файла библиотеки
  char        libName[128];                   // буфер имени библиотеки
  unsigned int libNameSize = sizeof (libName); // размер буфера имени библиотеки
  int         number;                         // буфер номера функции редактирования

  int res = ksGetMacroEditParam (ref, fileName, fileNameSize, libName,
                                 libNameSize, &number);
```

```
 if (res) {
  char buf[128];
  sprintf (buf, " Имя файла: %s\n Имя библиотеки: %s\n Номер функции: %i",
        fileName, libName, number);

  Message (buf);
 }
 else
  Error ("Ошибка");
```

GetMacroParamSize - пример использования

```
reference p;
 RequestInfo  info;
 //обнулить структуру  info;
 memset (&info, 0, sizeof (info));
 double x, y;
 info.prompt = "Укажите макроэлемент";
 int j = Cursor (&info, &x ,&y, 0);
 if (j) {
if (ExistObj (p = FindObj (x, y, 1e6)) && GetObjParam (p, 0, 0, 0) == MACRO_OBJ) {
int size = GetMacroParamSize (p);
char buf [128];
sprintf (buf, "Размер параметров макроэлемента = %d", size);
Message (buf);
 }
 }
```

ksOpenMacro - пример использования

```
Macro (0);
LineSeg(50,100, 100, 150, 1);
reference p = EndObj();

Message("Создан макроэлемент");

ksOpenMacro (p);
Macro (0);
reference cir = Circle(50, 50, 20, 1);
```

```
reference p1 = EndObj();
EndObj();

Message("Макроэлемент отредактирован");
```

ksOpenMacro - пример использования

```
 //самостоятельный объект вида
 reference  l = LineSeg(10, 10, 20, 20, 1);

 Macro(0);
  //внутренний объект макроэлемента
  reference l1 = LineSeg(50,100, 100, 150, 1);
 reference p = EndObj();

 ksOpenMacro (p);
  //внутренний объет макроэлемента
  Macro(0);
    reference cir = Circle(50, 50, 20, 1);
  reference p1 = EndObj();
 EndObj();
```

ksOpenMacro - пример использования

```
 Macro(0);
 reference p = EndObj();
 ksOpenMacro (p);
  //внутренний объет макроэлемента
  Macro(0);
    reference cir = Circle(50, 50, 20, 1);
  reference p1 = EndObj();
 EndObj();
```

ksAddObjectToMacro - пример использования

```
//создать постоянную группу
//(окружности принадлежат текущему виду)
reference  gr = NewGroup (0);
Circle(0, 0, 100, 1);
Circle(0, 0, 50, 1);
EndGroup();
```

```
//создать макроэлемент
Macro(1);
LineSeg (120, 100, 120, 150, 1);
reference m = EndObj();

//добавить в макро новый объект
ksOpenMacro(m);
LineSeg (100, 100, 150, 100, 1);
::EndObj();

//добавить в макроэлемент группу самостоятельных объектов.
//(объекты становятся внутренними)
```

ksDuplicateBoundaries - пример использования

```
reference pObj;
  RequestInfo  info;
  double x, y;
  memset (&info, 0, sizeof (info));
  info.prompt = "Укажите штриховку или заливку";
  int j = Cursor(&info, &x ,&y, 0);
  if (j) {
    if(ExistObj(pObj = FindObj(x, y, 1e6))) {
      int objType =  GetObjParam(pObj, 0, 0, 0);
      if (objType == HATCH_OBJ || objType == COLORFILL_OBJ) {
        Phantom phantom;     // ltdefine.h
        memset(&phantom, 0, sizeof(phantom));
        //в фантом положим копию границы
        phantom.type1.gr = ksDuplicateBoundaries( pObj);
        phantom.phType = 1; //сдвиг группы
        phantom.type1.scale = 1; //сдвиг группы

        info.prompt = "Укажите точку привязки границы";
        j = Cursor(&info, &x ,&y, &phantom);
        if (j) {
          MoveObj(phantom.type1.gr, x, y); //смещаем группу в новый   центр
          StoreTmpGroup(phantom.type1.gr); //временную группу делаем постоянной
          ClearGroup(phantom.type1.gr);
```

```
        DeleteObj(phantom.type1.gr);
      }
    }
    else
      Error("Объект должен быть штриховкой или заливкой");
  }
}
```

Macro, EndObj - пример использования

```
void Macro_Example (void) {

Macro(); /* определение макроэлемента */

LineSeg (10,  10,  10,  20,  1);
LineSeg (10,  20,   40, 20,  1);
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

