**СПРАВОЧНИК ПО API**

**1. Общие принципы написания программы**

Методы и свойства классов приложения позволяют выполнять действия, аналогичные действиям, которые выполняет пользователь при работе с приложением. Порядок выполнения методов должен соответствовать порядку соответствующих действий пользователя.

**2. Основные классы приложения**

Объект класса **SldWorks** предоставляет доступ к приложению.

Объект класса **ModelDoc2** предоставляет доступ к документам приложения.

Объект класса **PartDoc** предоставляет доступ к документам типа Деталь.

Если в приложении активным является документ типа Деталь, одна и та же ссылка на данные объектов классов **ModelDoc2** и **PartDoc** возвращается следующим методом класса **SldWorks**:

**Public Property ActiveDoc() As Object**

Объект класса **ModelDocExtension** предоставляет расширенный доступ к документу типа Деталь. Этот объект возвращается следующим свойством класса **ModelDoc2**:

**Public Property Extension() As ModelDocExtension**

Объект класса **FeatureManager** позволяет работать с элементами дерева построения. Этот объект возвращается следующим свойством класса **ModelDoc2**:

**Public Property FeatureManager() As FeatureManager**

Объект класса **SketchManager** предоставляет доступ к эскизам и возвращается следующим свойством класса **ModelDoc2**:

**Public Property SketchManager() As SketchManager**

Элементы дерева построения представляются объектами класса **Feature** и создаются специальными методами некоторых классов, в том числе классов **FeatureManager** и **ModelDoc2**.

Такие элементы эскиза, как линия, дуга окружности (окружность), сплайн и некоторые другие представляются объектами класса **SketchSegment**. Точкам эскиза соответствует класс **SketchPoint**. Линии эскиза, дуги окружности эскиза и сплайны эскиза дополнительно представляются объектами соответственно класса **SketchLine**, класса **SketchArc** и класса **SketchSpline**. Свойства и методы классов **SketchLine**, **SketchArc** и **SketchSpline** доступны по ссылке на данные соответствующих объектов класса **SketchSegment**. Элементы эскиза создаются специальными методами некоторых классов, в том числе классов **SketchManager** и **ModelDoc2**.

**3. Идентификация объектов элементов дерева построения**

Действие: чтение-запись имени элемента дерева построения.

Доступ: свойство класса **Feature**.

Описание:

**Public Property Name() As String**

Действие: чтение типа элемента дерева построения.

Доступ: свойство класса **Feature**.

Описание:

**Public Function GetTypeName2() As String**

Возвращаемое значение:

1) вырез по сечениям – “BlendCut”;

2) фаска – “Chamfer”;

3) круговой массив – “CirPattern”;

4) массив, управляемый кривой – “CurvePattern”;

5) вырез, полученный вытягиванием – “Cut”;

6) бобышка, полученная вытягиванием – “Extrusion”;

7) простое скругление – “Fillet”;

8) спираль/винтовая кривая – “Helix”;

9) вырез, созданный поворотом – “RevCut”;

10) бобышка, созданная поворотом – “Revolution”;

11) бобышка, созданная заметанием – “Sweep”;

12) вырез, созданный заметанием – “SweepCut”;

13) скругление переменного радиуса – “VarFillet”;

14) справочная ось – “RefAxis”;

15) справочная кривая – “ReferenceCurve”;

16) справочная плоскость – “RefPlane”

и др.

Действие: получить объект первого элемента дерева построения.

Доступ: метод класса **PartDoc**.

Описание:

**Public Function FirstFeature() As Object**

Возвращаемое значение:

объект класса **Feature**.

Действие: получить объект следующего элемента дерева построения.

Доступ: метод класса **Feature**.

Описание:

**Public Function GetNextFeature() As Object**

Возвращаемое значение:

объект класса **Feature**.

Примечание:

подавленные элементы также возвращаются.

Действие: получить объект последнего добавленного элемента дерева построения.

Доступ: метод класса **ModelDocExtension**.

Описание:

**Public Function GetLastFeatureAdded() As Feature**

Действие: получить грани.

Доступ: метод класса **Feature**.

Описание:

**Public Function GetFaces() As Variant**

Возвращаемое значение:

одномерный массив объектов класса **Face2**.

Действие: получить количество граней.

Доступ: метод класса **Feature**.

Описание:

**Public Function GetFaceCount() As Long**

Действие: получить кромки.

Доступ: метод класса **Face2**.

Описание:

**Public Function GetEdges() As Variant**

Возвращаемое значение:

одномерный массив объектов класса **Edge**.

Действие: получить количество кромок.

Доступ: метод класса **Face2**.

Описание:

**Public Function GetEdgeCount() As Long**

**3. Выделение**

Действие: снять выделение.

Доступ: метод класса **ModelDoc2**.

Описание:

**Public Sub ClearSelection2**( \_

**ByVal** All **As** **Boolean** \_

)

Аргументы:

если аргумент All равен **True**, то элементы убираются из всего списка выделения, если **False**, то элементы убираются только из активного списка выделения.

Действие: выделить элемент модели.

Доступ: метод класса **ModelDocExtension**.

Описание:

**Function** **SelectByID2**( \_

**ByVal** Name **As** **String**, \_

**ByVal** Type **As** **String**, \_

**ByVal** X **As** **Double**, \_

**ByVal** Y **As** **Double**, \_

**ByVal** Z **As** **Double**, \_

**ByVal** Append **As** **Boolean**, \_

**ByVal** Mark **As** **Long**, \_

**ByVal** Callout **As** **Callout**, \_

**ByVal** SelectOption **As** **Long** \_

) **As** **Boolean**

Аргументы:

Name – имя элемента модели или пустая строка;

Type – тип элемента модели или пустая строка;

X, Y, Z – координаты точки элемента модели;

если аргумент Append равен **True**, то невыделенный элемент модели будет добавлен к существующему выделению или выделенный элемент модели будет исключен из существующего выделения, если **False**, то существующее выделение будет снято и создано новое из невыделенного элемента модели или существующее выделение, включающее выделенный элемент модели, будет оставлено без изменения;

Mark – номер элемента модели в выделении;

Callout – объект класса **Callout** (или **Nothing**);

SelectOption – опции ‑ значение из перечисления **swSelectOption\_e**.

Возвращаемое значение:

**True**, если элемент модели стал выделенным.

Примечание:

1) аргумент Type может иметь следующие значения:

“DATUMPOINT” – справочная точка;

“AXIS” – справочная ось;

“PLANE” – справочная плоскость;

“REFCURVE” – справочная кривая;

“HELIX” – спираль/винтовая кривая;

“SKETCH” – эскиз;

“SKETCHSEGMENT” – элемент эскиза, если эскиз активен;

“EXTSKETCHSEGMENT” – элемент эскиза, если эскиз неактивен;

“SKETCHPOINT” – точка эскиза, если эскиз активен;

“EXTSKETCHPOINT” – точка эскиза, если эскиз неактивен;

“FACE” – грань;

“EDGE” – кромка;

“VERTEX” – вершина;

“BODYFEATURE” – тело;

“POINTREF” – точка;

“NOTHING” – ничего;

“EVERYTHING” – все;

2) аргумент Name для размера должен быть полным именем (например, “D1@Sketch2@Part1.SLDPRT”);

3) координаты точки элемента модели должны соответствовать той системе координат, в которой элемент был создан.

Элементы перечисления **swSelectOption\_e**:

**swSelectOptionDefault** – клавиша Shift не используется;

**swSelectOptionExtensive** ‑ клавиша Shift используется.

**4. Создание справочной геометрии дерева построения**

Действие: создать справочную точку.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertReferencePoint**( \_

**ByVal** NRefPointType **As** **Long**, \_

**ByVal** NRefPointAlongCurveType **As** **Long**, \_

**ByVal** DDistance\_or\_Percent **As** **Double**, \_

**ByVal** NumberOfRefPoints **As** **Long** \_

) **As** **Variant**

Аргументы:

NRefPointType – тип справочной точки – значение из перечисления **swRefPointType\_e**;

NRefPointAlongCurveType – тип справочной точки на кривой – значение из перечисления **swRefPointAlongCurveType\_e**;

DDistance\_or\_Percent – расстояние или процент длины, если аргумент NRefPointAlongCurveType равен **swRefPointAlongCurveDistance** или **swRefPointAlongCurvePercentage** соответственно;

NumberOfRefPoints – количество справочных точек, если аргумент NRefPointAlongCurveType равен **swRefPointAlongCurveEvenlyDistributed**.

