ADOBE® AFTER EFFECT® 脚本设计

参考手册

此参考手册为翻译 Adobe 官方文档《ADOBE® AFTER EFFECTS® CS6 SCRIPTING GUIDE》，仅供参考和学习。作者保留翻译权和著作权。

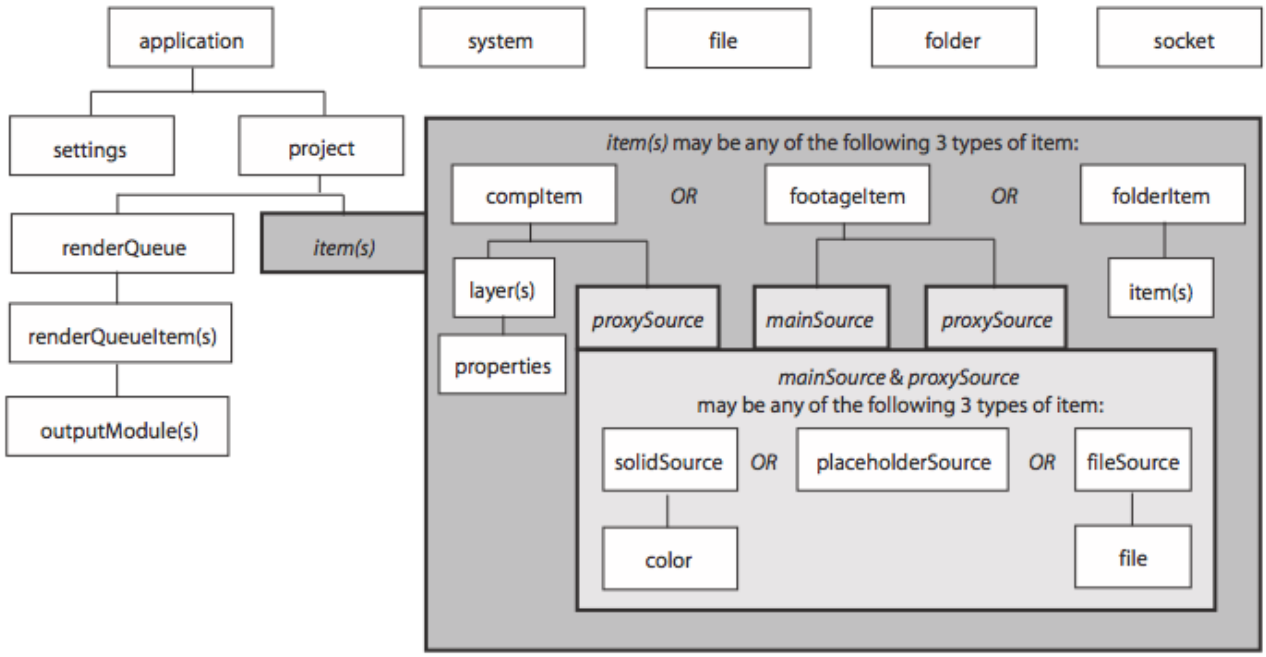
当前版本：V1.0.0

发布时间：08/04/2017

作者博客：worldcter.com

目录

对象模型



翻译规范：

application：应用

system：系统

file：文件

folder：文件夹

socket：socket

settings：设置

project：工程

renderQueue：渲染队列

renderQueueItem(s)：渲染项目

outputModule(s)：输出模块

item(s)：项目

compItem：合成项目

footageItem：素材项目

folderItem：文件夹项目

layer(s)：层

properties：属性

proxySource：代理源

mainSource：真实源

solidSource：层源

placeholderSource:占位符源

fileSource：文件源

color：颜色

全局函数

这些全局可用的函数在 AE 中是一个特殊的存在，任何 Javascript 对象或者函数都可以调用这些全局函数，它可以允许你在信息面板(Info panel)内显示三行以内的文字，还能把时间值在数字和字符串两种数据类型之间转换。

全局函数	描述
clearOutput()	清除信息面板中的文字
currentFormatToTime()	把字符串型的时间值转换为数字型
timeToCurrentFormat()	把数字型的时间值转换为字符串型
write()	在信息面板上显示文字，结尾不加换行
writeLn()	在信息面板上显示文字，结尾添加换行
isValid()	如果返回 true，那么指定的对象存在

clearOutput()

