Оглавление

Объектная модель САПФИР....2
Объекты параметрической модели....2
Объекты чертежей:....3
Сервисные объекты приложения....3
Сервисные объекты документа....3
Сервисные объекты модели....3
Описание объектов автоматизации САПФИР....4
Типы объектов....4
Приложение (Application)....6
Документ (Document)....8
Здание (AutoProject)....12
Этаж (AutoStorey)....13
Параметрическая модель (AutoModel)....14
Блок (AutoObjBlock)....17
Обозначение (AutoObjDim)....19
Заштрихованная область (AutoObjPolygon)....20
Текст (AutoObjText)....21
Координационные оси (AutoAxes)....22
Помещение (AutoZone)....23
Графическая модель (AutoModel3D)....24
3D сетка граней (AutoMesh)....26
Графическая чертежная область (AutoPolygon)....28
Графическая полилиния (AutoPolyLine)....29
Графическая линия (AutoLine)....33
Графический объект 2D/3D текст (AutoText)....36
Графический объект размеры и обозначения (AutoDim)....38
Фабрика объектов (AutoFactory)....39
Диалог выбора файла (AutoFileDialog)....39
Вид документа (AutoSapfirView)....40
Диалог настройки именованных параметров (AutoParametersDlg)....40
Инструментальная панель (AutoLocatorUser)....41
Контейнер слоев (AutoSetLayers)....41
Слой (AutoLayer)....42
Набор материалов (AutoSetMaterials)....42
Материал (AutoMaterial)....43
Слой многослойного материала (AutoLayerMat)....44
Геометрический вычислитель (AutoSolver)....44
Генератор геометрических моделей (AutoSurfCreator)....45
Набор контуров (AutoVectContour)....46
Параметрическое сечение (AutoMultiContour)....46
Матрица (AutoMatrix)....47
Описание именованных параметров....49
Типы параметров....49
Свойства параметров....50
Примеры....51
Служебные папки САПФИР....53

Объектная модель САПФИР

Программа САПФИР поддерживает технологию COM Automation и предоставляет пользователю набор объектов автоматизации. Объектная модель Сапфир показана на рис.1

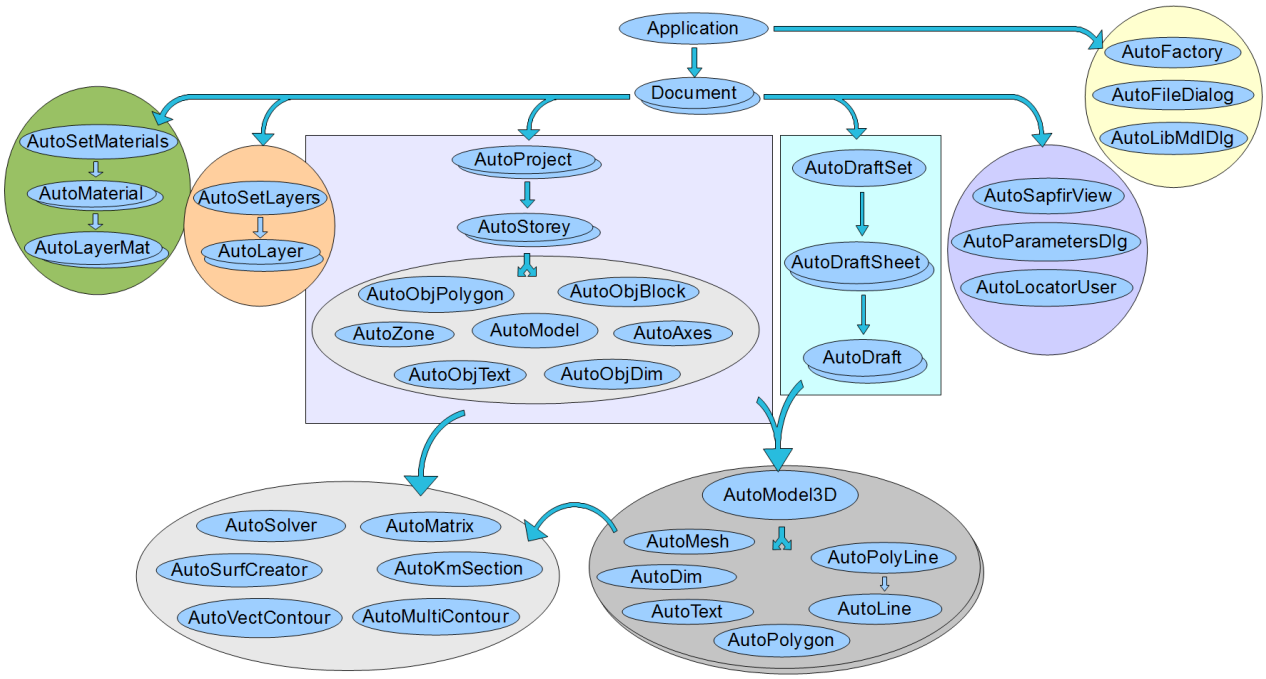

Рис.1.

Application - объект приложения САПФИР обеспечивает доступ к документам и сервисным объектам: фабрике объектов, диалогу выбора файлов и диалогу создания стандартных объектов.

Document – объект документ - контейнер проектов (зданий). Проектов в документе может быть несколько. Каждый документ сохраняется в отдельном файле с расширением «spf». Из документа можно получить доступ к объектам параметрической модели, к чертежам и сервисным объектам.

Объекты параметрической модели

Это основные объекты для построения параметрической модели проектируемого объекта. Доступ к ним возможен через объект документ.
AutoProject - объект **здание** - контейнер этажей.
AutoStorey - объект **этаж** - контейнер объектов

Объекты

AutoModel - **универсальный объект** - экземпляр объекта этажа: стена, балка, перекрытие, колонна, окно, дверь, лестница, призма, конус, линия, и др.
AutoObjBlock - специализированный объект **блок** - контейнер объектов.
AutoObjDim - специализированный объект **размеры и обозначения** - конкретный экземпляр таких объектов как: размерная линия, высотная отметка, угловой размер и др.
AutoObjPolygon - специализированный объект **полигон**
AutoObjText - специализированный объект **текст**
AutoAxes - специализированный объект **координационные оси**
AutoZone - специализированный объект **помещение**

** Специализированные объекты, в некоторых случаях, могут быть представлены универсальным объектом, например, возвращаются методом GetModelByIndex().*

Объекты чертежей:

AutoDraftSet – объект – **набор чертежных листов**
AutoDraftSheet – объект **чертежный лист**
AutoDraft – объект **чертеж**

Графические 2D и 3D объекты

Эти объекты используются для представления параметрических объектов в 3D и на чертежах.
AutoModel3D - графический объект **3D модель** - контейнер графических объектов
AutoMesh - объект **3D сетка** граней
AutoPolygon - графический объект **штриховка** - плоский замкнутый контур с заливкой цветом, текстурой и штриховкой
AutoPolyLine - графический объект **полилиния** - цепочка последовательных сегментов линий различных типов.
AutoLine - графический объект **линия** может быть одним из следующих типов - отрезок, дуга окружности, эллипса, сплайна или кривой Безье. Используется как сегмент полилинии или самостоятельно.
AutoText - графический объект 2D/3D **текст**
AutoDim - графический объект **размеры и обозначения**

Сервисные объекты приложения

AutoFactory - сервисный объект **фабрика объектов** используется для создания параметрических и графических объектов модели.
AutoFileDialog - сервисный объект **диалог выбора файла**, ориентированный на использование в САПФИР.
AutoLibMdlDlg - **сервисный диалог**, используемый для создания параметрических окон, дверей, сечений колонн и балок.

Сервисные объекты документа

AutoSapfirView - сервисный объект **вид документа**
AutoParametersDlg - сервисный объект **диалог настройки именованных параметров** объекта.
AutoLocatorUser - сервисный объект обеспечивающий доступ к текущей **инструментальной панели**.
AutoSetLayers - объект **контейнер слоев** документа.
AutoLayer - объект **слой**.
AutoSetMaterials - объект **набор материалов** документа.
AutoMaterial - объект **материал**.
AutoLayerMat - объект слой многослойного материала (конструкции)

Сервисные объекты модели

AutoSolver - объект **геометрический вычислитель**. Выполняет пересечение объектов (линий).
AutoSurfCreator - объект **генератор геометрических моделей**.
AutoVectContour - объект набор полилиний (контуров).
AutoKmSection - объект **генератор сечений сортамента стальных профилей**
AutoMultiContour - объект **параметрическое сечение** в балках и колоннах
AutoMatrix - объект **матрица**.

Описание объектов автоматизации САПФИР

Доступ к объектам автоматизации САПФИР и обращение к свойствам и методам объектов зависит от используемого языка программирования. Ниже будет показано использование объектов применительно к JavaScript. Все объекты автоматизации САПФИР имеют свойство - типа объекта. Значения типов объектов приведены ниже. Объявления констант, в том числе и типов, находятся в файле «C:\ProgramData\Lira Soft\Sapfir 1.3\Service\const_model.jsh».

Типы объектов

var TM_APP =1000; *// Приложение САПФИР (Application)*
var TM_DOC =1001; *// Документ САПФИР (Document)*

Типы параметрических объектов:

var TM_PROJECT = 3; *// Здание/проект (AutoProject)*
var TM_STOREY = 4; *// Этаж (AutoStorey)*
var TM_BLOCK = 5; *// Блок (AutoObjBlock)*
var TM_FEAPROJECT = 6; *// Конструкции*

var TM_WALL = 10; *// Стена (AutoModel)*
var TM_SLAB = 11; *// Перекрытие (AutoModel)*
var TM_COLUMN = 12; *// Колонна (AutoModel)*
var TM_BEAM = 13; *// Балка (AutoModel)*
var TM_ROOF = 15; *// Крыша (AutoModel)*
var TM_STAIRS = 16; *// Лестница (AutoModel)*
var TM_AXES = 17; *// Оси координационные (AutoModel или AutoAxes)**
var TM_LINE = 18; *// Линия построения (AutoModel)**
var TM_ZONE = 19; *// Зона/помещение (AutoModel или AutoZone)**
var TM_DIMENSION = 20; *// Размеры и обозначения (AutoModel или AutoObjDim)**
var TM_LOAD = 21; *// Нагрузка (AutoModel)*

var TM_HOLE = 30; *// Проем (AutoModel)*
var TM_WINDOW = 31; *// Окно (AutoModel)*
var TM_DOOR = 32; *// Дверь (AutoModel)*
var TM_SPHERE = 60; *// Сфера (AutoModel)*
var TM_PIVOTAL = 61; *// Поверхность вращения (AutoModel)*
var TM_PRISM = 62; *// Призма (AutoModel)*
var TM_PROFILE = 63; *// Кинематическая поверхность (профиль) (AutoModel)*
var TM_CONE = 64; *// Коническая поверхность (AutoModel)*
var TM_HIPPAR = 65; *// Гиперболический параболоид (AutoModel)*
var TM_LIGHT = 66; *// Источник света (AutoModel)*
var TM_POLY = 67; *// Заштрихованный полигон (AutoModel или AutoObjPolygon)**
var TM_TEXT = 68; *// Текст (AutoObjText)*
var TM_CYLINDER = 69; *// Цилиндрическая поверхность (AutoModel)*

var TM_FEASCHEMA = 80; *// Расчетная модель*
var TM_FEABAR = 81; *// Идеализированная модель стержня*
var TM_FEASHELL = 82; *// Идеализированная модель оболочки*

var TM_MULTICONT = 83; *// Объект параметрическое сечение (AutoMultiContour)*
var TM_OTHER = 128; *// Произвольный объект (AutoModel)*

Материалы

var TM_SET_MATERIALS = 250; *// Набор материалов*
var TM_MATERIAL = 251; *// Материал*
var TM_MULTI_MAT = 252; *// Многослойный материал (конструкция)*
var TM_LAYER_MAT = 253; *// Слой многослойного материала*
var TM_SET_LAYERS = 260; *// Набор слоев*
var TM_LAYER = 261; *// Слой*

var TM_MODVISIONSECTION = 34; *// Контейнер моделей вида «вертикальный разрез/сечение»*
var TM_MODVISIONPLAN = 35; *// Контейнер моделей вида «план»*

Графические 2D и 3D объекты

var TM3_LINE = 1; *// Отрезок (AutoLine)*
var TM3_ARC = 2; *// Дуга окружности/окружность (AutoLine)*
var TM3_BEZIER = 3; *// Кривая Безье/сплайн (AutoLine)*
var TM3_POLYLINE = 4; *// Полилиния (AutoPolyLine)*
var TM3_MESH3D = 5; *// Сетка 3D граней (AutoMesh)*
var TM3_MODEL3D = 6; *// 3D модель (AutoModel3D)*
var TM3_TEXT3D = 7; *// Текст (AutoText)*
var TM3_FACE = 8; *// Грань 3D сетки*
var TM3_DIMENSION = 11; *// Размеры и обозначения (AutoDim)*
var TM3_POLY = 12; *// Штриховка (AutoPolygon)*
var TM3_AXIS = 13; *// Координационная ось*
var TM3_FE_BAR = 14; *// Конечный элемент (КЭ) стержень*
var TM3_FE_TRIANGLE = 15; *// КЭ треугольник*
var TM3_FE_QUAD = 16; *// КЭ четырехугольник*
var TM3_FE_TETRAEDR = 17; *// КЭ тетраэдр*
var TM3_FE_PRYSM_3 = 18; *// КЭ трехгранная призма*
var TM3_FE_PRYSM_4 = 19; *// КЭ универсальный 8-ми узловой элемент*
var TM3_FE_SUPER = 20; *// КЭ суперэлемент*
var TM3_ELLIPSE = 21; *// Эллипс (дуга эллипса)*
var TM3_POINT = 22; *// 3D точка/маркер*

var TM_SET_DRAFTSHEETS = 270; *// Набор чертежных листов*
var TM_DRAFTSHEET = 271; *// Чертежный лист*
var TM_DRAFT = 100; *// Чертеж*