Macro (); /* вложенный макроэлемент */
LineSeg (40,  15,   50, 25,  1);
LineSeg (50,  25,   60, 15,  1);
LineSeg (60,  15,   40, 15,  1);
EndObj();

EndObj();   /* закончить формирование макроэлемента */

}; /* Macro_Example */
```

Contour - пример использования

```
void Contour_Example (void) {

reference p;

Contour(1); /* определение контура */

LineSeg (10,  10,  10,  20,  1);
```

```
LineSeg (10,  20,   40, 20,  1);
LineSeg (40,  20,   40, 30,  1);
LineSeg (40,  30,   70, 30,  1);
LineSeg (70,  30,   70, 10,  1);
LineSeg (70,  10,   10, 10,  1);

p = EndObj();   /* закончить формирование контура */

}; /* Contour_Example */
```

CommandWindow - пример использования

```
void CommandWindow_Example (void) {
RequestInfo  info;
  memset(&info, 0, sizeof(info));
info.commands = "!Окружность !Отрезок ";
info.commands = Объекты

int j=CommandWindow(&info);
  switch (j) {
     case 1:
       Circle(10,10,10,1);
       break;
     case 2:
       LineSeg(10,10, 20, 10, 1);
       break;
  }

}; /* CommandWindow_Example */
```

ksCalculate, ksCalculateReset - пример использования

```
//очистить массив переменных
  ksCalculateReset ();
  char buf [128];
  double resD;

  //добавить переменную A1 = 100
```

```
int  rezl =  ksCalculate ("A1 = 100", &resD);
sprintf (buf,"resD = %f rezl = %d", resD, rezl);
Message (buf);

//добавить переменную A2 = 200
rezl = ksCalculate ("A2 = 200", &resD);
sprintf (buf,"resD = %f rezl = %d", resD, rezl);
Message (buf);