Возвращаемое значение:

одномерный массив объектов класса **Feature**.

Элементы перечисления **swRefPointType\_e**:

**swRefPointInvalid** ‑ недействительная;

**swRefPointUndefined** ‑ неопределена;

**swRefPointAlongCurve** – вдоль кривой;

**swRefPointCenterEdge** – центр кромки;

**swRefPointFaceCenter** – центр грани;

**swRefPointFaceVertexProjection** – проекция вершины;

**swRefPointIntersection** ‑ пересечение.

Действие: создать справочную ось.

Доступ: метод класса **ModelDoc2**.

Описание:

**Function** **InsertAxis2**( \_

**ByVal** AutoSize **As** **Boolean** \_

) **As** **Boolean**

Аргументы:

если аргумент AutoSize равен **True**, то справочная ось автоматически установит свою длину.

Возвращаемое значение:

**True**, если справочная ось была создана.

Действие: создать справочную плоскость.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertRefPlane**( \_

**ByVal** FirstConstraint **As** **Long**, \_

**ByVal** FirstConstraintAngleOrDistance **As** **Double**, \_

**ByVal** SecondConstraint **As** **Long**, \_

**ByVal** SecondConstraintAngleOrDistance **As** **Double**, \_

**ByVal** ThirdConstraint As **Long**, \_

**ByVal** ThirdConstraintAngleOrDistance **As** **Double** \_

) **As** **Object**

Аргументы:

FirstConstraint – первое ограничение – значение из перечисления **swRefPlaneReferenceConstraints\_e**;

FirstConstraintAngleOrDistance – угол или расстояние для первого ограничения;

SecondConstraint – второе ограничение – значение из перечисления **swRefPlaneReferenceConstraints\_e**;

SecondConstraintAngleOrDistance – угол или расстояние для второго ограничения;

ThirdConstraint ‑ третье ограничение – значение из перечисления **swRefPlaneReferenceConstraints\_e**;

ThirdConstraintAngleOrDistance ‑ угол или расстояние для третьего ограничения.

Возвращаемое значение:

объект класса **RefPlane**.

Примечание:

до создания справочной плоскости элементы модели должны быть выделены методом **SelectByID2** со следующими порядковыми номерами:

0 – первый элемент;

1 – второй элемент;

2 – третий элемент.

Элементы перечисления **swRefPlaneReferenceConstraints\_e**:

**swRefPlaneReferenceConstraint\_Parallel** ‑ параллельность;

**swRefPlaneReferenceConstraint\_Perpendicular** ‑ перпендикулярность;

**swRefPlaneReferenceConstraint\_Coincident** ‑ совпадение;

**swRefPlaneReferenceConstraint\_Distance** ‑ расстояние;

**swRefPlaneReferenceConstraint\_Angle** ‑ угол;

**swRefPlaneReferenceConstraint\_Tangent** ‑ касательность;

**swRefPlaneReferenceConstraint\_Project** – проекция точки эскиза (вершины или начала координат);

**swRefPlaneReferenceConstraint\_MidPlane** – средняя плоскость;

**swRefPlaneReferenceConstraint\_OptionFlip** ‑ реверсирование;

**swRefPlaneReferenceConstraint\_OptionOriginOnCurve** – точка плоскости на кривой;

**swRefPlaneReferenceConstraint\_OptionProjectToNearestLocation** – проекция точки эскиза (вершины или начала координат) на касательную плоскость к поверхности, ближайшую к точке эскиза (вершине или началу координат);

**swRefPlaneReferenceConstraint\_OptionProjectAlongSketchNormal** – проекция точки эскиза на касательную плоскость к поверхности по нормали эскиза.

**5. Создание эскизов**

Действие: создать 3D эскиз или выйти из активного 3D эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Sub Insert3DSketch**( \_

**ByVal** UpdateEditRebuild **As Boolean** \_

)

Аргументы:

если аргумент UpdateEditRebuild равен **True**, то будут осуществлены обновление и перестройка.

Действие: создать 2D эскиз или выйти из активного 2D эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Sub InsertSketch**( \_

**ByVal** UpdateEditRebuild **As Boolean** \_

)

Аргументы:

если аргумент UpdateEditRebuild равен **True**, то будут осуществлены обновление и перестройка.

**6. Управление отображением элементов эскиза**

Действие: чтение-запись типа добавления элемента эскиза в базу данных.

Доступ: свойство класса **SketchManager**.

Описание:

**Public Property AddToDB() As Boolean**

Значение:

**True** – элемент эскиза будет добавлен в базу данных напрямую, минуя отображение.

Примечание:

значение **False** по умолчанию необходимо установить до завершения работы программы.

Действие: чтение-запись типа отображения элемента эскиза после его добавления в базу данных напрямую.

Доступ: свойство класса **SketchManager**.

Описание:

**Public Property DisplayWhenAdded() As Boolean**

Значение:

**False** – элемент эскиза не будет отображен после его добавления в базу данных напрямую.

Примечание:

значение **True** по умолчанию необходимо установить до завершения работы программы.

**7. Создание элементов эскиза**

Элемент эскиза создается только в активном эскизе. Линейные размеры задаются в м. Координата *z* точек элементов 2D эскиза игнорируется.

Действие: создать точку эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreatePoint**( \_

**ByVal** X **As** **Double**, \_

**ByVal** Y **As** **Double**, \_

**ByVal** Z **As** **Double** \_

) **As** **SketchPoint**

Аргументы:

X, Y, Z – координаты точки эскиза.

Действие: создать линию эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateLine**( \_

**ByVal** X1 **As** **Double**, \_

**ByVal** Y1 **As** **Double**, \_

**ByVal** Z1 **As** **Double**, \_

**ByVal** X2 **As** **Double**, \_

**ByVal** Y2 **As** **Double**, \_

**ByVal** Z2 **As** **Double** \_

) **As** **SketchSegment**

Аргументы:

X1, Y1, Z1 – координаты начала линии эскиза;

X2, Y2, Z2 – координаты конца линии эскиза.

Действие: создать окружность эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateCircle**( \_

**ByVal** XC **As** **Double**, \_

**ByVal** YC **As** **Double**, \_

**ByVal** ZС **As** **Double**, \_

**ByVal** XP **As** **Double**, \_

**ByVal** YP **As** **Double**, \_

**ByVal** ZP **As** **Double** \_

) **As** **SketchSegment**

Аргументы:

XC, YC, ZC – координаты центра;

XP, YP, ZP – координаты точки окружности эскиза.

Действие: создать окружность эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateCircleByRadius**( \_

**ByVal** XC **As** **Double**, \_

**ByVal** YC **As** **Double**, \_

**ByVal** ZC **As** **Double**, \_

**ByVal** Radius **As** **Double** \_

) **As** **SketchSegment**

Аргументы:

XC, YC, ZC – координаты центра;

Radius – радиус окружности эскиза.

Действие: создать дугу окружности эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **Create3PointArc**( \_

**ByVal** X1 **As** **Double**, \_

**ByVal** Y1 **As** **Double**, \_

**ByVal** Z1 **As** **Double**, \_

**ByVal** X2 **As** **Double**, \_

**ByVal** Y2 **As** **Double**, \_

**ByVal** Z2 **As** **Double**, \_

**ByVal** X3 **As** **Double**, \_

**ByVal** Y3 **As** **Double**, \_

**ByVal** Z3 **As** **Double** \_

) **As** **SketchSegment**

Аргументы:

X1, Y1, Z1 – координаты первой точки дуги окружности эскиза;

X2, Y2, Z2 – координаты второй точки дуги окружности эскиза;

X3, Y3, Z3 – координаты третьей точки дуги окружности эскиза.