Сервисные объекты

var TM_CONT = 1002; *// Набор полилиний (AutoVectContour)*
var TM_SUCRE = 1003; *// Генератор геометрических моделей (AutoSurfCreator)*
var TM_FACTORY = 1004; *// Фабрика объектов (AutoFactory)*
var TM_FILEDLG = 1005; *// Диалог выбора файла (AutoFileDialog)*
var TM_MATR = 1006; *// Матрица AutoMatrix*
var TM_PARAMDLG = 1007; *// Диалог настройки параметров объектов (AutoParametersDlg)*
var TM_LOCUSER = 1008; *// Объект доступа к инструментальной панели (AutoLocatorUser)*
var TM_LIBDLG = 1009; *// Сервисный диалог для окон,дверей, сечений (AutoLibMdlDlg)*
var TM_SAPVIEW = 1010; *// Вид документа (AutoSapfirView)*
var TM_SOLVER = 1011; *// Геометрический вычислитель (AutoSolver)*
var TM_KMSECTION= 1012; *// Металлический сортамент (AutoKmSection)*

Приложение (Application)

Получить объект приложения САПФИР (версии 1.3):

var SapfirApp = new ActiveXObject("**Sapfir.Application.1.3**");

Получить тип объекта.

var type = SapfirApp.**TypeModel**; *Получить тип объекта (TM_APP)*

var strType = SapfirApp.**TypeExternalObj**; *Получить строку «SapfirApp»*

var version = SapfirApp.**AppVersion**; *Получить строку версии приложения «1.3»*

Получить документ (Document).

var SapfirDoc = SapfirApp.**NewDocument**(); *Получить новый документ*

var SapfirDoc = SapfirApp.**OpenLastDocument** (); *Прочитать файл который открывался в последнем сеансе работы САПФИР*

var SapfirDoc = SapfirApp.**OpenDocument** (filePath, bdlg); *Открыть файл с заданным именем*

var filePath - *полный путь к файлу;*
var bdlg - *показывать диалог выбора файлов: 0 - нет, 1 - да.*

var SapfirDoc = SapfirApp.**GetActiveDoc** (); *Получить текущий документ*

var SapfirDoc = SapfirApp.**SetActiveDoc** (filePath); *Получить заданный документ. Если такой документ еще не открыт, то выполняется чтение файла.*

var filePath - *полный путь к файлу;*

Вывести сообщение

SapfirApp.**SetStatusString**(sText); *Вывести статусную строку*

var sText – *строка текста;*

var retCode = SapfirApp.**AfxMessage**(sText, nType); *Вывести диалог сообщения*

var sText - *строка текста;*
var nType – *тип сообщения:* MB_OK, MB_OKCANCEL, MB_ABORTRETRYIGNORE, MB_YESNOCANCEL, MB_YESNO, MB_RETRYCANCEL

SapfirApp.**PutDisplayString** (nWnd, sText, IDobj); *Вывести строку в окно служебной информации*

var nWnd – *тип окна:*
OUT_PROC - *окно процесса*
OUT_ERROR - *окно ошибок*
OUT_ESTIMATE - *окно расчетов*

var sText – *строка текста;*
var IDobj – *идентификатор объекта модели (ID), 0-не используется;*

Управление индикатором прогресса операции

SapfirApp.**EnableProgressBar** (nIndex, nTotal); *Инициализировать индикатор прогресса*

var nIndex – *задает минимальное значение для прогресса;*
var nTotal – *задает максимальное значение для прогресса;*

SapfirApp.**EnableProgressBar** (nCurr); *Задать текущее значение прогресса*

var nCurr - *текущее значение прогресса;*

SapfirApp.**DisableProgressBar**(nCurr); *Закрыть индикатор прогресса*

Список ранее открываемых проектов

var count = SapfirApp.**GetCountRecentFile**(); *Получить количество файлов*

var nameFile = SapfirApp.**GetNameRecentFile**(index); *Получить путь к файлу*

var index – *индекс(номер) файла [0,...,count -1];*

Получить путь к служебной папке

var specPath = SapfirApp.**GetSpecPath** (typePath); *Получить путь к папке заданного типа*

var typePath – *тип служебной папки (см. Служебные папки САПФИР)*

Фабрика объектов САПФИР (AutoFactory)

var Factory = SapfirApp.**GetFactory** (); *Получить фабрику объектов приложения*

Вид документа САПФИР (AutoSapfirView)

var SapfirView = SapfirApp.**GetActiveSapfirView** (); *Получить текущий вид документа*

Диалог выбора файлов (AutoFileDialog)

var FileDialog = SapfirApp.**GetFileDialog** (); *Получить диалог выбора файлов*

Параметры стиля интерфейса

SapfirApp.**GetSystemColor** (type, ColorBuf); *Получить текущие цвета интерфейса*

var type = т*ип объекта интерфейса:*

COLOR_SCROLLBAR, COLOR_BACKGROUND,
COLOR_ACTIVECAPTION, COLOR_INACTIVECAPTION,
COLOR_MENU, COLOR_WINDOW, COLOR_WINDOWFRAME,
COLOR_MENUTEXT, COLOR_WINDOWTEXT,
COLOR_CAPTIONTEXT, COLOR_ACTIVEBORDER,
COLOR_INACTIVEBORDER, COLOR_APPWORKSPACE,
COLOR_HIGHLIGHT, COLOR_HIGHLIGHTTEXT, COLOR_BTNFACE,
COLOR_BTNSHADOW, COLOR_GRAYTEXT, COLOR_BTNTEXT,
COLOR_INACTIVECAPTIONTEXT, COLOR_BTNHIGHLIGHT.

var ColorBuf = *new Array(); - возвращаемый массив параметров*

ColorBuf[0] – *RGB светлой части*
ColorBuf[1] – *RGB темной части*
ColorBuf[2] – *тип растяжки цвета: 0 — вертикальная, 1- горизонтальная*

Геометрический вычислитель (AutoSolver)

var Solver = SapfirApp. **GetSolver** (); *Получить геометрический вычислитель*

Сортамент стальных профилей (AutoKmSection)

var KmSec = SapfirApp. **GetKmSection** (); *Получить сортамент металлических профилей*

Блокировка приложения

При использовании приложения САПФИР из других программ можно установить блокировку приложения. При попытке выхода из САПФИР будет выдаваться предупреждение что программа используется другим приложением с именем sNameApp, что исключает случайную потерю данных.

SapfirApp. **SetLockedApp** (sNameApp); *Установить блокировку приложения*

SapfirApp. **ReleaseLockedApp** (sNameApp); *Снять блокировку приложения*
var sNameApp - *имя приложения, блокирующее САПФИР*

Документ (Document)

Получить тип объекта.

var type = SapfirDoc.**TypeModel**; *Получить тип объекта (TM_DOC)*

var strType = SapfirDoc.**TypeExternalObj**; *Получить строку «SapfirDoc»*

Получить приложение САПФИР (Application).

var SapfirApp = SapfirDoc.**GetApplication()**; *Получить приложение САПФИР*

Проект (AutoProject)

var Project = SapfirDoc.**NewProject ()**; *Получить новый проект/здание*

var Project = SapfirDoc. **GetActiveProject ()**; *Получить активный проект*

var count = SapfirDoc. **CountProjects**; *Количество проектов в документе*

var Project = SapfirDoc. **GetProjectByIndex (**index**)**; *Получить проект по индексу*
var index – *индекс(номер) проекта [0,...,count -1]*

var Project = SapfirDoc. **GetProjectByID (**idProject**)**; *Получить проект по ID*
var idProject – *идентификатор проекта (ID)*

SapfirDoc. **DelProjectByID (**idProject**)**; *Удалить проект по ID*
var idProject – *идентификатор проекта (ID)*

SapfirDoc. **RemoveAllProjects ()**; *Удалить все проекты*

Этаж (AutoStorey)

var Storey = SapfirDoc.**GetActiveStorey ()**; *Получить активный этаж*

SapfirDoc. **SetActiveStorey (**idStorey**)**; *Установить активный этаж*
var idStorey – *идентификатор текущего этажа*

Модель (AutoModel)

var Model = SapfirDoc. **FindModelByID (**idModel**)**; *Найти модель*
var idModel – *идентификатор модели*

Материал (AutoMaterial)

var Material = SapfirDoc. **GetMaterialByGUID (**strGUID**)**; *Найти материал*
var strGUID – *строка с GUID материала (глобальный идентификатор)*

var Material = SapfirDoc. **GetMaterialByID (**idMat**)**; *Найти материал*
var idMat – *идентификатор материала*

Набор материалов (AutoSetMaterials)

var SetMaterials = SapfirDoc. **GetSetMaterials ()**; *Получить набор материалов*

Контейнер слоев (AutoSetLayers)

var SetLayers = SapfirDoc. **GetSetLayers ()**; *Получить контейнер слоев*

Сгенерировать новый GUID

var strGUID = SapfirDoc.**GuidGen**(); *Сгенерировать новый глобальный идентификатор GUID*

Вывести сообщение

var retCode = SapfirDoc.**AfxMessage**(sText, nType); *Вывести диалог сообщения*
var sText - *строка текста;*
var nType – *тип сообщения:* MB_OK, MB_OKCANCEL, MB_ABORTRETRYIGNORE, MB_YESNOCANCEL, MB_YESNO, MB_RETRYCANCEL

SapfirDoc.**PutDisplayString** (nWnd, sText, idObj); *Вывести строку в окно служебной информации*
var nWnd – *тип окна:* OUT_PROC *- окно процесса*
OUT_ERROR *- окно ошибок*
OUT_ESTIMATE *- окно расчетов*
var sText – *строка текста;*
var idObj – *идентификатор объекта модели (ID), 0-не используется;*

Текущий язык

var CurLan = SapfirDoc.**CurrentLanguage**; *Получить текущий язык документа*
SapfirDoc.**CurrentLanguage =** CurLan; *Установить текущий язык документа*
var = CurLan – код языка: LANG_ENGLISH, LANG_RUSSIAN, LANG_UKRAINIAN;

Название документа

var sName = SapfirDoc.**Title** ; *Получить название документа*
SapfirDoc. **Title =** sName; *Установить название документа*
var sName – *строка названия документа*

Вид документа САПФИР (AutoSapfirView)

var SapfirView = SapfirDoc. **GetActiveSapfirView** (); *Получить текущий вид документа*

UNDO команд САПФИР

SapfirDoc. **BeginUndo** (); *Начало undo при добавлении/изменении объектов*

SapfirDoc. **EndUndoParam** (code, name, idObjects); *Конец undo при добавлении/изменении объектов*

var code - *код команды:*
var name - *название команды UNDO*
var idObjects - *массив ID объектов*

var SapfirView = SapfirDoc. **BeginUndo** (code, name, idObjects); *Начало undo при удалении объектов*

var SapfirView = SapfirDoc. **EndUndo** (); *Конец undo при удалении объектов*

Диалог настройки именованных параметров (AutoParametersDlg)

var ParDlg = SapfirDoc. **GetParametersDlg ()**; *Получить диалог настройки параметров*

*Инструментальная панель (*AutoLocatorUser*)*

var LocatorUser = SapfirDoc. **GetLocatorUser ()**; Получить объект для доступа к инструментальной панели САПФИР

Блоки объектов

SapfirDoc. **BlockGroupSelModel ()**; *Выделенные объекты поместить в блок*
SapfirDoc. **BlockUnGroupSelModel ()**; *"Рассыпать" выделенные блоки на объекты*

Перерисовать модель

SapfirDoc. **DrawModel (**mode**)**; *Перерисовать модель*

var mode – *режим перерисовки/регенерации модели*

UPDATE_MODELS - *перерисовать объекты*
ZOOM_ALL - *показать все*
VIEW_PR_PARALLEL - *установить параллельную проекцию*
VIEW_PR_PERSPECTIVE - *установить перспективную проекцию*
VIEW_PR_UPDATE_SELECT - *перерисовать выделенные объекты*

SapfirDoc. **ObjectLookUp (**ObjID**)**; *Показать заданный объект на все окно*

var ObjID – *идентификатор объекта*

Показать свойства объекта

SapfirDoc. **ObjectViewProp (**ObjID**)**; *Показать свойства объекта в окне свойств*

var ObjID – *идентификатор объекта*

Регенерация объектов

SapfirDoc. **RegenAllModels ()**; *Регенерить модели всех объектов*

Именованные параметры

SapfirDoc. **SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*

var param = *new Array();* - *массив параметров объекта*

param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *установить значения параметров*

OBP_NAMES - установить названия параметров
OBP_COMMENTS - *установить примечания параметров*

SapfirDoc. **GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - Получить названия параметров
OBP_COMMENTS - *Получить примечания параметров*

SapfirDoc. **ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = SapfirDoc. **Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Чертежи

var DraftSheets = SapfirDoc. **GetDraftSet ()**; Получить набор чертежных листов (AutoDraftSet)

SapfirDoc. **ReadDraftSheet** (nameFile); *Прочитать заданный файл чертежных листов*
var nameFile – полный путь к файлу

SapfirDoc. **WriteDraftSheet** (nameFile, idSheets); *Записать заданные чертежные листы в файл*
var nameFile – *полный путь к файлу*
var idSheets = *new Array(); - массив идентификаторов (ID) листов чертежей*

SapfirDoc. **ShowDraftSheet** (IdDraftSheet, mode); *Перерисовать чертежный лист*
var IdDraftSheet - *идентификатор (ID) чертежного листа*
var mode – *режим : 0 - рисовать если вид чертежа активный, 1 - рисовать всегда.*

Выделенные объекты

var count = SapfirDoc. **CountSelectedModel**; *Получить количество выделенных объектов*

var Model = SapfirDoc. **GetSelectedModel**(Index); *Получить выделенный объект, приведенный к типу AutoModel*
var Index – *индекс выделенного объекта*

var Model = SapfirDoc. **GetSelectedModel2**(Index); *Получить выделенный объект без приведения к типу AutoModel*
var Index – *индекс выделенного объекта*

var count = SapfirDoc. **UnSelectAllObject()**; *Развыделить все объекты*

var count = SapfirDoc. **CreateNotesSelectObj()**; *Создать выносные линии для выделенных объектов в текущем виде документирования*