//подсчитать результат выражения
rezl = ksCalculate ("(A1 A2) * sqrt(4)", &resD);
sprintf (buf,"resD = %f rezl = %d", resD, rezl);
Message (buf);
```

Представление математических объектов, участвующих в сопряжении

ksMatePoint Точка	Pc - координаты точки, V, radius1, radius2 - не используются.
ksMateLine Линия	Pc - координата точки на прямой, V - координаты вектора направления прямой, radius1, radius2 - не используются.
ksMatePlane Плоскость	Pc - координата точки на плоскости, V - координаты вектора нормали плоскости, radius1, radius2 - не используются.
ksMateCylinder Цилиндр	Pc - координата точки на оси цилиндра, V - направление оси цилиндра, radius1 - радиус цилиндра, radius2 = radius1.
ksMateCone Конус	Pc - координата точки на оси конуса, лежащей в основании конуса, V - направление оси конуса (от основания к сечению), radius1 - радиус основания конуса, radius2 - радиус сечения конуса; формально - radius2 = radius1tang(alpha), alpha - угол между осью и образующей (угол полураствора). Подразумевается, что расстояние между основанием и сечением конуса равно 1.
ksMateSphere Сфера	Pc-координата центра сферы, radius1 - радиус сферы, V, radius2 - не используются.
ksMateTorus Тор	Pc-координата центра тора, V-направление оси вращения тора, radius1 - радиус вращения (большой), radius2 - радиус образующей окружности (малый).

ksMateCircle Окружность	Pc-координата центра окружности, V – направление вектора нормали плоскости, в которой лежит окружность, radius1 - радиус окружности, radius2 - radius1.

Общий свет

Этот параметр характеризует свет, отраженный и рассеиваемый другими объектами.

Например, в комнате с белыми стенами и рассеянным освещением значение общего света больше, чем в области, освещенной направленным источником света.

Диффузия

Этот параметр характеризует степень рассеивания света поверхностью (при этом считается, что свет рассеивается равномерно во всех направлениях).

Зеркальность

Этот параметр характеризует способность поверхности воспроизводить отражение яркого света.

Чем меньше значение параметра зеркальности, тем более тусклая поверхность.

Чем больше значение параметра, тем более зеркальная поверхность.

Блеск

Этот параметр характеризует блеск поверхности, остроту бликов на ней.

Чем меньше значение параметра зеркальности, тем более матовая поверхность.

Чем больше значение параметра, тем более блестящая, глянцевая поверхность.

Прозрачность

Этот параметр характеризует способность поверхности пропускать падающий на нее свет.

Если значение этого параметра равно 1 (или 100%), то поверхность совершенно непрозрачная.

При моделировании деталей из стекла, органического стекла, слюды, бесцветного полиэтилена и подобных материалов значение прозрачности выбирают из диапазона между 0 и 1. Если значение этого параметра равно 0,5 (или 50%), то поверхность полупрозрачная.

Если значение этого параметра равно 0, то поверхность полностью прозрачная. Такая поверхность невидима.

Излучение

Этот параметр характеризует способность поверхности излучать свет.

(*)

В системах, поддерживающих работу со свойствами, например, Visual Basic, Delphi, Builder.

(**)

В системах, не поддерживающих работу со свойствами, например, C.

Синтаксис доступен при использовании Ptr-оболочки интерфейсов (smartpointer).

A...B

Типы динамических массивов

ADimParam - структура параметров углового размера

ADimSource - структура параметров привязки углового размера

ArcParam - структура параметров дуги

ArcParam1 - структура параметров дуги по точкам

AStyles - Типы отрисовки стрелок в размерах

Attribute - структура параметров табличного атрибута

AttributeType - структура параметров типа табличного атрибута

ATTypes - Типы данных в столбце табличного атрибута

AxisLineParam - структура параметров осевой линии

BaseParam - структура параметров обозначения базы

BezierParam - структура параметров кривой Безье

BezierPointParam - структура параметров узла кривой Безье

BrandLeaderParam - структура параметров линии-выноски для обозначения клеймения

BreakDimDrawing - структура параметров отрисовки размера с обрывом

C

CentreParam - структура параметров обозначения центра

CircleParam - структура параметров окружности

ColumnInfo - структура параметров столбца табличного атрибута

CON - структура параметров сопрягающей окружности

ConicArcParam - структура параметров конического сечения

ConstraintParam - структура параметрических связей и ограничений

CopyObjectParam - Структура параметров копирования объекта графического документа

CornerParam - структура параметров скругленных углов многоугольников

CTypes - Типы колонок спецификации

CurvePattern -структура параметров участка прерывистой кривой

CurvePatternEx - расширенная структура параметров картинки в стиле линии

CurvePicture - структура параметров картинки в стиле линии

CurveStyleParam - структура параметров стиля кривой

CutLineParam - структура параметров линии разреза/сечения

D-F

DataTypes - Типы данных

DimDrawing - структура параметров изображения линейного и углового размеров

DimensionPartsParam - структура параметров объектов, составляющих размер

DimensionsOptions - структура параметров для определения настроек размеров

DimText - структура параметров размерной надписи

DModes - Признаки размерной надписи

DocAttachedSpcParam - структура параметров документа, подключенного к спецификации

DocType - Типы документов системы КОМПАС

DocumentParam - структура параметров документа

EllipseArcParam - структура параметров дуги эллипса

EllipseArcParam1 - структура параметров дуги эллипса

EllipseParam - структура параметров эллипса

EndType - Типы действий с библиотеками моделей и фрагментов

EquidistantParam - структура параметров эквидистанты

ErrorCodes - Коды ошибок

FolderTypes - Типы системных папок

FStyles - Стандартные форматы листа

G-K

HatchLineParam - структура параметров линии штриховки

HatchParam - структура параметров штриховки

HatchParamEx - расширенная структура параметров штриховки

HatchStyleParam - структура параметров стиля штриховки

HotPointDescription - структура параметров характерной точки

HStyles - Системные стили штриховок

InertiaParam - структура параметров для расчета МЦХ плоской фигуры

InsertFragmentParam - структура параметров вставки фрагмента

InsertFragmentParamEx - расширенная структура параметров вставки фрагмента

ksAttribute - структура параметров табличного атрибута

ksAttributeType - структура параметров типа табличного атрибута

L

L3DExportFormats - форматы сохранения модели

LayerParam - структура параметров слоя

LayerTypes - состояния слоев и видов

LBreakDimParam - структура параметров линейного размера с обрывом

LBreakDimSource - структура параметров привязки линейного размера с обрывом

LDimParam - структура параметров линейного размера

LDimSource - структура параметров привязки линейного размера

LeaderParam - структура параметров линии-выноски

LenTypes - размерности длины

LibPathTypes - типы пути в библиотеке моделей

LibraryAttrTypeParam - структура параметров для типа атрибута в библиотеке типов атрибутов

LibraryStyleParam - структура параметров стиля в библиотеке стилей

LibStyle - структура параметров для подключения стиля из библиотеки

LineParam - структура параметров вспомогательной прямой

LineSegParam - структура параметров отрезка

LObjColors - цвета вывода объектов

LPalettes - цветность растра

LRasterFormats - форматы растра

LStyles - системные стили линий

LtRemoteElmSignType - типы значка объекта "Выносной элемент

LtVariant - структура параметров для хранения данных некоторого типа

LtViewType - типы видов

LTypes - типы библиотек стилей

M-O