Действие: создать дугу окружности эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateArc**( \_

**ByVal** XC **As** **Double**, \_

**ByVal** YC **As** **Double**, \_

**ByVal** ZC **As** **Double**, \_

**ByVal** X1 **As** **Double**, \_

**ByVal** Y1 **As** **Double**, \_

**ByVal** Z1 **As** **Double**, \_

**ByVal** X2 **As** **Double**, \_

**ByVal** Y2 **As** **Double**, \_

**ByVal** Z2 **As** **Double**, \_

**ByVal** Direction **As** **Integer** \_

) **As** **SketchSegment**

Аргументы:

XC, YC, ZC – координаты центра;

X1, Y1, Z1 – координаты первой точки дуги окружности эскиза;

X2, Y2, Z2 – координаты второй точки дуги окружности эскиза;

если аргумент Direction равен +1, то направление дуги окуружности эскиза противоположно ходу часовой стрелки, если ‑ -1, то направление дуги окружности эскиза совпадает с направлением хода часовой стрелки.

Действие: создать касательную дугу окуружности эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateTangentArc**( \_

**ByVal** X1 **As** **Double**, \_

**ByVal** Y1 As **Double**, \_

**ByVal** Z1 As **Double**, \_

**ByVal** X2 As **Double**, \_

**ByVal** Y2 As **Double**, \_

**ByVal** Z2 As **Double**, \_

**ByVal** ArcType **As** **Long** \_

) **As** **SketchSegment**

Аргументы:

X1, Y1, Z1 – координаты первой точки дуги окружности эскиза;

X2, Y2, Z2 – координаты второй точки дуги окружности эскиза;

ArcType – тип дуги окружности эскиза – значение из перечисления **swTangentArcTypes\_e**.

Элементы перечисления **swTangentArcTypes\_e**:

**swForward** – вектор касательной к дуге окружности будет направлен по касательной к кривой;

**swLeft** ‑ вектор касательной к дуге окружности будет направлен перпендикулярно к касательной к кривой поворотом налево;

**swBack** ‑ вектор касательной к дуге окружности будет направлен противоположно касательной к кривой;

**swRight** – вектор касательной к дуге окружности будет направлен перпендикулярно к касательной к кривой поворотом направо.

Действие: создать сплайн эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateSpline2**( \_

**ByVal** PointData **As** **Variant**, \_

**ByVal** SimulateNaturalEnds **As** **Boolean** \_

) **As** **SketchSegment**

Аргументы:

PointData – одномерный массив последовательно расположенных троек координат X, Y, Z точек сплайна эскиза;

если аргумент SimulateNaturalEnds равен **True**, то будут имитированы натуральные концы сплайна эскиза.

Примечание: в массиве PointData должны быть координаты минимум двух точек; индекс массива должен начинаться с 0.

Действие: создать фаску эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateChamfer**( \_

**ByVal** Type **As** **Long**, \_

**ByVal** Distance **As** **Double**, \_

**ByVal** AngleOrDist **As** **Double** \_

) **As** **SketchSegment**

Аогументы:

Type – тип – значение из перечисления **swSketchChamferType\_e**;

Distance – расстояние;

AngleOrDist – второе расстояние, если аргумент Type равен **swSketchChamfer\_DistanceDistance**, угол, если аргумент Type равен **swSketchChamfer\_DistanceAngle**, игнорируется, если аргумент Type равен **swSketchChamfer\_DistanceEqual**.

Действие: создать скругление эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **CreateFillet**( \_

**ByVal** Radius **As** **Double**, \_

**ByVal** ConstrainedCorners **As** **Long** \_

) **As** **SketchSegment**

Аргументы:

Radius – радиус;

ConstrainedCorners – тип действия, которое необходимо выполнить, если на скругление эскиза накладывается ограничение или размер – значение из перечисления **swConstrainedCornerAction\_e**;

Элементы перечисления **swConstrainedCornerAction\_e**:

s**wConstrainedCornerInteract** – запрос на дальнейшие действия;

**swConstrainedCornerKeepGeometry** – сохранение ограничения или размера путем создания воображаемой точки пересечения;

**swConstrainedCornerDeleteGeometry** – удаление ограничения или размера;

**swConstrainedCornerStopProcessing** – остановка действия.

**8. Параметры элементов эскиза и их редактирование**

Действие: получить тип сегмента эскиза.

Доступ: метод класса **SketchSegment**.

Описание:

**Public Function GetType() As Long**

Возвращаемое значение:

значение из перечисления **swSketchSegments\_e**.

Элементы перечисления **swSketchSegments\_e**:

**swSketchLINE** – линия;

**swSketchARC** – дуга окружности или окружность;

**swSketchELLIPSE** – эллипс;

**swSketchSPLINE** – сплайн;

**swSketchTEXT** – текст;

**swSketchPARABOLA** – парабола.

Действие: читать-записать тип элемента эскиза касательно справочной геометрии.

Доступ: свойство класса **SketchSegment**.

Описание:

**Public Property ConstructionGeometry() As Boolean**

Значение:

если **True**, то элемент эскиза будет преобразован в справочную геометрию.

Действие: читать-записать абсциссу точки эскиза.

Доступ: свойство класса **SketchPoint**.

Описание:

**Public Property X() As Double**

Действие: читать-записать ординату точки эскиза.

Доступ: свойство класса **SketchPoint**.

Описание:

**Public Property Y() As Double**

Действие: читать-записать ординату точки эскиза.

Доступ: свойство класса **SketchPoint**.

Описание:

**Public Property Z() As Double**

Действие: читать-записать угол наклона линии эскиза.

Доступ: свойство класса **SketchLine**.

Описание:

**Public Property Angle() As Double**

Действие: получить конечную точку линии эскиза.

Доступ: метод класса **SketchLine**.

Описание:

**Public Function GetEndPoint2() As Object**

Возвращаемоезначение:

объект класса **SketchPoint**.

Действие: получить начальную точку линии эскиза.

Доступ: метод класса **SketchLine**.

Описание:

**Public Function GetStartPoint2() As Object**

Возвращаемое значение:

объект класса **SketchPoint**.

Действие: получить тип дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function IsCircle() As Integer**

Возвращаемое значение:

если значение равно 1, то дуга окружности – это полная окружность, если значение равно 0, то дуга окружности – это неполная окружность.

Действие: получить центр дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function GetCenterPoint2() As Object**

Возвращаемое значение:

объект класса **SketchPoint**.

Действие: получить конечную точку дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function GetEndPoint2() As Object**

Возвращаемоезначение:

объект класса **SketchPoint**.

Действие: получить начальную точку дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function GetStartPoint2() As Object**

Возвращаемое значение:

объект класса **SketchPoint**.

Действие: получить радиус дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function GetRadius() As Double**

Действие: получить направление дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function GetRotationDir() As Integer**

Возвращаемое значение:

если значение равно 1, то направление противоположно ходу часовой стрелки, если значение равно -1, то направление совпадает с ходом часовой стрелки.

Действие: установить радиус дуги окружности эскиза.

Доступ: метод класса **SketchArc**.

Описание:

**Public Function SetRadius**( \_

**ByVal** Radius **As Double** \_

) **As Boolean**

Аргументы:

Radius – радиус.

Возвращаемое значение:

если **True**, то радиус установлен.

Действие: получить количество точек сплайна эскиза.

Доступ: метод класса **SketchSpline**.

Описание:

**Public Function GetPointCount() As Long**

Действие: получить точки сплайна эскиза.

Доступ: метод класса **SketchSpline**.

Описание:

**Public Function GetPoints2() As Variant**

Возвращаемое значение:

одномерный массив объектов класса **SketchPoint**.

Действие: наложить взаимосвязи на элементы эскиза.

Доступ: метод класса **ModelDoc2**.