Получить полный путь файлу

var FIlePath = SapfirDoc. **fullPath**; *Получить полный путь файлу*

Здание (AutoProject)

var type = Project.**TypeModel**; *Получить тип объекта : (TM_PROJECT)*

var IdProject = Project.**ID**; *Получить значение идентификатора объекта (ID)*

Project.**SetPosition**(pos_x, pos_y, pos_z); *Установить позицию проекта в глобальной системе координат*

Этажи

var Storey = Project.**NewStorey**(nameStorey) ; *Создать новый этаж*
var nameStorey – *Название нового этажа*

var countSt = Project.**CountStorey**; *Получить количество этажей в проекте*

var Storey = Project.**GetStoreyByIndex**(indexStorey) ; *Получить этаж по индексу*
var indexStorey – *индекс этажа в проекте (здании)*

var Storey = Project.**GetStoreyByID**(IdStorey) ; *Получить этаж по идентификатору (ID)*
var IdStorey – *идентификатор этажа в проекте (здании)*

Project.**DelStoreyByIndex** (indexStorey) ; *Удалить этаж по индексу*
var indexStorey – *индекс этажа в проекте (здании)*

Project.**DelStoreyByID**(IdStorey) ; *Удалить этаж по идентификатору (ID)*
var IdStorey – *идентификатор этажа в проекте (здании)*

Именованные параметры

Project.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var param = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

Project.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

Project.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = Project.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Project.**Parameter** (nameParam) = newValue; *// Установить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*
var newValue - *новое значение параметра*

Этаж (AutoStorey)

var type = Storey.**TypeModel**; *Получить тип объекта : (TM_STOREY)*

var IdModel = Storey.**ID**; *Получить значение идентификатора объекта (ID)*

Объекты этажа

var countModel = Storey.**CountModel**; *Получить количество объектов в этаже*

var Model = Storey.**GetModelByIndex**(indexModel) ; *По индексу получить объект, приведенный к AutoModel*

var indexModel – *индекс объекта в этаже*

var Model = Storey.**GetModelByIndex2**(indexModel) ; *По индексу получить объект, без приведенния к AutoModel*

var indexModel – *индекс объекта в этаже*

var Model = Storey.**GetModeByID**(IdModel) ; *Получить объект по идентификатору (ID)*

var IdModel – *идентификатор объекта в этаже*

Storey.**DelModeByIndex** (indexModel) ; *Удалить объект по индексу*

var indexModel – *индекс в объекта в этаже*

Storey.**DelModeByID**(IdModel) ; *Удалить объект по идентификатору (ID)*

var IdModel – *идентификатор объекта в этаже*

Storey.**RemoveAll** () ; *Удалить все объекты в этаже*

var Model = Storey.**NewModel** (typeModel) ; *Добавить в этаж новый объект, заданный своим типом. (см. типы параметрических объектов) Возвращается объект без приведения к AutoModel*

Применить матрицу преобразования

Storey. **MultMatrix** (ValMatr) ; *Умножить все объекты этажа на матрицу преобразования*

var ValMatr = new Array(a11, a12, a13, a14,
a21, a22, a23, a24,
a31, a32, a33, a34,
a41, a42, a43, a44);

Подрезка стен

Storey. **CorrectJoint** () ; *Выполнить взаимную подрезку торцов стен этажа*

Именованные параметры

Storey.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*

var param = *new Array(); - массив параметров объекта*

param[i] – *строковое обозначение i-го параметра (мнемоника)*

param[i+1] – *значение i-го параметра*

param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

Storey.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*

var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *Получить значения параметров*

OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

Storey.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = Storey.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Storey.**Parameter** ("M_NAME") = newValue; *// Установить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*
var newValue - новое значение параметра

Параметрическая модель (AutoModel)

var type = Model.**TypeModel**; *Получить тип объекта - возвращается реальный тип объекта (см. Типы объектов)*

var IdModel = Model.**ID**; *Получить значение идентификатора объекта (ID)*

Получить приложение САПФИР (Application).

var SapfirApp = Model.**GetApplication()**; *Получить приложение САПФИР*

Получить этаж (AppStorey).

var Storey = Model.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Получить «родительский» объект (AutoModel).

var ParentModel = Model.**GetParentObj**(); *Получить объект (блок или этаж), в котором находится данный объект*

Именованные параметры

Через механизм именованных параметров ***универсального объекта AutoModel*** можно получить доступ к свойствам, которые поддерживает реальный объекет.

Model.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var param = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

Model.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

Model.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

Model. **RemoveMnParam** (mode, param); *Удалить параметры объекта*
var mode – 0 – все дополнительные параметры
1 – заданные в param в дополнительные параметры
2 - все дополнительные параметры между двумя заданными включительно
var param – new Array(); - массив мнемоник удаляемых параметров

var Value = Storey.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Model.**Parameter** (nameParam) = newValue; *// Установить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*
var newValue - новое значение параметра

Осевая линия объекта (AutoPolyLine)

var PLine = Model. **GetAxisLine** (); *Получить осевую линию у следующих объектов: TM_WALL, TM_SLAB, TM_BEAM, TM_ROOF, TM_HOLE, TM_LINE.*

Массив контуров объекта (AutoVectContour)

var Contour = Model.**GetContour** (); *Получить массив контуров у следующих объектов: TM_ZONE, TM_COLUMN, TM_BEAM, TM_PIVOTAL, TM_PRISM, TM_MULTICONT.*

Графические 2D и 3D объекты (AutoModel3D)

var Model = Model.**GetModel3D** (); *Получить 3D модель визуализации объекта.*

var Model = Model.**GetPlanModel** (); *Получить 2D модель плана объекта.*

Проемы, Окна, Двери (AutoModel)

var ModelHole = Model. **NewHole** (); *Добавить в объект модель проема: TM_HOLE , TM_WINDOW, TM_DOOR. Работает только для объектов TM_WALL, TM_SLAB и TM_ROOF*

var ModelHole = Model. **SetPosHole** (pos, IdSegm); *Задать позицию проема:*
var pos *- смещение проема от начала сегмента осевой линии стены (крыши)*
var IdSegm *- идентификатор (ID) сегмента осевой линии стены (крыши)*

Параметрическое сечение (AutoMultiContour)

var MultiContour = Model.**GetMultiContour** (); *Получить параметрический контур сечения колонн (TM_COLUMN) и балок (TM_BEAM)*

Металлический сортамент (AutoKmSection)

var KmSec = Model. **GetKmSection** (); *Получить генератор сортамента стальных сечений*

Выделение объекта

var mode = Model.**Select**; Определить, является ли объект выделенным
mode = 0 – объект не выделенный, 1 — объект выделенный

Model.**Select** = mode; Установить/сбросить выделение объекта
var mode = 0 – сбросить выделение объекта, 1 — выделить объект

Слой моделирования объекта

var layer = Model.**LayerID** ; Получить идентификатор слоя моделирования данного объекта

Model.**LayerID** = layer; Установить идентификатор слоя моделирования данного объекта

Материал объекта

var **Material** = Model. **GetMaterial**() ; Получить материал объекта *(AutoMaterial*)

var IdMat = Model. **MaterialID** ; Получить идентификатор материала объекта

Model. **MaterialID** = IdMat; Установить идентификатор материала объекта

Геометрический вычислитель (AutoSolver)

var **Solver** = Model. **GetSolver**() ; *Получить геометрический вычислитель*

Фабрика объектов (AutoFactory)

var Factory = Model. **GetFactory**() ; *Получить фабрику объектов*

Дочерние объекты

var countModel = Model.**CountChild** ; *Получить количество дочерних (вложенных) объектов в блоке (только дляTM_BLOCK)*

var Model = Model.**GetChildByIndex** (indexModel) ; *По индексу получить объект, приведенный к AutoModel*

var indexModel – *индекс объекта в блоке*

var Model = Model.**GetChildByIndex2** (indexModel) ; *По индексу получить объект, без приведенния к AutoModel*

var indexModel – *индекс объекта в блоке*

Матрица объекта

Матрица объекта задает преобразование модели из локальной системы координат объекта (СК) в СК этажа и представляется в виде массива из 16 элементов:

```
var valMatr = new Array(    a11, a12, a13, a14,
                            a21, a22, a23, a24,
                            a31, a32, a33, a34,
                            a41, a42, a43, a44 );
```

Model.**SetMatrix** (valMatr) ; *Установить матрицу преобразования*

Model.**GetMatrix** (valMatr) ; *Получить матрицу преобразования*

Model. **MultMatrix** (valMatr) ; *Добавить матрицу преобразования (умножить)*

Очистка объекта

Model. **CleanBaseModel** () ; *Функция удаляет данные из объекта (проемы, подрезки, модели визуализации, дополнительные параметры и др.)*

Обновить модель объекта

Model. **RegenModel** () ; пересоздать модель визуализации объекта

Позиция объекта

Model. **SetPosition** (pos_x, pos_y, pos_z) ; *Установить позицию объекта*

Model.**GetPosition** (pos) ; *Получить позицию объекта*
var pos = new Array(); - возвращается pos[0]- pos_x, pos[1]- pos_y, pos[2]- pos_z

Подрезка объекта

Model. **CorrectJoint** () ; *Переподрезать торцы стены (только для объектов TM_WALL)*

Цвет объекта

Model. **SetColor** (colorLine, colorFill) ; *Задать параметры цвета объекта*
var colorLine – цвет ребер (линий) (RGB),
var colorFill — цвет заливки граней (RGBA)

Поворот объекта

Model. **SetRotate** (rot_X, rot_Y, rot_Z) ; *Повернуть объект. Для объектов: TM_WALL, TM_SLAB,TM_ROOF, TM_ZONE, TM_COLUMN, TM_AXES, TM_STAIRS, TM_HOLE, TM_WINDOW, TM_DOOR - разрешен поворот только вокруг оси Z (* rot_X=0 и rot_Y = 0 *)*
var rot_X – *поворот вокруг оси X*
var rot_Y – *поворот вокруг оси Y*
var rot_Z – *поворот вокруг оси Z*

Симметрия объекта

Model. **Symmetry** (param) ; *Симметрировать объект. Для объектов: TM_WALL, TM_SLAB, TM_ROOF, TM_ZONE, TM_COLUMN, TM_AXES, TM_STAIRS, TM_HOLE, TM_WINDOW, TM_DOOR - разрешена симметрия только относительно вертикальной плоскости (* param[2] = *0)*
var param = new Array() - *параметры плоскости симметрии*
param[0], param[1], param[2] - *вектор нормали плоскости V(A,B,C)*
param[3] – *растояние от начала системы координат до плоскости (D)*
**Плоскость задается уравнением A*x+B*y+C*z+D=0;*

Блок (AutoObjBlock)

var type = Block.**TypeModel**; *Получить тип объекта (TM_BLOCK)*

var IdModel = Block.**ID**; *Получить значение идентификатора объекта (ID)*

Получить этаж (AppStorey).

var Storey = Block.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Именованные параметры

Block.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var param = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

Block.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта*
см. Описание именованных параметров
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

Block.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = Block.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Block.**Parameter** (nameParam) = newValue; *// Установить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*
var newValue - ново
е значение параметра

Объекты блока

var countModel = Block.**CountModel**; *Получить количество объектов в этаже*

var Model = Block.**GetModelByIndex**(indexModel) ; *По индексу получить объект, приведенный к AutoModel*
var indexModel – *индекс объекта в этаже*

var Model = Block.**GetModelByIndex2**(indexModel) ; *По индексу получить объект, без приведенния к AutoModel*
var indexModel – *индекс объекта в этаже*

var Model = Block.**GetModeByID**(IdModel) ; *Получить объект по идентификатору (ID)*
var IdModel – *идентификатор объекта в этаже*

Block.**DelModeByIndex** (indexModel) ; *Удалить объект по индексу*
var indexModel – *индекс в объекта в этаже*

Block.**DelModeByID**(IdModel) ; *Удалить объект по идентификатору (ID)*
var IdModel – *идентификатор объекта в этаже*

var Model = Block.**NewModel** (typeModel) ; *Добавить в этаж новый объект, заданный своим типом. Возвращается объект без приведения к AutoModel*

Block. **MultMatrix** (ValMatr) ; *Добавить матрицу преобразования (умножить) для всех объектов блока*
var ValMatr = new Array(a11, a12, a13, a14,
a21, a22, a23, a24,
a31, a32, a33, a34,
a41, a42, a43, a44);

Обозначение (AutoObjDim)

var type = ObjDim.**TypeModel**; *Получить тип объекта (TM_DIMENSION)*

var IdModel = ObjDim.**ID**; *Получить значение идентификатора объекта (ID)*

Получить этаж (AppStorey).

var Storey = ObjDim.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Цвет объекта

ObjDim. **SetColor** (colorLine, colorText) ; *Задать параметры цвета*
var colorLine – цвет линий (RGB),
var colorText – цвет текста (RGB)

Задать параметры размера

ObjDim. **SetDimParam** (type, dir, name, points) ; *Задать параметры размера*
var type - *тип размера/обозначения :*
DIM_ELEV - высотная отметка
DIM_LINEAR - линейный размер
DIM_CHAIN - размерная цепочка
DIM_RADIAL - радиальный размер
DIM_ANGULAR - угловой размер
DIM_NOTE - текстовая выноска
DIM_DIAMETR - диаметр
DIM_AXIS - координационная ось
DIM_POINT - точечный маркер
DIM_ARC - длина дуги
DIM_MARKER_CIRCLE - круглый маркер

var dir - *направление измерения размера:* *DIM_DIR_X, DIM_DIR_Y, DIM_DIR_XY DIM_DIR_Z*

var name – *название размера (символьное обозначение)*
var points = new Array(x1,y1,z1, x2,y2,z2...); - *массив точек размера*