MarkerLeaderParam - структура параметров линии-выноски для обозначения маркировки

MarkTypes - Системные стили значков

MassInertiaParam - структура параметров для расчета МЦХ тел вращения и выдавливания

MathPointParam - структура параметров математической точки

MTypes - Размерности и типы тел

NumberTypeAttrParam - структура числового значения в колонке спецификации

NurbsParam - структура параметров кривой NURBS

NurbsPointParam - структура параметров узла NURBS

ObjTypes - Типы объектов и интерфейсы

OrdinatedDimParam - структура параметров размера высоты

OrdinatedDimTypes - Типы размеров высоты

OrdinatedDrawing - структура параметров изображения размера высоты

OrdinatedSource - структура параметров привязки размера высоты

OverlapObjectOptions - Параметры перекрывающихся объектов

P

ParagraphParam - структура параметров параграфа

ParamRestrictionTypes - Типы параметрических ограничений

ParamTypes - Типы параметров объектов

Phantom - структура параметров фантома

PhantomTypes - Типы фантомов

PlacementParam - структура параметров привязки

PointParam - структура параметров точки

PolylineParam - структура параметров ломаной линии

PolylineParamEx - расширенная структура параметров ломаной линии

PosLeaderParam - структура параметров позиционной линии-выноски

PropertyParam - структура параметров свойства отображаемого в окне свойств

ProjectionType - Типы проекций

PStyles - Системные стили отрисовки точек

PTypes - Типы фантомов

Q-R

QualityContensParam - структура параметров квалитета

QualityItemParam - структура параметров интервала квалитета

RasterFormatParam - структура параметров записи в растровый формат

RasterParam - структура параметров растрового объекта

RBreakDimParam - структура параметров радиального размера с изломом

RBreakDrawing - структура параметров изображения радиального размера с изломом

RDimDrawing - структура параметров изображения диаметрального и радиального размеров

RDimParam - структура параметров диаметрального и обычного радиального размера

RDimSource - структура параметров привязки диаметрального и радиального размеров

RecordTypeAttrParam - структура записи в колонке спецификации

RectangleParam - структура параметров прямоугольника

RectParam - структура параметров прямоугольника

RegularPolygonParam - структура параметров правильного многоугольника

RemoteElementParam - Параметры объекта "Выносной элемент"

RequestInfo - структура параметров запроса к системе

RoughPar - структура параметров обозначения шероховатости

RoughParam - структура параметров обозначения шероховатости с полкой

S

SetTypes - Типы настроек
SheetObjTypes - Типы объектов оформления чертежа
SheetOptions - Структура параметров оформления
SheetPar - структура параметров оформления документов
SheetSize - структура параметров нестандартного листа
ShelfPar - структура параметров выносной полки
SnapOptions - Структура параметров привязок в графическом документе
SortamentTypes - Коды типов сортаментов
SpcColumnParam - структура параметров колонки спецификации
SpcDescrParam - структура параметров описания спецификации
SpcObjParam - структура параметров объекта спецификации
SpcSortTypes - Типы сортировки объектов в разделе спецификации
SpcStrTypes - Типы строк спецификации
SpcStyleColumnParam - структура параметров стиля колонки спецификации
SpcStyleParam - структура параметров стиля спецификации
SpcStyleSectionParam - структура параметров стиля раздела спецификации
SpcSubSectionParam - структура параметров подраздела спецификации
SpcTuningSectionP - структура параметров настройки раздела спецификации
SpcTuningStyleParam - структура параметров настройки спецификации
SpcVariants - Варианты оформления спецификации
SpecRoughParam - структура параметров знака неуказанной шероховатости
SSTypes - Общие настройки привязок
StandartSheet - структура параметров стандартного листа
StandartViewTypes - Типы стандартных видов
StopTypes - Типы выхода из режима ksSystemControlStart
StructAssociationViewParam - Структура параметров ассоциативного вида
StructType2D - Типы интерфейсов
StTypes - Типы стилей и интерфейсы
STypes - Типы локальной привязки

T

TAN - структура параметров касательной кривой
TechnicalDemandParam - структура параметров технических требований
TextDocumentParam - Структура параметров текстового документа
TextFlags - Признаки начертания текста

TextItemFont - структура параметров шрифта компонента строки текста

TextItemParam - структура параметров компоненты строки текста

TextLineParam - структура параметров строки текста

TextParam - структура параметров текста

TextStyleParam - структура параметров стиля текста

ThicknessTypes - Параметры пера пользовательского стиля линии

ToleranceBranch - структура параметров опоры допуска формы

ToleranceParam - структура параметров обозначения допуска формы

TextAlign - Типы привязки текста

TStyles - Системные стили текстов

Type1 - структура параметров для сдвига группы

Type2 - структура параметров фантома-отрезка или фантома-окружности

Type3 - структура параметров фантома-отрезка с заданным углом и фантома-прямоугольника

Type5 - структура параметров фантома-половины прямоугольника с заданным углом

Type6 - структура параметров пользовательского фантома

U-Z

VariableParam - структура параметров параметрической переменной

ViewColorParam - Параметры цвета фона

ViewParam - структура параметров вида

ViewPointerParam - структура параметров стрелки направления взгляда

Методы получения

KompasObject::GetDynamicArray

KompasObject::ksGetLibraryStylesArray

ksAttributeObject::ksGetLibraryAttrTypesArray

ksAttributeTypeParam::GetColumns

ksBaseParam::GetPTextItem

ksBezierParam::GetMathPointArr

ksBrandLeaderParam::GetpPolyline

ksBrandLeaderParam::GetpTextline

ksColumnInfoParam::GetColumns

ksColumnInfoParam::GetFieldEnum

ksCurvePicture::GetFill

ksCurvePicture::GetPolygon

ksCurveStyleParam::GetPPattern

ksCutLineParam::GetpMathPoint

ksCutLineParam::GetpTextline
ksDimTextParam::GetTextArr
ksDocument2D::ksGetDocVariableArray
ksDocument2D::ksGetObjConstraints
ksLeaderParam::GetpPolyline
ksLeaderParam::GetpTextline
ksMarkerLeaderParam::GetpPolyline
ksMarkerLeaderParam::GetpTextline
ksMathematic2D::ksPointsOnCurve
ksNurbsParam::GetPKnot
ksNurbsParam::GetPPoint
ksPolylineParam::GetPMathPoint
ksPosLeaderParam::GetpPolyline
ksPosLeaderParam::GetpTextline
ksQualityContensParam::GetpQualityItems
ksRectangleParam::GetPCorner
ksRegularPolygonParam::GetPCorner
ksRoughPar::GetpText
ksSpcObjParam::GetDocArr
ksSpcStyleParam::GetArrAdditionalColumn
ksSpcStyleParam::GetArrColumn
ksSpcStyleParam::GetArrSection
ksSpcStyleSectionParam::GetArrAdditionalColumn
ksSpcStyleSectionParam::GetArrColumn
ksSpcTuningSectionParam::GetArrSubSection
ksSpcTuningStyleParam::GetArrSection
ksStamp::ksGetStampColumnText
ksTechnicalDemandParam::GetPGab
ksTextLineParam::GetTextItemArr
ksTextParam::GetTextLineArr
ksToleranceBranch::GetpMathPoint
ksToleranceParam::GetBranchArr
ksUserParam::GetUserArray
ksViewPointerParam::GetpTextline

См. также

KompasObject,

ksAttributeObject,
ksAttributeTypeParam,
ksBaseParam,
ksBezierParam,
ksBrandLeaderParam,
ksColumnInfoParam,
ksCurvePicture,
ksCurveStyleParam,
ksCutLineParam,
ksDimTextParam,
ksLeaderParam,
ksMarkerLeaderParam,
ksNurbsParam,
ksPolylineParam,
ksPosLeaderParam,
ksRectangleParam,
ksRegularPolygonParam,
ksRoughPar,
ksSpcObjParam,
ksSpcStyleParam,
ksSpcStyleSectionParam,
ksSpcTuningSectionParam,
ksSpcTuningStyleParam,
ksStamp,
ksTechnicalDemandParam,
ksTextLineParam,
ksTextParam,
ksToleranceBranch,
ksUserParam,
ksViewPointerParam.

Пример использования структуры параметров CornerParam и массива CORNER_ARR