Описание:

**Public Sub SketchAddConstraints**( \_

**ByVal** IdStr **As String** \_

)

Аргументы:

IdStr – тип взаимосвязи.

Примечание:

Некоторые типы взаимосвязи:

sgHORIZONTAL2D – горизонтальность;

sgHORIZONTALPOINTS2D – расположение точек на одной горизонтали;

sgVERTICAL2D – вертикальность;

sgVERTPOINTS2D‑расположение точек на одной вертикали;

sgCOLINEAR – коллинеарность;

sgCORADIAL – корадиальность;

sgPERPENDICULAR – перпендикулярность;

sgPARALLEL – параллельность;

sgTANGENT – касательность;

sgCONCENTRIC – концентричность;

sgCOINCIDENT – совпадение;

sgSYMMETRIC – симметричность;

sgATMIDDLE – средняя точка;

sgATINTERSECT – точка пересечения;

sgATPIERCE – точка пронзания;

sgFIXED – фиксация;

sgSAMELENGTH – одинаковая длина.

Действие: обрезать элемент эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **SketchTrim**( \_

**ByVal** Option **As** **Long**, \_

**ByVal** X **As** **Double**, \_

**ByVal** Y **As** **Double**, \_

**ByVal** Z **As** **Double** \_

) **As** **Boolean**

Аргументы:

Option – опция – значение из перечисления **swSketchTrimChoice\_e**;

X, Y, Z – координаты точки элемента эскиза.

Возвращаемое значение:

**True** – если элемент эскиза был обрезан.

Примечание: X, Y и Z имеют смысл в том случае, когда аргумент Option равен или s**wSketchTrimClosest**, или **swSketchTrimEntityPoint**.

Элементы перечисления **swSketchTrimChoice\_e**:

**swSketchTrimClosest** – обрезать до ближайшего элемента эскиза;

**swSketchTrimCorner** – обрезать угол;

**swSketchTrimTwoEntities**;

**swSketchTrimEntityPoint**;

**swSketchTrimEntities**;

**swSketchTrimOutside** – обрезать снаружи;

**swSketchTrimInside** – обрезать внутри.

Действие: удлинить элемент эскиза.

Доступ: метод класса **SketchManager**.

Описание:

**Public Function** **SketchExtend**( \_

**ByVal** X **As** **Double**, \_

**ByVal** Y **As** **Double**, \_

**ByVal** Z **As** **Double** \_

) **As** **Boolean**

Аргументы:

X, Y, Z – координаты точки элемента эскиза.

Возвращаемое значение:

**True** – если элемент эскиза был удлинен.

Действие: отразить элемента эскиза.

Доступ: метод класса **ModelDoc2**.

Описание:

**Public Sub SketchMirror()**

Примечание:

при выделении сегменту эскиза присваивается номер 2, а оси – номер 1.

**9. Создание элементов дерева построения**

Действие: создать бобышку вытягиванием.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureExtrusion2**(

**ByVal** Sd **As** **Boolean**, \_

**ByVal** Flip **As** **Boolean**, \_

**ByVal** Dir **As** **Boolean**, \_

**ByVal** T1 **As** **Integer**, \_

**ByVal** T2 **As** **Integer**, \_

**ByVal** D1 **As** **Double**, \_

**ByVal** D2 **As** **Double**, \_

**ByVal** Dchk1 **As** **Boolean**, \_

**ByVal** Dchk2 **As** **Boolean**, \_

**ByVal** Ddir1 **As** **Boolean**, \_

**ByVal** Ddir2 **As** **Boolean**, \_

**ByVal** Dang1 **As** **Double**, \_

**ByVal** Dang2 **As** **Double**, \_

**ByVal** OffsetReverse1 **As** **Boolean**, \_

**ByVal** OffsetReverse2 **As** **Boolean**, \_

**ByVal** TranslateSurface1 **As** **Boolean**, \_

**ByVal** TranslateSurface2 **As** **Boolean**, \_

**ByVal** Merge **As** **Boolean**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** T0 **As** **Integer**, \_

**ByVal** StartOffset **As** **Double**, \_

**ByVal** FlipStartOffset **As** **Boolean** \_

) **As** **Feature**

Аргументы:

если аргумент Sd равен **True**, то вытягивание осуществляется в одном (первом) направлении;

если аргумент Flip равен **True**, то направление будет реверсировано;

если аргумент Dir равен **True**, то направление будет реверсировано;

T1 – значение из перечисления **swEndConditions\_e**, которое задает тип ограничения в первом направлении;

T2 – значение из перечисления **swEndConditions\_e**, которое задает тип ограничения во втором направлении;

D1 – смещение в первом направлении;

D2 – смещение во втором направлении;

если аргумент Dchk1 равен **True**, то вытягивание в первом направлении будет осуществляться с уклоном;

если аргумент Dchk2 равен **True**, то вытягивание во втором направлении будет осуществляться с уклоном;

если аргумент Ddir1 равен **True**, то уклон в первом направлении будет осуществлен внутрь;

если аргумент Ddir2 равен **True**, то уклон во втором направлении будет осуществлен внутрь;

Dang1 – уклон в первом направлении;

Dang2 ‑ уклон во втором направлении;

если аргумент OffsetReverse1 равен **True**, то для случая, когда ограничение первого направления задается поверхностью, смещенной на расстояние D1 относительно заданной поверхности, смещение будет осуществлено в направлении от эскиза;

если аргумент OffsetReverse2 равен **True**, то для случая, когда ограничение второго направления задается поверхностью, смещенной на расстояние D2 относительно заданной поверхности, смещение будет осуществлено в направлении от эскиза;

если аргумент TranslateSurface1 равен **True**, то для случая, когда аргумент T1 равен **swEndCondOffsetFromSurface**, вытягивание в первом направлении будет осуществляться до поверхности, которая смещена относительно заданной поверхности на величину D1;

если аргумент TranslateSurface2 равен **True**, то для случая, когда аргумент T2 равен **swEndCondOffsetFromSurface**, вытягивание во втором направлении будет осуществляться до поверхности, которая смещена относительно заданной поверхности на величину D2;

если аргумент Merge равен **True**, то бобышка будет объединена с пересекающими ее телами;

если аргумент UseFeatScope равен **True**, то бобышка будет объединена только с выделенными пересекающими ее телами, если **False** – то бобышка может быть автоматически объединена со всеми пересекающими ее телами как уже существующими, так и созданными потом;

если аргумент UseAutoSelect равен **True**, то тела, пересекающие бобышку, будут выделены автоматически, если **False**, то эти тела должны быть выделены;

T0 – значение из перечисления **swStartConditions\_e**, которое задает начальные условия вытягивания;

StartOffset – смещение, если аргумент T0 равен **swStartOffset**;

если аргумент FlipStartOffset равен **True**, то в том случае, когда аргумент T0 равен **swStartOffset**, направление смещения будет реверсировано.