Именованные параметры

ObjDim.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var param = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

ObjDim.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

ObjDim.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = ObjDim.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

ObjDim.**Parameter** (nameParam) = newValue; *// Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue - новое значение параметра

Задать точки размера

ObjDim. **SetPoints** (arrayParam, mode); *Задать точки размера*
var points = new Array(x1,y1,z1, x2,y2,z2...); - *массив точек размера*

Получить графический 3D-размер (AutoDim)

var SpDim = ObjDim. **GetSpDim** (); *Получить графический размер.*

Заштрихованная область (AutoObjPolygon)

var type = ObjPolygon.**TypeModel**; *Получить тип объекта (TM_POLY)*

var IdModel = ObjPolygon.**ID**; *Получить значение идентификатора объекта (ID)*

Получить этаж (AppStorey).

var Storey = ObjPolygon.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Цвет объекта

ObjPolygon. **SetColor** (colorLine, colorFill) ; *Задать параметры цвета*
var colorLine – *цвет линий (RGB),*
var colorFill – *цвет заливки контура (RGB)*

Именованные параметры

ObjPolygon.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var arrayParam = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

ObjPolygon.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *Получить значения параметров*

OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

ObjPolygon.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = ObjPolygon.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

ObjPolygon.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue - новое значение параметра

Задать позицию объекта

ObjDim. **SetPosition** (pos_x, pos_y, pos_z); *Задать позицию объекта*

Получить графический штриховка (AutoPolygon)

var SpPolygon = ObjPolygon. **GetSpPolygon** (); *Получить графический объект — штриховка (полигон).*

Получить набор контуров

var Contour = ObjPolygon. **GetContour** (); *Получить объект - набор контуров*

Текст (AutoObjText)

var type = ObjText.**TypeModel**; *Получить тип объекта (TM_ TEXT)*

var IdModel = ObjText.**ID**; *Получить значение идентификатора объекта (ID)*

Получить этаж (AppStorey).

var Storey = ObjText.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Цвет объекта

ObjText. **SetColor** (colorText) ; *Задать цвет*
var colorText – цвет текста (RGB),

Именованные параметры

ObjText.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var arrayParam = *new Array(); - массив параметров объекта*
param[i] – *строковое обозначение i-го параметра (мнемоника)*
param[i+1] – *значение i-го параметра*
param[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

ObjText.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*
var param = *new Array(); - возвращаемый массив параметров объекта*

см. Описание именованных параметров
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

ObjText.**ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = ObjText.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

ObjText.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue - новое значение параметра

Задать позицию текста

ObjText. **SetPosition** (pos_x, pos_y, pos_z); *Задать позицию объекта*

Получить графический текст (AutoText)

var SpText = ObjText. **GetSpText** (); *Получить графический объект — текст*

Координационные оси (AutoAxes)

var type = Axes.**TypeModel**; *Получить тип объекта (TM_AXES)*

var IdModel = Axes.**ID**; *Получить значение идентификатора объекта (ID)*

Умножить на матрицу

Axes.**MultMatrix**(valMatr); *Умножить текст на матрицу преобразования*
var valMatr = new Array(a11, a12, a13, a14,
a21, a22, a23, a24,
a31, a32, a33, a34,
a41, a42, a43, a44);

Именованные параметры

var Value = Axes.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Axes.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue - новое значение параметра

Выделение объекта

var select = Axes.**Select** ; *Получить признак выделения объекта*

Axes.**Select =** select**;** *Установить признак выделения объекта*
var select — 1 - *установить выделение*
0 - сбросить выделение

Помещение (AutoZone)

var type = ObjZone.**TypeModel**; *Получить тип объекта (TM_ZONE)*

var IdModel = ObjZone.**ID**; *Получить значение идентификатора объекта (ID)*

Получить этаж (AppStorey).

var Storey = ObjZone.**GetParentStorey**(); *Получить этаж, в котором находится данный объект, независимо от вхождения в блоки*

Именованные параметры

ObjZone.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*

var arrayParam = *new Array(); - массив параметров объекта*

param[i] – *строковое обозначение i-го параметра (мнемоника)*

param[i+1] – *значение i-го параметра*

*p*aram[i+2] – *тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *установить значения параметров*

OBP_NAMES - *установить названия параметров*

OBP_COMMENTS - *установить примечания параметров*

ObjZone.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*

var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *Получить значения параметров*

OBP_NAMES - *Получить названия параметров*

OBP_COMMENTS - *Получить примечания параметров*

var Value = ObjZone.**Parameter** (nameParam); *Получить значение параметра объекта*

var nameParam - *строковое обозначение параметра (мнемоника)*

ObjZone.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*

var nameParam - *строковое обозначение размера*

var newValue - *новое значение параметра*

Умножить на матрицу

ObjZone.**MultMatrix**(valMatr); *Умножить зону на матрицу преобразования*

```
var valMatr = new Array(    a11, a12, a13, a14,
                            a21, a22, a23, a24,
                            a31, a32, a33, a34,
                            a41, a42, a43, a44 );
```

Выделение объекта

var select = ObjZone.**Select** ; *Получить признак выделения объекта*

ObjZone.**Select =** select**;** *Установить признак выделения объекта*

var select — 1 - *установить выделение*

0 - сбросить выделение

Симметрия объекта

ObjZone. **Symmetry** (param) ; *Симметрировать объект. Симметрия разрешена только*

относительно вертикальной плоскости (param[2] = *0*)

var param = new Array() - *параметры плоскости симметрии*

param[0], param[1], param[2] - *вектор нормали плоскости V(A,B,C)*

param[3] – *растояние от начала системы координат до плоскости (D)*

**Плоскость задается уравнением A*x+B*y+C*z+D=0;*

Задать позицию марки зоны

ObjZone. **SetMarkPoint** (pos_x, pos_y, pos_z); *Задать позицию марки зоны*

Задать позицию названия зоны

ObjZone. **SetNamePoint** (pos_x, pos_y, pos_z); *Задать позицию для названия зоны*

Графическая модель (AutoModel3D)

var type = Model3D.**TypeModel**; *Получить тип объекта (TM3_MODEL3D)*

var IdModel = Model3D.**ID**; *Получить значение идентификатора объекта (ID)*

Объекты Model3D

var countModel = Model3D.**CountModel**; *Получить количество объектов в Model3D*

var Mod3D = Model3D.**GetModelByIndex**(indexModel) ; *По индексу получить объект, приведенный к AutoModel3D*

var indexModel – *индекс объекта в этаже*

var Mod3D = Model3D.**GetModelByIndex2**(indexModel) ; *По индексу получить объект, без приведения к AutoModel3D*

var indexModel – *индекс объекта в этаже*

Model3D.**DelModeByIndex** (indexModel) ; *Удалить объект по индексу*

var indexModel – *индекс в объекта в этаже*

var Mod3D = Model3D.**NewModel** (typeModel) ; *Добавить в Model3D новый объект, заданный своим типом (см. типы графических 2D и 3D объектов). Возвращается объект без приведения кAutoModel3D*

var count = Model3D. **AddModel** (Mod3D) ; *Добавить объект Mod3D в Model3D.*

**Фактически добавляется копия объекта.*

var Mod3D – *графический объект 2D/3D (IDispatch*), созданный ранее, например, при помощи фабрики объектов.*

Model3D.**RemoveAll** () ; *Удалить все объекты в Model3D*

Model3D.**MultMatr**(valMatr); *Умножить все объекты в Model3D на матрицу преобразования*

```
var valMatr = new Array(    a11, a12, a13, a14,
                            a21, a22, a23, a24,
                            a31, a32, a33, a34,
                            a41, a42, a43, a44 );
```

Model3D.**SetMatr**(ValMatr); *Установить матрицу в Model3D*

```
var ValMatr = new Array(    a11, a12, a13, a14,
                            a21, a22, a23, a24,
```

a31, a32, a33, a34,
a41, a42, a43, a44);

Model3D. **Symmetry** (param) ; *Симметрировать объект* Model3D.
var param = new Array() - *параметры плоскости симметрии*
param[0], param[1], param[2] - *вектор нормали плоскости V(A,B,C)*
param[3] – *растояние от начала системы координат до плоскости (D)*
**Плоскость задается уравнением A*x+B*y+C*z+D=0;*

Материал объекта

var MatId = Model3D. **MatId** ; *Получить идентификатор материала объекта*

Model3D. **MatId =** MatId**;** *Установить идентификатор материала объекта*

Признак блока

var block = Model3D. **Block** ; *Получить признак блока*

Model3D. **Block =** block *Установить признак блока*
var block - 1 объект обозначен как блок, 0 - нет

Признак указываемости

var pickable = Model3D. **Pickable** ; *Получить признак указываемости объекта*

Model3D. **Pickable =** pickable *Установить признак указываемости объекта*
var pickable - *1 объект указывается, 0 - нет*

Признак валидности модели

var Validate = Model3D. **ValidateModel** ; *Получить признак валидности модели*

Model3D. **ValidateModel =** Validate *Установить признак валидности объекта*
var Validate - 1 объект валидный, 0 - нет

Фабрика объектов САПФИР (AutoFactory)

var Factory = Model3D.**GetFactory** (); *Получить фабрику объектов приложения*

Геометрический вычислитель (AutoSolver)

var Solver = Model3D. **GetSolver** (); *Получить геометрический вычислитель*

Генератор геометрических моделей (AutoSurfCreator)

var SurfCreator = Model3D. **GetSurfCreator** (); *Получить генератор геометрических моделей*

Объект матрица (AutoMatrix)

var Matrix = Model3D. **CreateMatrix** (); *Получить объект «Матрица»*

Охватывающий габарит объекта

Model3D. **GetBox** (valBox); *Получить охватывающий габарит объекта*
var valBox = *new Array(min.x, min.y, min.z, max.x, max.y, max.z) – возвращаемое*

значение габаритного охватывающего бокса.

Рассчитать нормали граней

Model3D. **CalculateNormals** (0); *Выполняется расчет векторов нормалей граней вложенных объектов AutoMesh,*

Именованные параметры

Model3D.**SetMnParam** (arrayParam, mode); *Занести в объект массив параметров, в том числе и дополнительные*

var arrayParam = *new Array();* *- массив параметров объекта*
param[i] *– строковое обозначение i-го параметра (мнемоника)*
param[i+1] *– значение i-го параметра*
param[i+2] *– тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

Model3D.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*

var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

Model3D. **ClearMnParam** (0); *Удалить все дополнительные параметры объекта*

var Value = Model3D.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

Model3D.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue *- новое значение параметра*

3D сетка граней (AutoMesh)

var type = Mesh.**TypeModel**; *Получить тип объекта (TM3_ MESH3D)*

var IdModel = Mesh.**ID**; *Получить значение идентификатора объекта (ID)*

Точки сетки

Mesh.**AddPoints**(Points); *Добавить массив точек*
var Point = new Array(x1,y1,z1, x2,y2,z2, ….); *- массив новых точек*

Mesh.**GetPoints**(Points); *Получить массив точек*
var Points = new Array(x1,y1,z1, x2,y2,z2, ….); *- возвращаемый массив точек сетки*

var count = Mesh.**CountPoints**; *Получить количество точек в сетке*

Mesh.**GetPoint**(index, point); *Получить точку по индексу*
var index *– индекс точки*
var point = new Array(x,y,z); *- координаты точки сетки*

Mesh.**SetPoint**(index, point); *Установить точку по индексу*
var index – *индекс точки*
var point = *new Array(x,y,z); - координаты точки сетки*

Mesh.**SetPoint**(index, point); *Установить точку по индексу*

Добавление граней

Mesh.**AddFaceElement** (type, index, atrib); *Добавить грань заданную индексами вершин*
var type = TM3_FACE *- добавляем грань сетки*
var index = new Array(ind1, ind2, ind3, ...indN) *- индексы точек грани (не менее 3-х).*
var atrib *- параметры грани :* BILATERAL *- двусторонняя грань*
HARD_NORMAL *- грань не сглаживается*

Mesh.**AddFace** (points); *Добавить грань заданную координатами точек*
var points = new Array(x1, y1, z1, x1, y2, z2,..) *- массив точек грани (не менее 3-х).*

Применить матрицу преобразования

Mesh.**MultMatr**(valMatr); *Умножить все точки сетки на матрицу преобразования*
var valMatr = new Array(a11, a12, a13, a14,
a21, a22, a23, a24,
a31, a32, a33, a34,
a41, a42, a43, a44);

Охватывающий габарит

Mesh. **GetBox** (valBox); *Получить охватывающий габарит сетки*
var valBox = *new Array(min.x, min.y, min.z, max.x, max.y, max.z) – возвращаемое значение габаритного охватывающего бокса.*

Рассчитать нормали граней

Mesh. **CalculateNormals** (0); *Выполняется расчет векторов нормалей граней сетки*

Материал

var MatId = Mesh.**MatId** ; *Получить идентификатор материала*

Mesh.**MatId =** MatId**;** *Установить идентификатор материала*

Цвет

Mesh. **SetColor** (colorLine, colorFill) ; *Задать параметры цвета*
var colorLine – *цвет ребер (RGB),*
var colorFill – *цвет заливки граней (RGB)*

Признак указываемости

var pickable = Mesh. **Pickable** ; *Получить признак указываемости объекта*

Mesh. **Pickable =** pickable *Установить признак указываемости объекта*
var pickable *- 1 объект указывается, 0 - нет*