```
reference arr = ::CreateArray (CORNER_ARR, 0); // создать массив параметров углов
 CornerParam par;// структура параметров угла
 par.index = 5;// индекс угла
 par.fillet = 0;// признак фаски
 par.l1 = 345;// длина фаски 1 сегмента
```

```
 par.l2 = 456.987;// длина фаски 2 сегмента
 ::AddArrayItem (arr, -1, &par, sizeof (CornerParam)); // добавить угол в массив
 par.index = 45;// индекс угла
 par.fillet = 0;// признак фаски
 par.l1 = 345;// длина фаски 1 сегмента
 par.l2 = 456.987;// длина фаски 2 сегмента
 ::AddArrayItem (arr, -1, &par, sizeof (CornerParam));// добавить угол в массив
 par.index = 65;// индекс угла
 par.fillet = 0;// признак фаски
 par.l1 = 345;// длина фаски 1 сегмента
 par.l2 = 456.987;// длина фаски 2 сегмента
 ::AddArrayItem (arr, -1, &par, sizeof (CornerParam));// добавить угол в массив
 par.index = 85;// индекс угла
 par.fillet = 0;// признак фаски
 par.l1 = 345;// длина фаски 1 сегмента
 par.l2 = 456.987;// длина фаски 2 сегмента
 ::AddArrayItem (arr, -1, &par, sizeof (CornerParam));// добавить угол в массив
 int count = ::GetArrayCount (arr);// количество элементов в массиве
 for ( int i = 0; i < count; i ) {
// выводим параметры углов в окне сообщения
char buf[128];
if ( ::GetArrayItem (arr, i, &par, sizeof (CornerParam))) {
::sprintf ( buf, "индекс элемента %i равен %i; %i, %f, %f", i,
par.index, par.fillet, par.l1, par.l2);
::Message (buf);
}
 }
 ::ExcludeArrayItem (arr, 2);// удаляем второй элемент массива
 count = ::GetArrayCount (arr);// количество элементов в массиве
 for ( int i = 0; i < count; i ) {
// выводим параметры углов в окне сообщения
char buf [128];
if ( ::GetArrayItem (arr, i, &par, sizeof (CornerParam))) {
::sprintf (buf, "индекс элемента %i равен %i; %i, %f, %f", i, par.index,
par.fillet, par.l1, par.l2 );
::Message( buf );
}
 }
```