Элементы перечисления **swEndConditions\_e**:

**swEndCondBlind** – «вслепую»;

**swEndCondThroughAll** – насквозь;

**swEndCondThroughNext** – через следующее тело;

**swEndCondUpToVertex** – до вершины;

**swEndCondUpToSurface** – до поверхности;

**swEndCondOffsetFromSurface** – расстояние от поверхности;

**swEndCondMidPlane** – средняя плоскость;

**swEndCondUpToBody** – до тела.

Элементы перечисления **swStartConditions\_e**:

**swStartSketchPlane** – плоскость эскиза;

**swStartSurface** ‑ поверхность;

**swStartVertex** ‑ вершина;

**swStartOffset** ‑ смещение.

Действие: создать бобышку по сечениям.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertProtrusionBlend2**( \_

**ByVal** Closed **As** **Boolean**, \_

**ByVal** KeepTangency **As** **Boolean**, \_

**ByVal** ForceNonRational **As** **Boolean**, \_

**ByVal** TessToleranceFactor **As** **Double**, \_

**ByVal** StartMatchingType **As** **Short**, \_

**ByVal** EndMatchingType **As** **Short**, \_

**ByVal** StartTangentLength **As** **Double**, \_

**ByVal** EndTangentLength **As** **Double**, \_

**ByVal** StartTangentDir **As** **Boolean**, \_

**ByVal** EndTangentDir **As** **Boolean**, \_

**ByVal** IsThinBody **As** **Boolean**, \_

**ByVal** Thickness1 **As** **Double**, \_

**ByVal** Thickness2 **As** **Double**, \_

**ByVal** ThinType **As** **Short**, \_

**ByVal** Merge **As** **Boolean**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** GuideCurveInfluence **As** **Integer** \_

) **As** **Feature**

Аргументы:

если аргумент Closed равен **True**, то направляющая кривая будет автоматически замкнута;

если аргумент KeepTangency равен **True**, то будут образованы касательные поверхности, если сечение состоит из касательных сегментов;

если аргумент ForceNonRational равен **True**, то, когда сечение имеет круговые или эллиптические сегменты, будет осуществляться их аппроксимация для лучшего сглаживания;

TessToleranceFactor – коэффициент, устанавливающий количество промежуточных сечений (по умолчанию равен 1, чем больше, тем больше будет создано промежуточных сечений);

StartMatchingType – тип касания направляющей кривой с исходным сечением ‑ значение из перечисления **swTangencyType\_e**;

EndMatchingType – тип касания направляющей кривой с конечным сечением ‑ значение из перечисления **swTangencyType\_e**;

StartTangentLength – длина вектора, касательного к поверхности, в исходном сечении, если аргумент StartMatchingType равен **swTangencyDirectionVector** или **swTangencyNormalToProfile**;

EndTangentLength – длина вектора, касательного к поверхности, в конечном сечении, если аргумент EndMatchingType равен **swTangencyDirectionVector** или **swTangencyNormalToProfile**;

если аргумент StartTangentDir равен **False**, то направление касательного вектора в исходном сечении будет реверсировано;

если аргумент EndTangentDir равен **False**, то направление касательного вектора в конечном сечении будет реверсировано;

если аргумент IsThinBody равен **True**, то будет создано тонкостенное тело;

Thickness1 – толщина тонкостенного тела, если аргумент ThinType равен **swThinWallOneDirection** или **swThinWallOppDirection**, или **swThinWallMidPlane**; толщина тонкостенного тела в первом направлении, если аргумент ThinType равен **swThinWallTwoDirection**;

Thickness2 – толщина тонкостенного тела во втором направлении, если аргумент ThinType равен **swThinWallTwoDirection**;

ThinType – тип толщины тонкостенного тела – значение из перечисления **swThinWallType\_e**;

Merge ‑ см. создать вытянутую бобышку;

UseFeatScope ‑ см. создать вытянутую бобышку;

UseAutoSelect ‑ см. создать вытянутую бобышку;

GuideCurveInfluence – тип влияния направляющей кривой на сегменты сечения – значение из перечисления **swGuideCurveInfluence\_e**.

Примечание:

1) для выделения необходимо использовать метод **SelectByID2**; сечению присваивается номер 1, направляющей кривой – номер 2, центровой линии – номер 4, начальному направлению касания – номер 8, конечному направлению касания – номер 32;

2) аргументы StartMatchingType и EndMatchingType могут принимать только следующие значения: **swTangencyNone**, **swTangencyNormalToProfile**, **swTangencyDirectionVector**, **swTangencyAllFaces**.

Элементы перечисления **swTangencyType\_e**:

**swTangencyNone** – направление касательной к направляющей кривой не задано;

**swTangencyNormalToProfile** – направление касательной к направляющей кривой по нормали к сечению;

**swTangencyDirectionVector** – направление касательной к направляющей кривой по заданному направлению;

**swTangencyAllFaces** – к направляющей кромке грани добавляются кромки соседних граней;

**swMinimumTwist** – направление касательной к направляющей кривой будет обеспечивать минимальное скручивание сечения, устраняющее пересечение сечений.

Элементы перечисления **swThinWallType\_e**:

**swThinWallOneDirection** – толщина тонкостенного тела в первом направлении;

**swThinWallOppDirection** – толщина тонкостенного тела во втором направлении;

**swThinWallMidPlane** – одинаковая толщина тонкостенного тела в двух направлениях;

**swThinWallTwoDirection** – разная толщина тонкостенного тела в двух направлениях.

Элементы перечисления **swGuideCurveInfluence\_e**:

**swGuideCurveInfluenceNextGuide** – до следующей направляющей кривой;

**swGuideCurveInfluenceNextSharp** – до следующего острого угла сечения;

**swGuideCurveInfluenceNextEdge** – до следующего сегмента сечения;

**swGuideCurveInfluenceNextGlobal** – влияние на все сечение.

Действие: создать бобышку заметанием.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertProtrusionSwept3**( \_

**ByVal** Propagate **As** **Boolean**, \_

**ByVal** Alignment **As** **Boolean**, \_

**ByVal** TwistCtrlOption **As** **Short**, \_

**ByVal** KeepTangency **As** **Boolean**, \_

**ByVal** BAdvancedSmoothing **As** **Boolean**, \_

**ByVal** StartMatchingType **As** **Short**, \_

**ByVal** EndMatchingType **As** **Short**, \_

**ByVal** IsThinBody **As** **Boolean**, \_

**ByVal** Thickness1 **As** **Double**, \_

**ByVal** Thickness2 **As** **Double**, \_

**ByVal** ThinType **As** **Short**, \_

**ByVal** PathAlign **As** **Short**, \_

**ByVal** Merge **As** **Boolean**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** TwistAngle **As** **Double**, \_

**ByVal** BMergeSmoothFaces **As** **Boolean** \_

) **As** **Feature**

Аргументы:

если аргумент Propagate равен **True**, то бобышка будет распространена вдоль следующей кромки, касательной к выбранной кромке как пути;

если аргумент Alignment равен **True**, то бобышка может выходить за пределы существующих тел, с которыми она пересекается и на которые она воздействует;

TwistCtrlOption – опция контроля скручивания – значение из перечисления **swTwistControlType\_e**;

если аргумент KeepTangency равен **True**, то, когда сечение имеет касательные сегменты, получаемые поверхности также будут касательными;

если аргумент BAdvancedSmoothing равен **True**, то, когда сечение имеет круговые или эллиптические сегменты, будет осуществляться их аппроксимация для лучшего сглаживания;

StartMatchingType – тип касания направляющей кривой с исходным сечением;

EndMatchingType – тип касания направляющей кривой с конечным сечением;

IsThinBody ‑ см. создать бобышку по сечениям;

Thickness1 ‑ см. создать бобышку по сечениям;

Thickness2 ‑ см. создать бобышку по сечениям;

ThinType ‑ см. создать бобышку по сечениям;

PathAlign – тип выравнивания сечения, которое имеет смысл, когда аргумент TwistCtrlOption равен **swTwistControlFollowPath**;

Merge ‑ см. создать бобышку по сечениям;

UseFeatScope ‑ см. создать бобышку по сечениям;

UseAutoSelect ‑ см. создать бобышку по сечениям;

TwistAngle – угол скручивания конечного сечения, если аргумент TwistCtrlOption равен **swTwistControlConstantTwistAlongPath**;

если аргумент BMergeSmoothFaces рвен **True**, то гладкие грани будут объединены.

Примечание:

1) для выделения необходимо использовать метод **SelectByID2**; сечению присваивается номер 1, направляющей кривой – номер 2, пути – номер 4;

2) аргумент PathAlign может принимать только следующие значения: 0 – выравнивание сечения относительно нормали к пути; 2 ‑ выравнивание сечения относительно заданного направления; 3 – к пути в виде кромки грани добавляются кромки смежных граней, выравнивание сечения относительно нормали к пути;

3) аргументы StartMatchingType и EndMatchingType могут принимать только следующие значения: 0 – направление касательной к направляющей кривой не задано, 2 ‑ направление касательной к направляющей кривой по направлению касательной к пути.