Графическая чертежная область (AutoPolygon)

var type = Polygon.**TypeModel**; *Получить тип объекта (TM3_ POLY)*

var IdModel = Polygon.**ID**; *Получить значение идентификатора объекта (ID)*

Цвет

Polygon. **SetColor** (colorLine, colorFill) ; *Задать параметры цвета*
var colorLine – *цвет линий штриховки (RGB),*
var colorFill – *цвет заливки контура (RGB)*

var color = Polygon. **ColorLine** ; *Получить цвет линий штриховки*

Polygon. **ColorLine =** color *Установить цвет линий штриховки*
var color – *цвет (RGB),*

var color = Polygon. **ColorOutLine** ; *Получить цвет линии контура*

Polygon. **ColorOutLine =** color *Установить цвет линии контура*
var color – *цвет (RGB),*

var color = Polygon. **ColorFill** ; *Получить цвет заливки контура*

Polygon. **ColorFill =** color *Установить цвет заливки контура*
var color – *цвет (RGBA),*

var IDcolor = Polygon. **ColorFillID** ; *Получить ID цвета заполнения фона*

Polygon. **ColorFillID =** IDcolor *Установить ID цвета заполнения фона*
var color – *цвет (RGBA),*

Типы линий

var type = Polygon. **TypeLine** ; *Получить тип линии штриховки*

Polygon. **TypeLine =** type *Установить тип линии штриховки*
var type - 0 – сплошная*, 1- пунктирная, 2 — штриховая, 3- штрих-пунктирная*

var type = Polygon. **TypeOutLine** ; *Получить тип линии контура*

Polygon. **TypeOutLine =** type *Установить тип линии контура*
var type - 0 – сплошная*, 1- пунктирная, 2 — штриховая, 3- штрих-пунктирная*

Толщины линий

var type = Polygon. **WidthLine** ; *Получить толщину линии штриховки*

Polygon. **WidthLine =** type *Установить толщину линии штриховки*
var type - 0 – *тонкая, 1- основная, 2 — утолщенная, 3-особая*

var type = Polygon. **WidthOutLine** ; *Получить толщину линии контура*

Polygon. **WidthOutLine =** type *Установить толщину линии контура*
var type - 0 – *тонкая, 1- основная, 2 — утолщенная, 3-особая*

Шаг штриховки

var value = Polygon. **Spacing** ; *Получить шаг штриховки*

Polygon. **Spacing =** value; *Установить шаг штриховки*
var value - *шаг штриховки в [м]*

Векторная штриховка

var mode = Polygon. **Hatching** ; *Получить тип векторной штриховки*

Polygon. **Hatching =** mode; *Установить тип векторной штриховки*
var mode - *векторная штриховка: 1 — есть: 0 - нет*

Угол штриховки

var angle = Polygon. **AngleHatching** ; *Получить угол векторной штриховки*

Polygon. **AngleHatching =** angle; *Установить угол векторной штриховки*
var angle - *угол векторной штриховки в [градусах]*

Признак рисования линии контура

var mode = Polygon. **OutLine** ; *Получить признак рисования линии контура*

Polygon. **OutLine =** mode *Установить признак рисования линии контура*
var mode - *1 - линию контура рисовать; 0 - нет*

Признак заполнения фона

var mode = Polygon. **FillBckg** ; *Получить признак заполнения фона*

Polygon. **FillBckg =** mode *Установить признак заполнения фона*
var mode - *1 - фон заполнять цветом/текстурой ; 0 - нет*

Признак указываемости

var pickable = Polygon. **Pickable** ; *Получить признак указываемости объекта*

Polygon. **Pickable =** pickable *Установить признак указываемости объекта*
var pickable - *1 объект указывается, 0 - нет*

Массив контуров объекта (AutoVectContour)

var Contour = Polygon.**GetContour** (); *Получить массив контуров:*

Графическая полилиния (AutoPolyLine)

var type = PolyLine.**TypeModel**; *Получить тип объекта (TM3_POLYLINE)*

var IdModel = PolyLine.**ID**; *Получить значение идентификатора объекта (ID)*

Тип полилинии

var type = PolyLine. **Type** ; *Получить тип линии*

PolyLine. **Type =** type *Установить тип полилинии*
var type - 0 – сплошная*, 1- пунктирная, 2 — штриховая, 3- штрих-пунктирная*

Толщина полилинии

var type = PolyLine. **Width** ; *Получить толщину полилинии*

PolyLine. **Width =** type *Установить толщину полилинии*
var type - 0 – *тонкая, 1- основная, 2 — утолщенная, 3-особая*

Цвет полилинии

var color = PolyLine. **Color** ; *Получить цвет полилинии*

PolyLine. **Color =** color *Установить цвет полилинии*
var color – *цвет (RGB),*

*Сегменты полилинии (**AutoLine**)*

var count = PolyLine. **CountLine** ; *Получить количество сегментов полилинии*

var Line = PolyLine. **GetLine**(index) ; *Получить сегмент полилинии (**AutoLine**) по индексу*
var index – *индекс сегмента*

var count = PolyLine. **AddModel**(PL) ; *Добавить сегмент в полилинию (добавляется копия)*
var PL – *существующая линия типов: TM3_LINE, TM3_ARC, TM3_BEZIER или TM3_ELLIPSE, созданная, например, при помощи фабрики моделей*
var count - *количество сегментов полилинии*

var Line = PolyLine. **AddLine**(type, param) ; *Добавить сегмент заданного типа*
var type – *тип сегмента и параметров:*
var param = new Array(); - *массив параметров сегмента, заданный в соответствии с типом сегмента:*

- var type = TM3_LINE: - отрезок
  var param = new Array(x1,y1,z1, x2,y2,z2); - *две точки отрезка*

- var type = TM3_ARC | TMA_T1T2T3– *дуга окружности по 3-м точкам;*
  var param = new Array(x1,y1,z1, x2,y2,z2, x3,y3,z3);
  где: x1,y1,z1, x2,y2,z2, x3,y3,z3 - *Три последовательные точки дуги*

- var type = TM3_ARC | TMA_C_RAD_U1U2– *дуга: центр, радиус и два угла*
  var param = new Array(Cx, Cy ,Cz, R, U1,U2);
  где: Cx, Cy ,Cz - *Координаты центра окружности,*
  R - *радиус окружности,*
  U1,U2 - *углы начала и конца дуги*

- var type = TM3_ARC | TMA_C_RAD_OR– *окружность: центр, радиус направление*
  var param = new Array(Cx, Cy ,Cz, R, Dir); -
  где: Cx, Cy ,Cz - *Координаты центра окружности,*
  R - *радиус окружности,*
  Dir - *0- ориентация против часовой стрелки, 1 — по часовой*

- var type = TM3_ARC | TMA_C_RAD_OR_T1T2– *дуга: центр, радиус, направление, точки начала и конца дуги*
  var param = new Array(Cx, Cy ,Cz, R, Dir, x1,y1,z1, x2,y2,z2); -
  где: Cx, Cy ,Cz - *Координаты центра окружности,*
  R - *радиус окружности,*
  Dir - *0- ориентация против часовой стрелки, 1 — по часовой*
  x1,y1,z1, x2,y2,z2 - *точки начала и конца дуги*
- var type = TM3_ELLIPSE | TMA_C_SM_SP: - *эллипс: центр, размер оси, отношение осей*
  var param = new Array(Cx, Cy ,Cz, R, S); -
  где: Cx, Cy ,Cz - *Координаты центра эллипса,*
  R - *размер оси*
  S - *отношение размера второй оси к заданной (R)*

- var type = TM3_ELLIPSE | TMA_C_SM_SP_D_U1U2: - *дуга эллипса: центр, размер оси, отношение осей, направление дуги, два угла*
  var param = new Array(Cx, Cy ,Cz, R, S, Dir, U1, U2);
  где: Cx, Cy ,Cz - *Координаты центра эллипса,*
  R- *размер оси*
  S – *отношение размера второй оси к заданной (R)*
  Dir – *0- ориентация против часовой стрелки, 1 — по часовой*
  U1,U2 – *углы начала и конца дуги*

- var type = TM3_ELLIPSE | TMA_C_SM_SP_D_U1U2_N: - *дуга эллипса: центр, размер оси, отношение осей, направление дуги, два угла, вектор нормали плоскости эллипса*
  var param = new Array(Cx, Cy ,Cz, R, S, Dir, U1, U2, Nx, Ny, Nz);
  где: Cx, Cy ,Cz - *Координаты центра эллипса,*
  R - *размер оси*
  S - *отношение размера второй оси к заданной (R)*
  Dir - *0 - ориентация против часовой стрелки, 1 — по часовой*
  U1,U2 - *углы начала и конца дуги*
  Nx, Ny, Nz - *вектор нормали плоскости эллипса*

- var type = TM3_BEZIER | TMA_BEZIER– *кривая безье: задана двумя опорными точками и двумя контрольными*
  var param = new Array(P1x, P1y ,P1z, Pc1x, Pc1y ,Pc1z,
  Pc2x, Pc2y ,Pc2z, P2x, P2y ,P2z);
  где: P1 и P2 - *Координаты опорных точек*
  Pc1 и Pc2 - *Координаты контрольных точек*

- var type = TM3_BEZIER | TMA_SPLINE– *кривая кубический сплайн: задана массивом опорных точек.*
  var param = new Array(P1x, P1y ,P1z, P2x, P2y ,P2z …..PNx, PNy ,PNz);
  где: P1 и P2 …PN — *Координаты опорных точек (не менее 3-х)*

Установить точки полилинии

PolyLine. **SetPoints**(points) ; *Задать массив точек полилинии. Формируется полилиния из отрезков.*
var points = new Array(x1, y1, z1, ... xn, yn, zn); - *заданный массив точек*

Добавить точки к полилинии

PolyLine. **AddPoints**(points) ; *Добавить массив точек полилинии. Добавляется новые отрезки*
var points = new Array(x1, y1, z1, … xn, yn, zn); - *заданный массив точек*

Получить точки полилинии

var count = PolyLine. **GetPoints**(points) ; *Получить массив точек полилинии полученных путем аппроксимации кривых линий.*
var points = new Array(x1, y1, z1, ... xn, yn, zn); - *возвращаемый массив точек*
var count – полученное количество точек

Эквидистанта

PolyLine. **Equidistant**(dist) ; *Получить эквидистантную полилинию*
var dist – *параметр эквидистанты (величина смещения)*

Копирование полилинии

PolyLine. **Copy**(PL) ; *Скопировать полилинию PL в PolyLine*
var PL – *полилиния источник*

Инверсия

PolyLine. **Invert**() ; *Выполнить инверсию полилинии: изменить на противоположный порядок следования сегментов в полилинии и направление каждого сегмента*

Нормаль к полилинии

PolyLine. **GetNormal**(vect) ; *Получить вектор нормали контура полилинии*
var vect = new Array(vx, vy, vz) - *возвращаемый вектор нормали к полилинии*

Касательная к полилинии

PolyLine. **GetTangentVectByPnt**(pnt, vect) ; *Получить вектор касательной к полилинии в заданной точке*
var pnt = new Array(x, y, z) - *координаты точки на полилинии*
var vect = new Array(vx, vy, vz) - *возвращаемый вектор касательной к полилинии*

Габарит полилинии

PolyLine. **GetBox** (valBox); *Получить охватывающий габарит полилинии*
var valBox = *new Array(min.x, min.y, min.z, max.x, max.y, max.z) – возвращаемое значение габаритного охватывающего бокса.*

Центр гравитации полилинии

PolyLine. **GetCenterOfGravity**(pnt) ; *Получить координаты центра гравитации полилинии*
var pnt = new Array(vx, vy, vz) - *возвращаемые координаты центра гравитации*

Применить матрицу преобразования

PolyLine. **MultMatrix** (valMatr) ; *Умножить все сегменты на матрицу преобразования*

```
var valMatr = new Array(    a11, a12, a13, a14,
                            a21, a22, a23, a24,
                            a31, a32, a33, a34,
                            a41, a42, a43, a44 );
```

Признак указываемости

var pickable = PolyLine. **Pickable** ; *Получить признак указываемости полилинии*

PolyLine. **Pickable =** pickable *Установить признак указываемости полилинии*
var pickable - *1 объект указывается, 0 - нет*

Признак замыкания

var closed = PolyLine. **Closed** ; *Получить признак замыкания полилинии*

PolyLine. **Closed =** closed *Установить признак замыкания полилинии*
var closed - *1 полилиния замкнутая, 0 - нет*

Признак замыкания

var hole = PolyLine. **HoleCont** ; *Получить признак отверстия (внутреннего контура)*

PolyLine. **HoleCont =** hole *Установить признак отверстия (внутреннего контура)*
var hole - *1 отверстие, 0 - нет*

Графическая линия (AutoLine)

var type = Line.**TypeModel**; *Получить тип реального объекта :*
TM3_LINE - отрезок, TM3_ARC — дуга окружности,
TM3_ELLIPSE — дуга эллипса, TM3_BEZIER - *кривая Безье/сплайн*
var IdModel = Line.**ID**; *Получить значение идентификатора объекта (ID)*
Line.**ID=**IdModel; *Установить значение идентификатора объекта (ID)*

Тип линии

var type = Line. **Type** ; *Получить тип линии*

Line. **Type =** type *Установить тип линии*
var type - 0 – сплошная*, 1- пунктирная, 2 — штриховая, 3- штрих-пунктирная*

Толщина линии

var type = Line. **Width** ; *Получить толщину линии*

Line. **Width =** type *Установить толщину линии*
var type - 0 – *тонкая, 1- основная, 2 — утолщенная, 3-особая*

Цвет линии

var color = Line. **Color** ; *Получить цвет линии*

Line. **Color =** color *Установить цвет линии*
var color – *цвет (RGB),*

Установить точки линии

Line. **SetPoints**(Points) ; *Задать точки для отрезка (TM3_LINE)*
var Points = new Array(x1, y1, z1, x2, y2, z2); - *две точки отрезка*

Line. **SetPoints2D**(x1, y1, x2, y2) ; *Задать точки для отрезка (TM3_LINE)*
x1, y1, x2, y2 - *две точки отрезка (z=0);*

Line. **SetPoints3D**(x1, y1, z1, x2, y2, z2) ; *Задать точки для отрезка (TM3_LINE)*
x1, y1, z1, x2, y2, z2 - *две точки отрезка*