```
 if ( ::GetArrayItem (arr, 1, &par, sizeof (CornerParam))) {
// изменяем параметры элемента с индексом 1
par.index = 0;
par.fillet = 0;
par.l1 = 0;
par.l2 = 0;
::SetArrayItem( arr, 1, &par, sizeof(CornerParam) );
}
 count = ::GetArrayCount (arr);// количество элементов в массиве
 for ( int i = 0; i < count; i ) {
// выводим параметры углов в окне сообщения
char buf[128];
if ( ::GetArrayItem (arr, i, &par, sizeof (CornerParam))) {
::sprintf (buf, "индекс элемента %i равен %i; %i, %f, %f", i, par.index,
par.fillet, par.l1, par.l2);
::Message( buf );
}
 }
 ::ClearArray (arr);// очищаем массив
 char buf[128];
 ::sprintf (buf, "кол-во элементов = %i", ::GetArrayCount (arr));
 ::Message (buf);
 ::DeleteArray (arr);// удаляем массив
```

Пример использования структуры параметров CurveStyleParam

```
CurveStyleParam par;  //структура параметров стиля кривой
memset (&par, 0, sizeof (par));
CurvePattern cPatt;  //структура параметров участка штриховой кривой

par.pattern = CreateArray (CURVE_PATTERN_ARR, 0);
cPatt.visibleSeg = 15;
cPatt.invisibleSeg = 7;
AddArrayItem (par.pattern, -1, &cPatt, sizeof (cPatt));
cPatt.visibleSeg = 3;
cPatt.invisibleSeg = 7;
AddArrayItem (par.pattern, -1, &cPatt, sizeof (cPatt));
strcpy (par.name, "style from library"); //имя стиля
par.color = RGB (255, 0, 0);       //цвет линии
```

```
par.paperWidth = 0.8;          //толщина пера на бумаге
par.screenWidth = 3;           //толщина линии на экране
par.curveType = 1|LIKE_BASIC_LINE;    //прерывистая линия с параметрами пера
//как у системной основной линии
par.even = 1;                  //кривая всегда оканчивается штрихом
UINT tl = AddStyle (CURVE_STYLE, &par, sizeof (par), 0);
LineSeg (20, 20, 70, 20, tl);