Элементы перечисления **swTwistControlType\_e**:

**swTwistControlFollowPath** – сечение перемещается вдоль касательной к пути;

**swTwistControlKeepNormalConstant** – выравнивание сечения относительно нормали к исходному сечению;

**swTwistControlFollowPathFirstGuideCurve** – выравнивание сечения относительно нормали к пути, профилирование в соответствии с одной направляющей кривой;

**swTwistControlFollowFirstSecondGuideCurves** – выравнивание сечения относительно нормали к пути, профилирование в соответствии с двумя направляющими кривыми;

**swTwistControlConstantTwistAlongPath** – направление нормали к сечению постоянно относительно касательной к пути, равномерное скручивание сечения относительно этой касательной на заданный угол.

Действие: создать бобышку поворотом.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureRevolve2**( \_

**ByVal** SingleDir **As** **Boolean**, \_

**ByVal** IsSolid **As** **Boolean**, \_

**ByVal** IsThin **As** **Boolean**, \_

**ByVal** IsCut **As** **Boolean**, \_

**ByVal** ReverseDir **As** **Boolean**, \_

**ByVal** BothDirectionUpToSameEntity **As** **Boolean**, \_

**ByVal** Dir1Type **As** **Integer**, \_

**ByVal** Dir2Type **As** **Integer**, \_

**ByVal** Dir1Angle **As** **Double**, \_

**ByVal** Dir2Angle **As** **Double**, \_

**ByVal** OffsetReverse1 **As** **Boolean**, \_

**ByVal** OffsetReverse2 **As** **Boolean**, \_

**ByVal** OffsetDistance1 **As** **Double**, \_

**ByVal** OffsetDistance2 **As** **Double**, \_

**ByVal** ThinType **As** **Integer**, \_

**ByVal** ThinThickness1 **As** **Double**, \_

**ByVal** ThinThickness2 **As** **Double**, \_

**ByVal** Merge **As** **Boolean**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean** \_

) **As** **Feature**

Аргументы:

если аргумент SingleDir равен **True**, то бобышка будет создана в одном (первом) направлении;

если аргумент IsSolid равен **True**, то будет создано твердое тело;

если аргумент IsThin равен **True**, то будет создано тонкостенное тело;

если аргумент IsCut равен **True**, то будет создан вырез поворотом;

если аргумент ReverseDir равен **True**, то будет осуществлено реверсирование направления поворота;

если аргумент BothDirectionUpToSameEntity равен **True**, то бобышка будет создана в обоих направлениях, причем аргумент SingleDir должен быть равен **False**, а каждый из аргументов Dir1Type и Dir2Type должен быть равен или **swEndCondUpToVertex**, или **swEndCondUpToSurface**, или **swEndCondOffsetFromSurface**;

Dir1Type ‑ тип ограничения направления 1 – значение из перечисления **swEndConditions\_e**;

Dir2Type ‑ тип ограничения направления 2 – значение из перечисления **swEndConditions\_e**;

Dir1Angle – угол поворота в радианах в направлении 1;

Dir2Angle ‑ поворота в радианах в направлении 2;

OffsetReverse1 – см. создать бобышку вытягиванием;

OffsetReverse2 – см. создать бобышку вытягиванием;

OffsetDistance1 – смещение в направлении 1, если аргумент Dir1Type равен **swEndCondOffsetFromSurface**;

OffsetDistance2 – смещение в направлении 2, если аргумент Dir2Type равен **swEndCondOffsetFromSurface**;

ThinType ‑ см. создать бобышку по сечениям;

ThinThickness1 ‑ см. создать бобышку по сечениям;

ThinThickness2 ‑ см. создать бобышку по сечениям;

Merge ‑ см. создать бобышку вытягиванием;

UseFeatScope ‑ см. создать бобышку вытягиванием;

UseAutoSelect ‑ см. создать бобышку вытягиванием.

Примечание:

1) выделение должно осуществляться методом **SelectByID2**, причем оси присваивается номер 16.

2) аргументы Dir1Type и Dir2Type могут принимать только следующие значения: **swEndCondBlind**, **swEndCondUpToVertex**, **swEndCondUpToSurface**, **swEndCondOffsetFromSurface** и **swEndCondMidPlane**.

Действие: создать вырез вытягиванием.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureCut3**( \_

**ByVal** Sd **As** **Boolean**, \_

**ByVal** Flip **As** **Boolean**, \_

**ByVal** Dir **As** **Boolean**, \_

**ByVal** T1 **As** **Integer**, \_

**ByVal** T2 **As** **Integer**, \_

**ByVal** D1 **As** **Double**, \_

**ByVal** D2 **As** **Double**, \_

**ByVal** Dchk1 **As** **Boolean**, \_

**ByVal** Dchk2 **As** **Boolean**, \_

**ByVal** Ddir1 **As** **Boolean**, \_

**ByVal** Ddir2 **As** **Boolean**, \_

**ByVal** Dang1 **As** **Double**, \_

**ByVal** Dang2 **As** **Double**, \_

**ByVal** OffsetReverse1 **As** **Boolean**, \_

**ByVal** OffsetReverse2 **As** **Boolean**, \_

**ByVal** TranslateSurface1 **As** **Boolean**, \_

**ByVal** TranslateSurface2 **As** **Boolean**, \_

**ByVal** NormalCut **As** **Boolean**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** AssemblyFeatureScope **As** **Boolean**, \_

**ByVal** AutoSelectComponents **As** **Boolean**, \_

**ByVal** PropagateFeatureToParts **As** **Boolean**, \_

**ByVal** T0 **As** **Integer**, \_

**ByVal** StartOffset **As** **Double**, \_

**ByVal** FlipStartOffset **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Sd ‑ см. создать бобышку вытягиванием;

Flip ‑ см. создать бобышку вытягиванием;

Dir ‑ см. создать бобышку вытягиванием;

T1 – см. создать бобышку вытягиванием;

T2 – см. создать бобышку вытягиванием;

D1 – см. создать бобышку вытягиванием;

D2 – см. создать бобышку вытягиванием;

Dchk1 ‑ см. создать бобышку вытягиванием;;

Dchk2 ‑ см. создать бобышку вытягиванием;

Ddir1 ‑ см. создать бобышку вытягиванием;

Ddir2 ‑ см. создать бобышку вытягиванием;

Dang1 – см. создать бобышку вытягиванием;

Dang2 ‑ см. создать бобышку вытягиванием;

OffsetReverse1 ‑ см. создать бобышку вытягиванием;

OffsetReverse2 ‑ см. создать бобышку вытягиванием;

TranslateSurface1 ‑ см. создать бобышку вытягиванием;

TranslateSurface2 ‑ см. создать бобышку вытягиванием;

аргумент NormalCut равен **True**, если вытянутый вырез создается перпендикулярно листу, **False** – если тело не является листом;

UseFeatScope ‑ см. создать бобышку вытягиванием;

UseAutoSelect ‑ см. создать бобышку вытягиванием;

если аргумент AssemblyFeatureScope равен **True**, то вытянутый вырез может затрагивать только выделенные компоненты сборки, если **False** – то вытянутый вырез может затрагивать все компоненты сборки, как существующие, так и добавленные потом;

если аргумент AutoSelectComponents равен **True**, то компоненты сборки, пересекающие бобышку, будут выделены автоматически, если **False**, то эти компоненты должны быть выделены;

если аргумент PropagateFeatureToParts равен **True**, то вытянутый вырез будет передан в соответствующие файлы компонентов сборки.

T0 – см. создать бобышку вытягиванием;

StartOffset – см. создать бобышку вытягиванием;

FlipStartOffset ‑ см. создать бобышку вытягиванием.

Действие: создать вырез по сечениям.