Получить точки линии

var count = Line. **GetPoints**(points) ; *Получить массив точек линии полученных путем аппроксимации кривых линий.*
var points = new Array(x1, y1, z1, ... xn, yn, zn); - *возвращаемый массив точек*
var count – полученное количество точек

Копирование линии

Line. **Copy**(Ln) ; *Скопировать линию Ln в Line*
var Ln – *линия источник*

Получить параметры линии

var type = Line. **GetControlPar**(param) ; *Получить набор управляющих параметров линии*

var type -*тип линии*
var param = new Array(); - *полученный массив параметров линии, в соответствии с типом линии:*

- type = TM3_LINE - *отрезок*
  param[0],param[1],param[2] - *первая точка отрезка*
  param[3],param[4],param[5] - *вторая точка отрезка*

- type = TM3_ARC - *окружность или дуга окружности*
  param[0],param[1],param[2] - *центр окружности*
  param[3] - *радиус окружности*
  param[4] - *направление дуги: 0 – против часовой стрелки; 1 – по часовой стрелке*

  Если параметров больше 5, то это дуга окружности
  param[5],param[6],param[7] - *точка начала дуги*
  param[8],param[9],param[10] - *точка конца дуги*
  param[11],param[12] - *углы начала и конца дуги*

- type = TM3_BEZIER - *кривая безье*
  param[0],param[1],param[2] - *первая управляющая точка*
  param[3],param[4],param[5] - *первая контрольная точка*
  param[6],param[7],param[8] - *вторая контрольная точка*
  param[9],param[10],param[11] - *вторая управляющая точка*

- type = TM3_ELLIPSE - *эллипс или дуга эллипса*
  param[0],param[1],param[2] - *центр эллипса*
  param[3] - *размер оси эллипса*
  param[4] - *отношение размера второй оси к первой*

  Если параметров больше 5, то это дуга эллипса
  param[5] - *направление дуги: 0 – против часовой стрелки; 1 – по часовой стрелке*
  param[6],param[7],param[8] - *точка начала дуги*
  param[9],param[10],param[11] - *точка конца дуги*
  param[12],param[13] - *углы начала и конца дуги*

Установить параметры линии

Line. **SetControlPar**(type, param) ; *Установить набор управляющих параметров линии*
var type – тип линии и параметров
var param = new Array(); - *массив параметров линии, заданный в соответствии с типом линии:*

- var type = TM3_LINE: - отрезок
  var param = new Array(x1,y1,z1, x2,y2,z2); - *две точки отрезка*

- var type = TM3_ARC | TMA_T1T2T3– *дуга окружности по 3-м точкам;*
  var param = new Array(x1,y1,z1, x2,y2,z2, x3,y3,z3);
  где: x1,y1,z1, x2,y2,z2, x3,y3,z3 - *Три последовательные точки дуги*

- var type = TM3_ARC | TMA_C_RAD_OR– *окружность: центр, радиус направление*
  var param = new Array(Cx, Cy ,Cz, R, Dir); -
  где: Cx, Cy ,Cz - *Координаты центра окружности,*
  R - *радиус окружности,*
  Dir - *0- ориентация против часовой стрелки, 1 — по часовой*

- var type = TM3_ARC | TMA_C_RAD_OR_T1T2– *дуга: центр, радиус, направление, точки начала и конца дуги*

var param = new Array(Cx, Cy ,Cz, R, Dir, x1,y1,z1, x2,y2,z2); -
где: Cx, Cy ,Cz - *Координаты центра окружности,*
R - *радиус окружности,*
Dir - *0- ориентация против часовой стрелки, 1 — по часовой*
x1,y1,z1, x2,y2,z2 - *точки начала и конца дуги*

- var type = TM3_ELLIPSE | TMA_C_SM_SP: - *эллипс: центр, размер оси, отношение осей*
var param = new Array(Cx, Cy ,Cz, R, S); -
где: Cx, Cy ,Cz - *Координаты центра эллипса,*
R - *размер оси*
S - *отношение размера второй оси к заданной (R)*

- var type = TM3_ELLIPSE | TMA_C_SM_SP_D_U1U2: - *дуга эллипса: центр, размер оси, отношение осей, направление дуги, два угла*
var param = new Array(Cx, Cy ,Cz, R, S, Dir, U1, U2);
где: Cx, Cy ,Cz - *Координаты центра эллипса,*
R - *размер оси*
S - *отношение размера второй оси к заданной (R)*
Dir - *0- ориентация против часовой стрелки, 1 — по часовой*
U1,U2 - *углы начала и конца дуги*

- var type = TM3_ELLIPSE | TMA_C_SM_SP_D_U1U2_N: - *дуга эллипса: центр, размер оси, отношение осей, направление дуги, два угла, вектор нормали плоскости эллипса*
var param = new Array(Cx, Cy ,Cz, R, S, Dir, U1, U2, Nx, Ny, Nz);
где: Cx, Cy ,Cz - *Координаты центра эллипса,*
R - *размер оси*
S - *отношение размера второй оси к заданной (R)*
Dir - *0- ориентация против часовой стрелки, 1 — по часовой*
U1,U2 - *углы начала и конца дуги*
Nx, Ny, Nz - *вектор нормали плоскости эллипса*

- var type = TM3_BEZIER – *кривая безье: задана двумя опорными точками и двумя контрольными*
var param = new Array(P1x, P1y , P1z, Pc1x, Pc1y ,Pc1z,
Pc2x, Pc2y ,Pc2z, P2x, P2y ,P2z);
где: P1 и P2 - *Координаты опорных точек*
Pc1 и Pc2 - *Координаты контрольных точек*

Нормаль к линии

Line. **GetNormal**(vect) ; *Получить вектор нормали линии*
var vect = new Array(vx, vy, vz) - *возвращаемый вектор нормали к линии*

Line. **SetNormal**(vx, vy, vz) ; *Установить вектор нормали линии*
vx, vy, vz - *заданный вектор нормали к линии*

Касательная к линии

Line. **GetTangentVectByPnt**(pnt, vect) ; *Получить вектор касательной к линии в заданной точке*
var pnt = new Array(x, y, z) - *координаты точки на линии*
var vect = new Array(vx, vy, vz) - *возвращаемый вектор касательной к линии*

Признак указываемости

var pickable = Line. **Pickable** ; *Получить признак указываемости объекта*

Line. **Pickable =** pickable *Установить признак указываемости объекта*
var pickable - *1 объект указывается, 0 - нет*

Подрезка линии

Line.**TrimLine**(x,y,z, mode) ; *Подрезать линию по точке*
var x,y,z – координаты точки
var mode – 0 подрезка начала линии; 1 – подрезка конца линии

Графический объект 2D/3D текст (AutoText)

var type = ModelText.**TypeModel**; *Получить тип объекта (TM3_TEXT3D)*

var IdModel = ModelText.**ID**; *Получить значение идентификатора объекта (ID)*

Строка текста

ModelText.**Text =** strText; *Установить строку текста*

var strText = ModelText.**Text**t; *Получить строку текста*

Высота шрифта

ModelText. **Height =** height; *Установить высоту шрифта в [мм]*

var height = ModelText.**Height** ; *Получить высоту шрифта*

Цвет текста

ModelText. **Color** = color; *Установить цвет текста (RGB)*

var color = ModelText. **Color**; *Получить цвет текста (RGB)*

Позиция текста

ModelText. **SetPositionXY**(x, y); *Установить позицию текста в плоскоти*

ModelText. **SetPosition**(position); *Установить позицию текста в пространстве*
var position = new Array(x,y,z);

Зона текста

ModelText. **SetZoneXY**(x1, y1, x2, y2); *Установить зону текста – отрезок задающий ориентацию и позицию текста*
где: x1, y1 - *координаты точки начала зоны*
x2, y2 - *координаты точки конца зоны*

Угол наклона текста

ModelText. **Angle** = angle; *Установить угол наклона строки текста*

var angle = ModelText. **Angle**; *Получить угол наклона строки текста*

Угол наклона текста

ModelText. **Angle** = angle; *Установить угол наклона строки текста*

var angle = ModelText. **Angle**; *Получить угол наклона строки текста*

Привязка текста

ModelText. **Alignment** = alignment; *Установить режим привязки текста*

var alignment = ModelText. **Alignment**; *Получить режим привязки текста*

var alignment - *параметры привязки:*

горизонтальная привязка

TAH_CENTER - *привязка по центру*
TAH_RIGHT - *горизонтальная привязка справа*
TAH_LEFT - *горизонтальная привязка слева*

вертикальная привязка

TAV_BOTTOM - *привязка по низу первой строки*
TAV_TOP - *привязка по верху габарита текста*
TAV_BOTTOM_BOX - *привязка по низу габарита текста*
TAV_CENTER - *привязка по центру габарита текста*

Растягивание текста

ModelText. **FitStretch** = mode; *Установить режим растягивания текста*

var mode = ModelText. **FitStretch**; *Получить режим растягивания текста*

var mode= *1 — растягивать строку текста до размеров зоны; 0 - нет:*

Сжатие текста

ModelText. **SetFitIn** = mode; *Установить режим сжатия текста*

var mode = ModelText. **SetFitIn**; *Получить режим сжатия текста*

var mode= *1 — сжимать строку текста до размеров зоны; 0 - нет:*

Автомасштаб текста

ModelText. **AutoScale** = mode; *Установить режим сжатия текста*

var mode = ModelText. **AutoScale**; *Получить режим сжатия текста*

var mode= *1 — текст автоматически рассчитывает свою высоту в соответствии с масштабом вида документирования; 0 - нет:*

Толщина 3D текста

ModelText. **Thickness** = thick; *Установить толщину 3D текста*

var thick = ModelText. **Thickness**; *Получить толщину 3D текста*

var thick= *толщина 3D текста*

Признак указываемости

var pickable = ModelText. **Pickable** ; *Получить признак указываемости текста*

ModelText. **Pickable =** pickable *Установить признак указываемости текста*

var pickable - *1 объект указывается, 0 - нет*

Графический объект размеры и обозначения (AutoDim)

var type = ModelDim.**TypeModel**; *Получить тип объекта (TM3_DIMENSION)*

var IdModel = ModelDim.**ID**; *Получить значение идентификатора объекта (ID)*

Задать цвет размера

ModelDim. **SetColor** (colorLine, colorText) ; *Задать параметры цвета*
var colorLine – цвет линий (RGB),
var colorText – цвет текста (RGB)

Задать параметры размера

ModelDim. **SetDimParam** (type, dir, name, points) ; *Задать параметры размера*
var type - *тип размера/обозначения :*
DIM_ELEV - высотная отметка
DIM_LINEAR - линейный размер
DIM_CHAIN - размерная цепочка
DIM_RADIAL - радиальный размер
DIM_ANGULAR - угловой размер
DIM_NOTE - текстовая выноска
DIM_DIAMETR - диаметр
DIM_AXIS - координационная ось
DIM_POINT - точечный маркер
DIM_ARC - длина дуги
DIM_MARKER_CIRCLE - круглый маркер

var dir - *направление измерения размера:* *DIM_DIR_X, DIM_DIR_Y, DIM_DIR_XY DIM_DIR_Z*

var name – *название размера (символьное обозначение)*
var points = new Array(x1,y1,z1, x2,y2,z2...); - *массив точек размера*

Именованные параметры размера

var Value = ModelDim.**Parameter** (nameParam); *Получить значение параметра объекта*
var nameParam - *строковое обозначение параметра (мнемоника)*

ModelDim.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*
var nameParam - *строковое обозначение размера*
var newValue - *новое значение параметра*

Задать точки размера

ModelDim. **SetPoints** (arrayParam, mode); *Задать точки размера*
var points = new Array(x1,y1,z1, x2,y2,z2...); - *массив точек размера*

Признак указываемости

var pickable = ModelDim. **Pickable** ; *Получить признак указываемости размера*

ModelDim. **Pickable =** pickable *Установить признак указываемости размера*
var pickable - *1 размер указывается, 0 - нет*

Фабрика объектов (AutoFactory)

var type = Factory.**TypeModel**; *Получить тип объекта (TM_FACTORY)*

Создать параметрический объект

var Object = Factory. **NewModel** (typeModel) ; *Создать параметрический объект*
var typeModel – *тип параметрической модели (см. Типы параметрических объектов)*
var Object - *созданный объект (IDispatch*)*

Создать графический объект

var Mode3D = Factory. **NewModel3D** (typeModel) ; *Создать графический объект*
var typeModel – *тип графической модели (см. Графические 2D и 3D объекты)*
var Object - *созданный объект (IDispatch*)*

Диалог выбора файла (AutoFileDialog)

var type = FileDialog.**TypeModel**; *Получить тип объекта (TM_FILEDLG)*

Название диалога

FileDialog. **DialogTitle** = title; *Установить название диалога*

var title = FileDialog. **DialogTitle**; *Получить название диалога*

Путь по умолчанию

FileDialog. **InitDir** = path; *Установить путь по умолчанию*

var path = FileDialog. **InitDir**; *Получить путь по умолчанию*

Фильтр расширений

FileDialog. **Filter** = filter; *Установить фильтр расширений, например,*
FileDialog. **Filter** *="JPEG image (*.jpg)|*.jpg|Bitmap image (*.bmp)|*.bmp||"*

var filter = FileDialog. **Filter**; *Получить фильтр расширений*

FileDialog. **Filter** = indexFilter; *Установить индекс текущего расширения*

var indexFilter = FileDialog. **Filter**; *Получить индекс текущего расширения*

Флаги

FileDialog. **Flags** = flags; *Установить флаги. Флаги установленные по умолчанию :*
OFN_EXPLORER | OFN_FILEMUSTEXIST | OFN_ENABLEHOOK | OFN_OVERWRITEPROMPT;

var flags = FileDialog. **Flags**; *Получить флаги*

Режим создания диалога

FileDialog. **ModeOpen** = mode; *Установить режим создания диалога*

var mode = FileDialog. **Filter**; *Получить режим создания диалога*
var mode = 1 – режим открытия файлов, 0 – режим записи файлов