int t = GetStyleParam (CURVE_STYLE, tl, &par, sizeof (par));
if (t) {
char buf[128];
sprintf (buf, "curveType = %s",
par.curveType & LIKE_BASIC_LINE ? "LIKE_BASIC_LINE" : "NO");
Message (buf);
}
else
Error("Ошибка");
```

Типы специальных символов на концах аннотационного объекта (аннотационная линия и аннотационная дуга)

ARROW_INSIDE_SYMBOL	1	стрелка (ласточкин хвост) изнутри
ARROW_OUT_SIDE_SYMBOL	2	стрелка (ласточкин хвост) снаружи
TICK_TAIL_SYMBOL	3	засечка с продолжением кривой (с хвостиком)
UP_HALF_ARROW_SYMBOL	4	верхняя половина стрелки изнутри
DOWN_HALF_ARROW_SYMBOL	5	нижняя половина стрелки изнутри
BIG_ARROW_INSIDE_SYMBOL	6	большая стрелка изнутри (7мм)
ARROW_ORDINATE_DIM_SYMBOL	7	стрелка для размера высоты (штрихи длиной 4 мм под углом 45 градусов
TRIANGLE_SYMBOL	8	треугольник по направлению кривой
CIRCLE_RAD2_SYMBOL	9	окружность радиусом 2 мм тонкой линией - для шероховатости и линии-выноски
CENTRE_MARKER_SYMBOL	10	обозначение фиктивного центра в виде большого креста
GLUE_SIGN_SYMBOL	11	знак склеивания
SOLDER_SIGN_SYMBOL	12	знак пайки
SEWING_SIGN_SYMBOL	13	знак сшивания
CRAMP_SIGN_SYMBOL	14	знак соединения внахлестку металлическими скобами
CORNER_CRAMP_SIGN_SYMBOL	15	знак углового соединения металлическими скобами
MONTAGE_JOINT_SYMBOL	16	знак монтажного шва
TICK_SYMBOL	17	засечка без продолжения кривой (без хвостика)

TRIANGLE_CURR_CS	18	треугольник по текущей СК - для базы
ARROW_CLOSED_INSIDE	19	закрытая стрелка изнутри
ARROW_CLOSED_OUTSIDE	20	закрытая стрелка снаружи
ARROW_OPEN_INSIDE	21	открытая стрелка изнутри
ARROW_OPEN_OUTSIDE	22	открытая стрелка снаружи
ARROW_RIGHTANGLE_INSIDE	23	стрелка 90 градусов изнутри
ARROW_RIGHTANGLE_OUTSIDE	24	стрелка 90 град снаружи
SYMBOL_DOT	25	точка (диаметр равен длине стрелки размера)
SYMBOL_SMALLDOT	26	точка маленькая (диаметр равен 0.6 длины стрелки размера)
AUXILIARY_POINT	27	вспомогательная точка
LEFT_TICK_SYMBOL	28	засечка с наклоном влево

Структура не заполняется, создается только массив строк

Структура не заполняется, определяется только цвет штриховки

Зависимость параметра от типа компиляции

В зависимости от типа компиляции используется как ANSI или Unicode вариант.

При использовании в проекте предопределенного определения _UNICODE, предназначенного для компиляции проекта под Unicode, константе присваивается значение, заданное с в объявлении с суффиксом W.

Например:

```
#ifdef _UNICODE
#define ALLPARAM_T ALLPARAM_W
#else
#define ALLPARAM_T ALLPARAM
#endif // !UNICODE
```

Пример использования

_BRACKETS размер в круглых скобках

_BRACKETS | _SQUARE_BRACKETS - размер в квадратных скобках

IImportedSurfaceDefinition::AddCurve - пример использования

```
void WorkImportedSurfaceAddCurve( KompasObject &kompas ){

  int pointCountAll = 6; //кратное трем (x, y, z)
```

```
int curveCount = 3;

double x = 50, dx = 30;
double y = 50, dy = 40;
double z = 0,  dz = 100;


int pointCount = pointCountAll / 3;

SAFEARRAYBOUND sabound[1];
SAFEARRAY FAR *psa;
// Создание safe array double
sabound[0].cElements = pointCountAll;
sabound[0].lLbound = 0;
psa = ::SafeArrayCreate(VT_R8, 1, sabound);

VARIANT v;
VariantInit(&v);
V_VT(&v) = VT_ARRAY | VT_R8;
V_ARRAY(&v) = psa;
long index;
double d;


if ( kompas.m_lpDispatch ) {
  //получить активный 3d документ
  ksDocument3D doc( kompas.ActiveDocument3D() );
  if ( doc.m_lpDispatch ) {
    //получить компонент документа
    ksPart part( doc.GetPart( pTop_Part) );
    if ( part ) {
      //получить новый объект импортированной поверхности
      ksEntity entity(part.NewEntity(o3d_importedSurface));
      if (entity) {
        //Определение импортированной поверхности
        ksImportedSurfaceDefinition importedSurfaceDef( entity.GetDefinition());

        //добавляем в импортированную поверхность кривые по массиву точек
```

```
        if (importedSurfaceDef){
          for ( uint i = 0; i < curveCount; i ) {
            for (uint j = 0; j < pointCount; j ) {
              //наполняем safe array. Каждая точка представлена тремя координатами x, y, z
              index = j*3;
              d = x  dx*i;
              ::SafeArrayPutElement(psa, &index, &d);
              index = j*3  1;
              d = y - dy*(i%2);
              ::SafeArrayPutElement(psa, &index, &d);
              index = j*3  2;
              d = z  dz*j;
              ::SafeArrayPutElement(psa, &index, &d);
            }
            //добавить кривую по массиву точек
            importedSurfaceDef.AddCurve ( v );
          }
          //создать импортированную поверхность
          entity.Create();
        }
      }
    }
  }
  else
    kompas.ksError( "3d документ не активизирован" );
 }
 //удалить safe array double
 if ( psa )
  ::SafeArrayDestroy(psa);

}
```

Пример задания Placement детали

Последовательность установки направления осей задается абсолютными координатами направляющих векторов осей.

Для их задания нужно передавать сумму координат точки привязки и смещения.