Доступ: Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertCutBlend**( \_

**ByVal** Closed **As** **Boolean**, \_

**ByVal** KeepTangency **As** **Boolean**, \_

**ByVal** ForceNonRational **As** **Boolean**, \_

**ByVal** TessToleranceFactor **As** **Double**, \_

**ByVal** StartMatchingType **As** **Short**, \_

**ByVal** EndMatchingType **As** **Short**, \_

**ByVal** IsThinBody **As** **Boolean**, \_

**ByVal** Thickness1 **As** **Double**, \_

**ByVal** Thickness2 **As** **Double**, \_

**ByVal** ThinType **As** **Short**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Closed ‑ см. создать бобышку по сечениям;

KeepTangency ‑ см. создать бобышку по сечениям;

ForceNonRational ‑ см. создать бобышку по сечениям;

TessToleranceFactor ‑ см. создать бобышку по сечениям;

StartMatchingType ‑ см. создать бобышку по сечениям;

EndMatchingType ‑ см. создать бобышку по сечениям;

IsThinBody ‑ см. создать бобышку по сечениям;

Thickness1 ‑ см. создать бобышку по сечениям;

Thickness2 ‑ см. создать бобышку по сечениям;

ThinType ‑ см. создать бобышку по сечениям;

UseFeatScope ‑ см. создать бобышку вытягиванием;

UseAutoSelect ‑ см. создать бобышку вытягиванием.

Примечания:

см. создать бобышку по сечениям.

Действие: создать вырез заметанием.

Доступ: Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertCutSwept4**( \_

**ByVal** Propagate **As** **Boolean**, \_

**ByVal** Alignment **As** **Boolean**, \_

**ByVal** TwistCtrlOption **As** **Short**, \_

**ByVal** KeepTangency **As** **Boolean**, \_

**ByVal** BAdvancedSmoothing **As** **Boolean**, \_

**ByVal** StartMatchingType **As** **Short**, \_

**ByVal** EndMatchingType **As** **Short**, \_

**ByVal** IsThinBody **As** **Boolean**, \_

**ByVal** Thickness1 **As** **Double**, \_

**ByVal** Thickness2 **As** **Double**, \_

**ByVal** ThinType **As** **Short**, \_

**ByVal** PathAlign **As** **Short**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** TwistAngle **As** **Double**, \_

**ByVal** BMergeSmoothFaces **As** **Boolean**, \_

**ByVal** AssemblyFeatureScope **As** **Boolean**, \_

**ByVal** AutoSelectComponents **As** **Boolean**, \_

**ByVal** PropagateFeatureToParts **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Propagate ‑ см. создать бобышку заметанием;

Alignment ‑ см. создать бобышку заметанием;

TwistCtrlOption ‑ см. создать бобышку заметанием;

KeepTangency ‑ см. создать бобышку заметанием;

BAdvancedSmoothing ‑ см. создать бобышку заметанием;

StartMatchingType ‑ см. создать бобышку заметанием;

EndMatchingType ‑ см. создать бобышку заметанием;

IsThinBody ‑ см. создать бобышку по сечениям;

Thickness1 ‑ см. создать бобышку по сечениям;

Thickness2 ‑ см. создать бобышку по сечениям;

ThinType ‑ см. создать бобышку по сечениям;

PathAlign ‑ см. создать бобышку заметанием;

UseFeatScope ‑ см. создать бобышку вытягиванием;

UseAutoSelect ‑ см. создать бобышку вытягиванием;

TwistAngle ‑ см. создать бобышку заметанием;

BMergeSmoothFaces ‑ см. создать бобышку заметанием;

AssemblyFeatureScope ‑ см. создать вырез вытягиванием;

AutoSelectComponents ‑ см. создать вырез вытягиванием;

PropagateFeatureToParts ‑ см. создать вырез вытягиванием.

Примечания:

см. создать бобышку заметанием.

Действие: создать вырез поворотом.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureRevolveCut2**( \_

**ByVal** Angle **As** **Double**, \_

**ByVal** ReverseDir **As** **Boolean**, \_

**ByVal** Angle2 **As** **Double**, \_

**ByVal** RevType **As** **Integer**, \_

**ByVal** Options **As** **Integer**, \_

**ByVal** UseFeatScope **As** **Boolean**, \_

**ByVal** UseAutoSelect **As** **Boolean**, \_

**ByVal** AssemblyFeatureScope **As** **Boolean**, \_

**ByVal** AutoSelectComponents **As** **Boolean**, \_

**ByVal** PropagateFeatureToParts **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Angle – угол поворота в радианах в направлении 1;

если аргумент ReverseDir равен **True**, то направление поворота будет реверсировано;

Angle2 ‑ угол поворота в радианах в направлении 2;

RevType – тип вращения – значение из перечисления **swRevolveType\_e**;

Options – опции – значение **swAutoCloseSketch**, установка которого обусловливает выход из эскиза, если он активен;

UseFeatScope ‑ см. создать бобышку вытягиванием;

UseAutoSelect ‑ см. создать бобышку вытягиванием;

AssemblyFeatureScope ‑ см. создать вырез вытягиванием;

AutoSelectComponents ‑ см. создать вырез вытягиванием;

PropagateFeatureToParts ‑ см. создать вырез вытягиванием.

Действие: создать скругление.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureFillet**( \_

**ByVal** Options **As** **Integer**, \_

**ByVal** R1 **As** **Double**, \_

**ByVal** Ftyp **As** **Integer**, \_

**ByVal** OverflowType **As** **Integer**, \_

**ByVal** Radii **As** **Object**, \_

**ByVal** SetBackDistances **As** **Object**, \_

**ByVal** PointRadiusArray **As** **Object** \_

) **As** **Object**

Аргументы:

Options – опции – значение из перечисления **swFeatureFilletOptions\_e**;

R1 – радиус скругления, если аргумент Ftyp равен **swFeatureFilletType\_Simple**;

Ftyp – тип скругления – значение из перечисления **swFeatureFilletType\_e**;

OverflowType – тип ограничения скругления существующей кромкой – значение из перечисления **swFilletOverFlowType\_e**;

Radii – массив, содержащий значения радиусов скругления в вершинах кромки, если аргумент Ftyp равен **swFeatureFilletType\_VariableRadius**, или значения радиусов скругления разных кромок;

SetBackDistances – массив, содержащий расстояния от вершины вдоль кромки, определяющие положения точек, в которых задается уменьшающийся радиус скругления;

PointRadiusArray – массив, содержащий радиусы скругления в заданных справочных точках кромки.

Примечание:

1) при выделении граней первой грани присваивается номер 2, второй грани присваивается номер 4;

2) при создании полного скругления дополнительно к п. 1) центральной грани присваивается номер 512;

3) если аргумент Ftyp не равен **swFeatureFilletType\_Face**, то опции **swFeatureFilletUseHelpPoint**, **swFeatureFilletUseTangentHoldLine**, **swFeatureFilletCurvatureContinuous**, **swFeatureFilletConstantWidth**, **swFeatureFilletNoTrimNoAttached** не задаются;

4) если аргумент Ftyp равен **swFeatureFilletType\_VariableRadius**, то необходимо задать справочные точки.

Элементы перечислеия **swFeatureFilletOptions\_e**:

**swFeatureFilletPropagate** – распространение скругления на всю цепочку кромок, в которой соседние кромки являются касательными;

**swFeatureFilletUniformRadius** – скругление одним и тем же радиусом разных кромок;

**swFeatureFilletVarRadiusType** – скругление и прилежащая грань не являются касательными;

**swFeatureFilletUseHelpPoint** – использование вспомогательной точки;

**swFeatureFilletUseTangentHoldLine** – использование кривой сопряжения скругления и грани;

**swFeatureFilletCornerType** – кромка между смежными скруглениями не скругляется;

**swFeatureFilletAttachEdges** – присоединение кромок;

**swFeatureFilletKeepFeatures** – сохранение элементов дерева построения, которые попадают в область действия скругления;

**swFeatureFilletCurvatureContinuous** – непрерывная кривизна скругления граней;

**swFeatureFilletConstantWidth** – постоянная ширина скругления;

**swFeatureFilletNoTrimNoAttached** – после добавления переходного скругления грани не обрезаются и не соединяется;

**swFeatureFilletReverseFace1Dir** – реверсирование нормали к грани 1;

**swFeatureFilletReverseFace2Dir** ‑ реверсирование нормали к грани 2;

**swFeatureFilletPropagateFeatToParts** – скругление передается в соответствующие файлы компонентов сборки.

Элементы перечислеия **swFeatureFilletType\_e**:

**swFeatureFilletType\_Simple** – простое скругление;

**swFeatureFilletType\_VariableRadius** – скругление переменного радиуса;

**swFeatureFilletType\_Face** – добавление переходного скругления между парой граней;

**swFeatureFilletType\_FullRound** – полное скругление.

Элементы перечислеия **swFilletOverFlowType\_e**:

**swFilletOverFlowType\_Default** – по умолчанию;

**swFilletOverFlowType\_KeepEdge** – если скругление ограничивается существующей кромкой, то скругление искажается вдоль кромки, а кромка не искажается;

**swFilletOverFlowType\_KeepSurface** – если скругление ограничивается существующей кромкой, то скругление не искажается, а искажается участок кромки, ограничивающий скругление.