Автоматическое расширение имени файла по умолчанию

FileDialog. **DefExt** = strExt; *Установить расширение файла, которое будет автоматически добавляться к имени файла, если оно явно не задано*

var strExt = FileDialog. **DefExt**; *Получить автоматическое расширение файла*

Получить имя файла

var strName = FileDialog. **GetFileName()**; *Получить имя файла*

Получить полный путь к файлу

var strPath = FileDialog. **GetPathName**(); *Получить полный путь к файлу*

Открыть диалог выбора файла

var code = FileDialog. **DoModal**(); *Открыть диалог выбора файла*

Вид документа (AutoSapfirView)

var type = SapfirView.**TypeModel**; *Получить тип объекта (TM_SAPVIEW)*

var SapfirDoc = SapfirView.**GetDocument** (); *Получить документ вида*

var DraftSheet = SapfirView. **GetDraftSheet** (); *Получить чертежный лист вида, если вид является чертежным*

var typeView = SapfirView.**TypeView** ; *Получить тип вида:*

typeView = VISTYPE_PLAN *- вид «План»*
VISTYPE_FASAD *- вид «Фасад»*
VISTYPE_3D *- вид «3D вид»*
VISTYPE_SECT *- вид «Сечение/Разрез»*
VISTYPE_DRAFT *- вид «Чертежный лист»*

Диалог настройки именованных параметров (AutoParametersDlg)

var type = ParDlg.**TypeModel**; *Получить тип объекта (TM_PARAMDLG)*

Именованные параметры

ParDlg. **SetMnParam** (param, mode); *Занести в диалог массив параметров*

var param = *new Array(); - массив параметров*

param[i] *– строковое обозначение i-го параметра (мнемоника)*
param[i+1] *– значение i-го параметра*
param[i+2] *– тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*
OBP_COMMENTS - *установить примечания параметров*

ParDlg. **GetMnParam** (param, mode); *Получить из диалога массив всех параметров*

var param = *new Array(); - возвращаемый массив параметров см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

ParDlg. **ClearMnParam** (0); *Удалить все дополнительные параметры*

Открыть диалог настройки параметров

var code = ParDlg. **DoModal**(); *Открыть диалог настройки параметров*

Инструментальная панель (AutoLocatorUser)

var type = LU.**TypeModel** *Получить тип объекта (TM_LOCUSER)*

Установить активный объект

LU.**SetActiveObj** (idObect) *установить активный объект*
var idObect – *идентификатор объекта (ID)*

var idObect = LU.**GetActiveObj** (idObect) *получить активный объект*
var idObect – *идентификатор объекта (ID)*

Режим работы локатора

LU. **PickMode =** mode *установить режим работы локатора*

var mode = LU. **PickMode** *получить режим работы локатора*
var mode =*1 — режим указания(выделения); 0 – режим ввода точек (создания)*

Контейнер слоев (AutoSetLayers)

var type = SetLayers.**TypeModel** *Получить тип объекта (TM_SET_LAYERS)*

var Layer = SetLayers.**NewLayer()** *Создать новый слой (TM_LAYER)*

var count = SetLayers.**CountLayers** *Получить количество слоев*

var Layer = SetLayers.**GetLayerByIndex (**index**)** *Получить слой по индексу (TM_LAYER)*

var count = SetLayers. **CountCombi** *Получить количество наборов слоев*

var name = SetLayers. **NameCombi**(index) *Получить название набора слоев*

var index = SetLayers. **GetCombiIndByName**(name) *Получить индекс набора по названию*

var Layer = SetLayers. **GetLayerById**(idLayer) *Получить слой по идентификатору*

Слой (AutoLayer)

var type = Layer.**TypeModel**; *Получить тип объекта (TM_LAYER)*

var IdLayer = Layer.**ID**; *Получить значение идентификатора объекта (ID)*

Имя слоя

Layer.**Name** = name *Установить имя слоя*

var name = Layer. **Name** *Получить имя слоя*

Расширение имени слоя

Layer. **Ext** = ext *Установить расширение имени слоя*

var ext = Layer. **Ext** *Получить расширение имени слоя*

Видимость слоя

Layer. **Invisible**(index) = invisible - *Установить признак невидимости слоя*

var invisible = Layer. **Invisible**(index) *Получить признак невидимости слоя*
var index - *индекс набора слоев*
var invisible = *1 - слой невидимый; 0 – слой видимый*

Блокировка слоя

Layer. **Locked**(index) = locked - *Установить признак блокировки слоя*

var locked = Layer. **Locked**(index) *Получить признак невидимости слоя*
var index - *индекс набора слоев*
var locked = *1 - слой заблокированный; 0 – нет*

Количество наборов

var count = Layer. **CombiCount** – *получить число наборов в которых используется данный слой*

Набор материалов (AutoSetMaterials)

var type = SetMaterials.**TypeModel**; *Получить тип объекта (TM_SET_MATERIALS)*

var Material = SetMaterials.**NewMaterial()**; *Создать новый материал (AutoMaterial)*

var Material = SetMaterials.**NewMultiMat ()**; *Создать новый многослойный материал (AutoMaterial)*

var count = SetMaterials.**CountMaterials** ; *Получить количество материалов*

var Material = SetMaterials.**GetMaterialByIndex** (index); *Получить материал по индексу*
var index - *индекс материала*

var Material = SetMaterials. **GetMaterialByID** (id); *Получить материал по ID*
var id - *идентификатор материала*

var Material = SetMaterials. **GetMaterialByGUID** (GUID); *Получить материал по GUID*
var *GUID* - *строка глобального идентификатора материала*

Материал (AutoMaterial)

var type = Material.**TypeModel**; *Получить тип объекта (TM_MATERIAL)*

var idMat = Material.**ID**; *Получить значение идентификатора материала (ID)*

var guidMat = Material.**Guid** ; *Получить глобальный идентификатор материала*

Material.**Name** = name *Установить имя материала*

var name = Material.**Name** *Получить имя материала*

var Value = Material.**Parameter** (nameParam); *Получить значение параметра материала*
var nameParam - *строковое обозначение параметра (мнемоника)*

Material.**Parameter** (nameParam) = newValue; *Установить значение параметра материала*
var nameParam - *строковое обозначение параметра (мнемоника)*
var newValue - *новое значение параметра*

Многослойный материал

var count = Material. **CountLayerMat** *Получить количество слоев многослойного материала*

var LayerMat = Material. **GetLayerMat** (index); *Получить слой материала по индексу (AutoLayerMat)*
var index - *индекс слоя материала*

var LayerMat = Material. **AddLayerMat** (id); *Добавить слой в материал*
var id – *идентификатор добавляемого материала*

var LayerMat = Material. **AddLayerMatGuid** (GUID); *Добавить слой в материал*
var GUID – *глобальный идентификатор добавляемого материала*

var LayerMat = Material. **ThickLayers** (GUID); *Получить суммарную толщину всех слоев*

Модуль Юнга

Material. **YoungModulus** = value *Установить значение модуля Юнга*

var value = Material. **YoungModulus** *Получить значение модуля Юнга*

Коэффициент Пуассона

Material. **PoissonRatio** = value *Установить значение коэффициента Пуассона*

var value = Material. **PoissonRatio** *Получить значение коэффициента Пуассона*

Плотность

Material. **Density** = value *Установить значение плотности [кг/м^3]*

var value = Material. **Density** *Получить значение плотности [кг/м^3]*

Коэффициент линейного расширения

Material. **Expansion** = value *Установить коэффициент линейного расширения*

var value = Material. **Expansion** *Получить коэффициент линейного расширения*

Цена

Material. **Price** = value *Установить цену*

var value = Material. **Price** *Получить цену*

Учет при прочностном расчете

Material. **PropConstruct** = value *Установить признак учета при прочностном расчете*

var value = Material. **PropConstruct** *Получить признак учета при прочностном расчете*

var value = 1 — *игнорировать*
2 — *нагрузка*
3 — *несущий конструктив*

Слой многослойного материала (AutoLayerMat)

var type = LayerMat.**TypeModel**; *Получить тип объекта (TM_LAYER_MAT)*

var Material = LayerMat.**GetMaterial** ; *Получить материал слоя (AutoMaterial)*

var idMaterial = LayerMat. **SetMaterialID** ; *Получить идентификатор материала слоя*

var guid = LayerMat. **SetMaterialGUID** ; *Получить глобальный идентификатор материала слоя*

LayerMat. **Thickness** = value *Установить толщину слоя*

var value = LayerMat. **Thickness** *Получить толщину слоя*

Геометрический вычислитель (AutoSolver)

var type = Solver.**TypeModel**; *Получить тип объекта (TM_SOLVER)*

Solver.**CrossSegSeg** (Line1, Line2, CrossParam); *Выполнить пересечение линий*

var Line1, Line2 - *пересекаемые линии (AutoLine)*
var CrossParam = new Array() - *возвращаемые параметры пересечения*
CrossParam.length / 5 - – количество точек пересечения
CrossParam.[i+0],[i+1],[i+2] – соответственно (x,y,z) i-й точки пересечения
CrossParam.[i+3] - параметр i-й точки пересечения на Line1
CrossParam.[i+4] - параметр i-й точки пересечения на Line2

var contour =Solver.**CutContour** (PL, PointsXY); *Разрезать контур, заданный полилинией PL, прямой линией, проходящей через две точки Points*

var PL - *пересекаемая полилиния (AutoPolyLine)*
var PointsXY = new Array(x1,y1, x2,y2) - *две точки прямой*
var contour *– возвращаемый массив контуров (AutoVectContour)*

var contour =Solver.**TrimLineByCont** (PL, Line); *подрезать линию по контуру*

var PL - *контур заданный полилинией (AutoPolyLine)*
var Line - *подрезаемая линия (AutoLine)*

Генератор геометрических моделей (AutoSurfCreator)

var type = SurfCreator.**TypeModel**; *Получить тип объекта (TM_SUCRE)*

Получить контейнер полилиний (AutoVectContour)

var VectContour = SurfCreator. **GetContour** (); *Получить контейнер полилиний*

Цвет поверхности

SurfCreator. **SetColor** (colorLine, colorFill); *Установить цвет создаваемой поверхности*
var colorLine – цвет ребер (линий) (RGB),
var colorFill — цвет заливки граней (RGBA)

Создать гранную поверхность

var Model = SurfCreator.**CreateSurface** (type, param); *создать сетку граней (AutoMesh)*
var type – *тип создаваемой поверхности:*
var param = *new Array() - набор параметров создаваемой поверхности:*

- ***создать призму по предварительно заданному контуру***
  предварительно необходимо в VectContour занести контур основания
  type=TMS_PRISM;
  param = new Array(thick); *- параметры*
  var thick – *толщина призмы*

- ***создать цилиндрическую поверхность (наклонная призма) по предварительно заданному контуру***
  предварительно необходимо в VectContour занести контур основания
  type=TMS_CYLINDER
  param = new Array(vx, vy, vz, thick); *- параметры*
  var vx, vy, vz – *вектор выдавливания (экструзии) поверхности*
  var thick – *величина выдавливания*

- ***создать сферу***
  type=TMS_SPHERE;
  param = new Array(R, DivP, DivM); *- параметры*
  var R – *радиус сферы*
  var DivP – *число разбиений по параллелям*
  var DivM– *число разбиений по меридианам*

- ***создать поверхность вращения по предварительно заданному контуру***
  предварительно необходимо в VectContour занести контур образующей
  type=TMS_PIVOTAL;
  param = new Array(); *- параметры (нет)*

- ***создать кинематическую поверхность по предварительно заданному контуру***
  предварительно необходимо в VectContour занести две линии: траекторию и образующую
  type=TMS_PROFILE;
  param = new Array(atrib); *- параметры*
  var atrib = smooth | fict;
  var smooth = *3- есть сглаживание боковой поверхности, 0 — нет*
  var fict = *4 - устанавливаются фиктивные ребра образующей, 0 - нет*

- ***создать пирамиду (конус) по предварительно заданному контуру***
  предварительно необходимо в VectContour занести контур основания
  type=TMS_CONE;
  param = new Array(x,y,z); *- координаты вершины конуса*

Матрица

var Matrix = Solver.**CreateMatrix ()**; *Получить объект матрица (AutoMatrix)*

Набор контуров (AutoVectContour)

var type = VectContour.**TypeModel**; *Получить тип объекта (TM_CONT)*

VectContour.**RemoveAll()**; *Удалить все полилинии из контейнера*

var count= VectContour.**CountLine**; *Получить количество полилиний в контейнере*

var PL = VectContour.**GetLine**(Index); *Получить полилинию (AutoPolyLine) по индексу*

var PL = VectContour.**NewPolyLine** (); *Создать новую полилинию в контейнере*

VectContour. **TrimLineByCont** (Points); *Обрезать отрезок по полилинии в контейнере*
var Points = new Array(x1, y1, z1, x2, y2, z2); *точки отрезка до и после операции*

VectContour. **GetBox** (valBox); *Получить охватывающий габарит полилиний*
var valBox = *new Array(min.x, min.y, min.z, max.x, max.y, max.z) – возвращаемое значение габаритного охватывающего бокса.*

VectContour. **GetCenterOfGravity**(pnt) ; *Получить координаты центра гравитации полилиний*
var pnt = new Array(vx, vy, vz) *- возвращаемые координаты центра гравитации*

VectContour. **Copy**(VC) ; *Скопировать массив полилиний VC в VectContour*
var VC – *массив полилиний источник*

VectContour. **Symmetry** (mode) ; *Симметрировать массив полилиний.*
var mode = *0 — симметрия относительно вертикальной оси*
1 — симметрия относительно горизонтальной оси

Параметрическое сечение (AutoMultiContour)

var type = MultiCont.**TypeModel**; *Получить тип объекта (TM_MULTICONT)*

Именованные параметры

MultiCont.**SetMnParam** (param, mode); *Занести в объект массив параметров, в том числе и дополнительные*
var arrayParam = *new Array();* *- массив параметров объекта*
param[i] *– строковое обозначение i-го параметра (мнемоника)*
param[i+1] *– значение i-го параметра*
param[i+2] *– тип i-го параметра, определяет способ его редактирования: см. Описание именованных параметров*
var mode - режим: OBP_VALUES - *установить значения параметров*
OBP_NAMES - *установить названия параметров*