```
// Получить указатель на интерфейс размещения детали
```

```
IPlacement *iPlacement = m_pPart->GetPlacement();
// Установить точку привязки:
iPlacement->SetOrigin(x, y, z);

// Установить направления осей:
iPlacement->SetAxis(
x  1,
y  0,
z  0,
0); // ось X
iPlacement->SetAxis(
x  0,
y  1,
z  0,
1); // ось Y
// Обновить Placement:
m_pPart->UpdatePlacement();
```

Варианты оформления спецификации

Параметр variant может иметь следующие значения для выбора варианта оформления спецификации:

0	- простая	
1	- групповая, вариант А	
2	- групповая, вариант Б	
3	- групповая, вариант В	- временно недоступен
4	- групповая, вариант Г	- временно недоступен

Типы колонок спецификации

SPC_CLM_FORMAT	1	- формат
SPC_CLM_ZONE	2	- зона
SPC_CLM_POS	3	- позиция
SPC_CLM_MARK	4	- обозначение
SPC_CLM_NAME	5	- наименование
SPC_CLM_COUNT	6	- количество
SPC_CLM_NOTE	7	- примечание
SPC_CLM_MASSA	8	- масса
SPC_CLM_MATERIAL	9	- материал
SPC_CLM_USER	10	- пользовательская
SPC_CLM_KOD	11	- код
SPC_CLM_FACTORY	12	- предприятие-изготовитель

Типы объектов спецификации

В API7 соответствует перечисление ksSpecificationObjectTypeEnum - Типы объектов спецификации.

SPC_BASE_OBJECT	1	- базовый объект
SPC_COMMENT	2	- вспомогательный объект
SPC_SECTION_NAME	3	- заголовок раздела
SPC_BLOCK_NAME	4	- заголовок блока исполнений
SPC_RESERVE_STR	5	- резервная строка
SPC_EMPTY_STR	6	- пустая строка в конце страницы

Типы сортировки объектов в разделе спецификации

SPC_SORT_OFF	0	- нет сортировки
SPC_SORT_COMPOS	1	- составная сортировка
SPC_SORT_ALPHABET	2	- сортировка по алфавиту
SPC_SORT_UP	3	- сортировка по возрастанию числового значения
SPC_SORT_DOCUMENT	4	- сортировка раздела документация
SPC_SORT_DOWN	5	- сортировка по убыванию числового значения

ksUserParam::fileName - пример использования

Получить имя файла пользовательской библиотеки, при помощи которой можно редактировать компонент.

```
  //получить интерфейс ksUserParam у KompasObject
ksUserParam userParam(kompasObject.GetParamStruct(ko_UserParam));
userParam.Init();
  //заполнить параметры интерфейса
ksPart.GetUserParam(userParam);
  //Получить имя файла пользовательской библиотеки
BSTR userLibraryFileName = userParam.fileName;
```

ksUserParam::number - пример использования

Получить номер команды пользовательской библиотеки, при помощи которой можно редактировать компонент.

```
  //получить интерфейс ksUserParam у KompasObject
ksUserParam userParam( kompasObject.GetParamStruct(ko_UserParam) );
userParam.Init();
  //заполнить параметры интерфейса
ksPart.GetUserParam( userParam );
  //Получить номер команды
```

```
long number = userParam.number;
```

Пример изменения порядка NURBS

Изменить порядок NURBS кривой на 4.

nurbs.Degree = 4 нужно устанавливать не раньше, чем в NURBS кривую добавятся 4 точки.

```
nurbs = nurbses.Add()
nurbs.Style = kompas6_constants.ksCSNormal
nurbs.Closed = False
#nurbs.Degree = 4
nurbs.AddPoint(0, 49.7220581829, 124.864112265, 1.0)
nurbs.AddPoint(1, 79.97915965, 161.394027451, 1.0)
nurbs.AddPoint(2, 110.974239202, 124.864112265, 1.0)
nurbs.AddPoint(3, 173.702376389, 181.688424777, 1.0)
nurbs.Degree = 4
nurbs.Update()
```