Действие: создать фаску.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **InsertFeatureChamfer**( \_

**ByVal** Options **As** **Integer**, \_

**ByVal** ChamferType **As** **Integer**, \_

**ByVal** Width **As** **Double**, \_

**ByVal** Angle **As** **Double**, \_

**ByVal** OtherDist **As** **Double**, \_

**ByVal** VertexChamDist1 **As** **Double**, \_

**ByVal** VertexChamDist2 **As** **Double**, \_

**ByVal** VertexChamDist3 **As** **Double** \_

) **As** **Feature**

Аргументы:

Options – опции – значение из перечисления **swFeatureChamferOption\_e**;

ChamferType – тип фаски – значение из перечисления **swChamferType\_e**;

Width – ширина фаски, если аргумент ChamferType равен s**wChamferAngleDistance**;

Angle – угол, если аргумент ChamferType равен s**wChamferAngleDistance**;

OtherDist – другие расстояния, если аргумент ChamferType равен **swChamferDistanceDistance** или **swChamferVertex**;

VertexChamDist1 – расстояние на первой стороне, если аргумент ChamferType равен **swChamferDistanceDistance** или **swChamferVertex**;

VertexChamDist2 ‑ расстояние на второй стороне, если аргумент ChamferType равен **swChamferDistanceDistance** или **swChamferVertex**;

VertexChamDist3 ‑ расстояние на третьей стороне, если аргумент ChamferType равен **swChamferDistanceDistance** или **swChamferVertex**.

Элементы перечисления **swFeatureChamferOption\_e**:

**swFeatureChamferFlipDirection** – реверсировать направление;

**swFeatureChamferKeepFeature** – сохранить элементы дерева построения;

**swFeatureChamferTangentPropagation** – распространить по касательной;

s**wFeatureChamferPropagateFeatToParts** ‑ передать фаску в соответствующие файлы компонентов сборки.

Элементы перечисления **swChamferType\_e**:

**swChamferAngleDistance** – угол и расстояние;

**swChamferDistanceDistance** – расстояние и расстояние;

**swChamferVertex** ‑ вершина;

**swChamferEqualDistance** – равные расстояния.

Действие: создать линейный массив.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureLinearPattern2**( \_

**ByVal** Num1 **As** **Integer**, \_

**ByVal** Spacing1 **As** **Double**, \_

**ByVal** Num2 **As** **Integer**, \_

**ByVal** Spacing2 **As** **Double**, \_

**ByVal** FlipDir1 **As** **Boolean**, \_

**ByVal** FlipDir2 **As** **Boolean**, \_

**ByVal** DName1 **As** **String**, \_

**ByVal** DName2 **As** **String**, \_

**ByVal** GeometryPattern **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Num1 – количество элементов в линейном массиве в направлении 1, включая исходный элемент;

Spacing1 – расстояние между элементами в линейном массиве в направлении 1;

Num2 ‑ количество элементов в линейном массиве в направлении 2, включая исходный элемент;

Spacing2 ‑ расстояние между элементами в линейном массиве в направлении 2;

если аргумент FlipDir1 равен **True**, то направление 1 будет реверсировано;

если аргумент FlipDir2 равен **True**, то направление 2 будет реверсировано;

DName1 – имя размера в направлении 1;

DName2 ‑ имя размера в направлении 2;

если аргумент GeometryPattern равен **True**, то будет создан геометрический массив.

Примечание:

1) если выделяются исходные элементы массива в виде элементов дерева построения, то им присваивается порядковый номер 4, а выделяемым направлениям – номера 1 и 2;

2) если выделяются исходные элементы массива в виде компонентов сборки, то им присваивается порядковый номер 1, а выделяемым направлениям – номера 2 и 4.

Действие: создать круговой массив.

Доступ: метод класса **FeatureManager**.

Описание:

**Function** **FeatureCircularPattern3**( \_

**ByVal** Number **As** **Integer**, \_

**ByVal** Spacing **As** **Double**, \_

**ByVal** FlipDirection **As** **Boolean**, \_

**ByVal** DName **As** **String**, \_

**ByVal** GeometryPattern **As** **Boolean**, \_

**ByVal** EqualSpacing **As** **Boolean** \_

) **As** **Feature**

Аргументы:

Number – количество элементов в круговом массиве, включая исходный элемент;

Spacing – угловое (в радианах) расстояние между элементами в круговом массиве или общий угол (в радианах), если аргумент **EqualSpacing** равен **True**;

если аргумент FlipDirection равен **True**, то направление будет реверсировано;

DName – имя углового размера;

если аргумент GeometryPattern равен **True**, то будет создан геометрический массив;

если аргумент EqualSpacing равен **True**, то элементы в круговом массиве будут распределены равномерно.

Примечание:

при выделении исходных элементов в виде элементов дерева построения им необходимо присвоить порядковый номер 4, а при выделении исходных элементов в виде компонентов сборки им присваивается порядковый номер 1, при этом при выделении оси ей присваивается порядковый номер 2.

Действие: создать массив, управляемый кривой.

Доступ: метод класса **ModelDoc2**.

Описание:

**Sub** **FeatureCurvePattern**( \_

**ByVal** Num1 **As** **Integer**, \_

**ByVal** Spacing1 **As** **Double**, \_

**ByVal** Num2 **As** **Integer**, \_

**ByVal** Spacing2 **As** **Double**, \_

**ByVal** FlipDir1 **As** **Boolean**, \_

**ByVal** FlipDir2 **As** **Boolean**, \_

**ByVal** EqualSpacing1 **As** **Boolean**, \_

**ByVal** EqualSpacing2 **As** **Boolean**, \_

**ByVal** UseCentroid **As** **Boolean**, \_

**ByVal** AlignToSeed **As** **Boolean**, \_

**ByVal** OffsetCurve **As** **Boolean**, \_

**ByVal** PatternSeedOnly **As** **Boolean** \_

)

Аргументы:

Num1 – количество элементов в направлении 1, включая исходный;

Spacing1 – шаг в направлении 1;

Num2 ‑ количество элементов в направлении 2;

Spacing2 ‑ шаг в направлении 2;

если аргумент FlipDir1 равен **True**, то направление 1 будет реверсировано;

если аргумент FlipDir2 равен **True**, то направление 2 будет реверсировано;

если аргумент EqualSpacing1 равен **True**, то в направлении 1 шаг будет одинаковый;

если аргумент EqualSpacing2 равен **True**, то в направлении 2 шаг будет одинаковый;

если аргумент UseCentroid равен **True**, то в качестве справочной точки будет использован центр;

если аргумент AlignToSeed равен **True**, то элементы будут выравниваться относительно исходного элемента, если **False**, то элементы будут выравниваться относительно касательной к кривой;

если аргумент OffsetCurve равен **True**, то расстояние от нормали к кривой до каждого элемента будет постоянным, если **False**, то расстояние от точки кривой до элемента будет постоянным;

если аргумент PatternSeedOnly равен **True**, то в направлении 2 будет скопирован только исходный элемент.