OBP_COMMENTS - *установить примечания параметров*

MultiCont.**GetMnParam** (param, mode); *Получить из объекта массив всех параметров*

var param = *new Array(); - возвращаемый массив параметров объекта см. Описание именованных параметров*

var mode - режим: OBP_VALUES - *Получить значения параметров*
OBP_NAMES - *Получить названия параметров*
OBP_COMMENTS - *Получить примечания параметров*

var Value = MultiCont.**Parameter** (nameParam); *Получить значение параметра объекта*

var nameParam - *строковое обозначение параметра (мнемоника)*

MultiCont.**Parameter** (nameParam) = newValue; *Установить значение параметра объекта*

var nameParam - *строковое обозначение размера*

var newValue - *новое значение параметра*

MultiCont. **RemoveMnParam** (mode, param); *Удалить параметры объекта*

var mode – 0 – *все дополнительные параметры*
1 – *заданные в param в дополнительные параметры*
2 - *все дополнительные параметры между двумя заданными включительно*

var param – new Array(); - *массив мнемоник удаляемых параметров*

Очистка объекта

MultiCont. **CleanBaseModel** () ; *Функция удаляет данные из объекта (контура, дополнительные параметры и др.)*

Массив контуров объекта (AutoVectContour)

var Contour = MultiCont.**GetContour** (); *Получить массив контуров*

Получить приложение САПФИР (Application).

var SapfirApp = MultiCont.**GetApplication()**; *Получить приложение САПФИР*

Получить базовую модель (AutoModel).

var Object = MultiCont.**GetBaseModel ()**; *Получить базовый объект (AutoModel), к которому относится данный контур*

Матрица (AutoMatrix).

var type = Matrix.**TypeModel**; *Получить тип объекта (TM_MATR)*

Matrix.**SetValue**(valMatr, mode) ; *Установить значения матрицы*

Matrix.**GetValue**(valMatr) ; *Получить значения матрицы*

var valMatr = new Array(a11, a12, a13, a14,
a21, a22, a23, a24,
a31, a32, a33, a34,
a41, a42, a43, a44);

Matrix.**Translate** (x, y, z, mode) ; *Задать смещение*

var x, y, z - *величина смещения по соответствующим осям*
var mode = *0 – инициализация смещения*
1 – добавление смещения

Matrix. **Rotate** (x, y, z, mode) ; *Задать поворот*
var x, y, z - *углы поворота вокруг соответствующих осей*
var mode = *0 – инициализация поворота*
1 – добавление поворота

Matrix. **Scale** (x, y, z, mode) ; *Задать масштабирование*
var x, y, z - *коэффициенты масштабирования по соответствующим осям*
var mode = *0 – инициализация масштабирования*
1 – добавление масштабирования

Matrix. **Scale** (Xx, Xy, Xz, Yx, Yy, Yz, Zx, Zy, Zz, mode) ; *Задать преобразование системы координат*
Xx, Xy, Xz – *вектор оси X заданной системы координат*
Yx, Yy, Yz – *вектор оси Y заданной системы координат*
Zx, Zy, Zz – *вектор оси Z заданной системы координат*
var mode = *0 – инициализация преобразование*
1 – добавление преобразование

Matrix. **Symmetry** (param, mode) ; *Задать симметрию*
var param = new Array() - *параметры плоскости симметрии*
param[0], param[1], param[2] - *вектор нормали плоскости V(A,B,C)*
param[3] – *растояние от начала системы координат до плоскости (D)*
**Плоскость задается уравнением A*x+B*y+C*z+D=0;*
var mode = *0 – инициализация симметрию*
1 – добавление симметрию

Matrix. **Identity** () ; *Установить единичную матрицу*

Matrix. **Inverse** () ; *Получить инверсную матрицу*

Matrix. **MultPoints** (points) ; *Пересчитать точки через матрицу*
var points = new Array(x1,y1,z1, x2,y2,z2...); - *массив точек*

Описание именованных параметров

Каждый объект обладает определенным набором параметров. Параметры характеризуются мнемоникой (уникальным символьным обозначением), типом, значением, названием и комментарием. Используя мнемонику можно получить или установить значение параметра. Параметры могут быть встроенные и дополнительные. Используя метод *SetMnParam()*, можно в объект добавить произвольные дополнительные параметры.

Константы описаны в файле:
«C:\ProgramData\Lira Soft\Sapfir 1.3\Service\const_model.jsh».

Типы параметров

Простые

NPA_PT_STRING *- строка*
NPA_PT_INT *- целое число*
NPA_PT_FLOAT *- вещественное число*
NPA_PT_BOOLEAN *- логическое значение true / false*

Специальные

NPA_PT_COLORREF *- цвет (диалог настройки RGB)*
NPA_PT_RGBA *- цвет (диалог настройки RGBA)*
NPA_PT_COLORMAT *- материал (диалог настройки текстуры)*
NPA_PT_MATERIAL - материал (диалог настройки материала)
NPA_PT_LAYER - слой моделирования (диалог настройки слоя моделирования)
NPA_PT_LAYER_COMBI - набор слоев моделирования (диалог настройки набора слоев моделирования)
NPA_PT_SORTAMENT - металлический сортамент (диалог настройки металлического сортамента)
NPA_PT_SCALE - масштаб (диалог настройки масштаба)
NPA_PT_FONT - тип шрифта (диалог настройки шрифта)
NPA_PT_MNEMO *- параметр может принимать одно из заданных мнемониками опционных значений (см. пример)*
NPA_PT_SLIDER *- ползунок, значение в пределах [0..100]*
NPA_PT_FOLDER *- путь к папке (диалог выбора пути)*

Пути к файлам

Для выбора файлов используется универсальный диалог выбора файлов. В зависимости от типа параметра, автоматически настраиваются фильтры файлов в диалоге.

NPA_PT_FILEPATH *- путь к файлам*
NPA_PT_FILEPATH_3D *- путь к файлам 3D моделей (obn, 3ds).*
NPA_PT_FILEPATH_IMAGE *- путь к файлам растровых изображений (jpg, bmp)*

Дополнительно, сразу после выше указанных параметров, могут быть заданы специальные сервисные параметры:

NPA_PT_TYPELIBPATH *- путь по умолчанию настраивается к файлам библиотеки в соответствии со значением параметра (см. Служебные папки САПФИР)*
NPA_PT_FILEFILTER *- фильтр для файлов. В значении параметра необходимо задать строку фильтра, например: "JPEG image (*.jpg)|*.jpg|Bitmap image (*.bmp)|*.bmp||"*

Сервисные типы

NPA_UNDEFINED - *неопределенный параметр*
NPA_GROUP_OPEN - *открыть группу параметров (см. примеры)*
NPA_GROUP_CLOSE - *закрыть группу параметров*

Свойства параметров

Свойства параметра определяют дополнительные особенности параметра. Свойство задается при добавлении параметра (см. SetMnParam()).

Единицы измерения длины

NPA_PU_D1_S - *малые [мм]*
NPA_PU_D1_M - *средние [мм*]
NPA_PU_D1_L - *большие [м]*

Единицы измерения площади

NPA_PU_D2_S - *малые [мм^2]*
NPA_PU_D2_M - *средние [м^2]*

Прочие единицы измерения

NPA_PU_D3_M - *единицы измерения объема [м^3]*
NPA_PU_ANGLE - *угловые величины [°]*
NPA_PU_TIME - *время*
NPA_PU_FORCE - *сила*
NPA_PU_MASS - *масса*

Прочие свойства

NPA_HIDDEN - *скрытый параметр*
NPA_READONLY - *параметр только для чтения*
NPA_SERVICE - *сервисный параметр -дополнения к основному параметру, (см. пример)*
NPA_DONT_SAVE - *не сохранять в файл*
NPA_AUTO_REGEN - *при модификации параметра необходимо вызвать внешнюю функцию авторегенерации модели. Используется для параметрических объектов, созданных внешними программами*
NPA_OPTIONAL - *опционное значение параметра(см. пример)*
NPA_VALUE_LIST - свойство группы — показывает список значений всех параметров в группе

Примеры

Занесение простых параметров в объект

```
var param = new Array();      // Объявили массив для параметров
        // первый параметр (существующий )
param[0] = "M_NAME";                // Мнемоника параметра объекта
param[1] = "Стена внешняя";         // Значение параметра
param[2] =  NPA_PT_STRING;          // Тип параметра
        // второй параметр (новый)
param[3] = "MY_CODE";               // мнемоника  дополнительного параметра
param[4] = 123;                     // Значение параметра
param[5] =  NPA_PT_INT | NPA_READONLY;      // Тип параметра + свойство
        // третий параметр (новый)
param[6] = "MY_MASS";               // мнемоника  дополнительного параметра
param[7] = 75;                      // Значение параметра
param[8] =  NPA_PT_FLOAT | NPA_PU_MASS ;    // Тип параметра + свойство
Obj.SetMnParam(ParamObj, OBP_VALUES);       // Занести параметры в объект Obj

Obj.SetMnParam(ParamObj, OBP_OBP_NAMES);          // Занести названия параметров в
                                                  // объект Obj
```

Занесение перечисляемых (опционных) параметров в объект

```
var param = new Array();      // Объявили массив для параметров
// задаем параметр
param.push("M_MODE");         // Мнемоника параметра,
param.push("M_AUTO");         // Значение параметра – это мнемоника одного из опционных
                              // значений
param.push(NPA_PT_MNEMO  | NPA_OPTIONAL);         // Тип параметра

// задаем массив зарезервированных опционных  значений параметра
param.push("M_AUTO");               // Мнемоника первого зарезервированного значения
param.push("автоматически");        // Название первого значения
param.push(NPA_OPTIONAL);           // Тип параметра - опционное значение

param.push("M_MANUAL");             // Мнемоника второго зарезервированного значения
param.push("вручную");              // Название второго значения
param.push(NPA_OPTIONAL);           // Тип параметра - опционное значение

param.push("0");   param.push("0");   param.push(0);        // Пустой параметр - конец списка
                                                            // зарезервированных значений
```

Занесение группы параметров в объект

```
var param = new Array();      // Объявили массив для параметров
        // открываем группу
param.push("GR_O");         param.push("");         param.push(NPA_GROUP_OPEN |
NPA_VALUE_LIST);
        // заносим параметры в группу
param.push("VAL_A");        param.push("A");        param.push(NPA_PT_STRING);
param.push("VAL_B");        param.push("B");        param.push(NPA_PT_STRING);
param.push("VAL_C");        param.push("C");        param.push(NPA_PT_STRING);
        // закрываем группу
param.push("GR_O");         param.push("");         param.push(NPA_GROUP_CLOSE);
```

Задание сервисных параметров

```
var param = new Array();                    // Объявили массив для параметров
param.push("M_PATH_MODEL");                 // Мнемоника основного параметра
param.push("OP15_12.OBN");                  // Файл модели 3D
param.push(NPA_PT_FILEPATH_3D);             // Тип параметра — 3D файл

param.push("M_DATA_PATH");                  // Мнемоника сервисного  параметра
param.push(TP_WINDOWS);                     // Папка окон
param.push(NPA_SERVICE | NPA_PT_TYPELIBPATH | NPA_READONLY | NPA_HIDDEN);
        // где:  NPA_SERVICE                - сервисный параметр,
        //       NPA_PT_TYPELIBPATH         - путь к стандартной библиотеке моделей
        //       NPA_READONLY               - параметр только для чтения
        //       NPA_HIDDEN                 - скрытый параметр
```

Задание сервисных параметров

```
var ParamName = new Array();                // Объявили массив для названий параметров
ParamName.push("FR_GRO");                   // мнемоника параметра
ParamName.push("Группа");                   // название параметра
ParamName.push(0);                          // Тип параметра (не обрабатывается)

ParamName.push("M_MODE");                   // мнемоника параметра
ParamName.push("Режим");                    // название параметра
ParamName.push(0);                          // Тип параметра (не обрабатывается)
Obj.SetMnParam(ParamName, OBP_NAMES);       // Занести названия параметров в объект
```

Получение значения существующего параметра объекта

```
var arrayParam = new Array();               // Объявили массив для параметров
Obj.GetMnParam (arrayParam, OBP_VALUES);    // Получить из объекта массив значений
                                            // всех параметров

var Value = Obj.Parameter ("M_NAME");       // Получить значение параметра объекта
```

Установка значения существующего параметра объекта

```
Obj.Parameter ("M_NAME") = "Плита";         // Установить значение параметра объекта
```

Служебные папки САПФИР

Константы описаны в файле «C:\ProgramData\Lira Soft\Sapfir 1.3\Service\const_model.jsh».

TP_INITIAL - *папка с файлами начальных настроек*
TP_FONTS - *папка с чертежными шрифтами*
TP_SERVICE_HTML - *папка с сервисными файлами HTML*
TP_PALETTE - *папка с файлами палитр*
TP_TEXTURES - *папка с файлами текстур*
TP_LIBRARY - *папка глобальной библиотеки*
TP_OBJECTS - *папка с библиотечными объектами*
TP_DOORS - *папка с библиотечными объектами «Двери»*
TP_WINDOWS - *папка с библиотечными объектами «Окна»*
TP_COLUMNS - *папка с библиотечными объектами «Колонны»*
TP_STAIRS - *папка с библиотечными объектами «Лестницы»*
TP_BEAMS - *папка с библиотечными объектами «Балки»*
TP_DRAWINGS - *папка с файлами фрагментов чертежей*
TP_APPLICATION - *папка приложения*
TP_AUTOSAVE - *папка с файлами автосохранения*
TP_UNDO - *папка с файлами undo*
TP_PLUGINDLL - *папка с файлами DLL плагинов*
TP_LIRA_BIN - *папка с файлами DLL LIRA-KM*