全局函数：clearOutput()

描述：清除信息面板中的文字

参数：无

返回：无

currentFormatToTime()

全局函数：currentFormatToTime(formattedTime,fps,isDuration)

描述：

把一个指定帧的时间值从字符型转换到秒数，需要给定一个帧速率。举个例子，如果当前帧时间值为 0:00:12(这个字符型时间值决定于工程设置)，并且帧速率为 24fps，那么转换后的结果就会是 0.5 秒(12/24)。如果帧速率为 30fps，那么时间就会是 0.4 秒(12/30)。

如果这个时间是持续的，那么这些帧就会从 0 开始计算。否则，就会从工程的起始帧计算。

参数：

formattedTime	指定帧的当前时间值，字符串类型，指定的数值类型取决于当前工程显示时间的格式
fps	每秒的帧数，是一个浮点型值
isDuration	可选的，当值为 true 时，时间是持续的（以 0 帧开始计算）。当值为 false（默认）时，时间从工程初始帧开始计算。

返回：浮点型，秒数。

笔记：第一个参数，实际上就是时间轴面板左上角那个时间值，默认的是“小时：分钟：秒：帧”的格式，底下还会有一个当前帧的数值。这个参数不管是用实际时间值还是当前帧，其实作用都是将当前时间转换成秒数，方便脚本里其他函数的计算。

isValid()

全局函数：isValid(obj)

描述：

判断指定的 AE 对象（比如合成、层、遮罩等等）是否存在。一些操作，比如 PropertyBase moveTo()方法，可能会使参数指定的相关对象失效。这个函数允许你在调用对象之前测试一下它们是否有效。

参数：

obj	一个 AE 的对象，用来检查是否有效

返回：布尔值

举例:

```
var layer = app.project.activeItem.layer(1); // 假定这个层有三个遮罩
alert(isValid(layer)); // 显示“true”
var mask1 = layer.mask(1);
var mask2 = layer.mask(2);
var mask3 = layer.mask(3);
mask3.moveTo(1); // 把第三个遮罩移动到遮罩层的顶部位置
alert(isValid(mask1)); // 显示“false” ； 遮罩2 和遮罩3 也时一样的结果
```

笔记：第一个警告框显示 true 没什么问题，第二个警告框之所以是 false，是因为 mask1 的值变了，选中层按 M，能看到遮罩层也是有顺序的，一开始把第一个遮罩命名为 mask1，现在 mask1 的值变成了第三个遮罩，所以会返回 false。注意的是返回 false 是因为值发生了变化，而不是简单的因为遮罩命名发生了变化，所以即便把处在第一层的遮罩 3 改名为遮罩 1，还是会返回 flase。

timeToCurrentFormat()

全局函数：timeToCurrentFormat(time,fps,isDuration)

描述：currentFormatToTime 的反函数。

参数：与 currentFormatToTime 相同

返回：字符串型，当前工程的时间格式

write()

全局函数：write(text)

描述：在信息面板上显示文字，结尾不加换行

参数：

text	要显示的字符串，如果太长以至于超过了信息面板的限制，那么会被删减

返回：无

举例：

```
write("This text appears in Info panel ");
write("with more on same line.");
```

wirteLn()

全局函数：writeLn(text)

描述：在信息面板上显示文字，结尾加换行

参数：

text	要显示的字符串

返回：无

应用对象

app

描述：

提供一个 AE 内部对象和应用设置的访问接口。通过使用它的名称，这个全局对象随时都可以被调用，app.

在 AE 内部，应用对象的属性提供了一些指定对象的访问接口，应用对象的方法可以创建工程、打开已有工程、控制浏览文件夹模式、清理内存和退出 AE 等等。当 AE 退出时，它会关闭当前的工程，询问用户是否保存当前工程或必要时放弃这个工程，并且必要时创建一个工程文件。

属性

属性	描述
project	AE 当前工程文件
isoLanguage	当前运行的程序的本地化信息（语言和地区）
version	AE 的版本号
buildName	AE 的内部版本名称
buildNumber	AE 的内部版本号
isWatchFolder	当值为 true 时，AE 正在以查看文件夹模式运行
isRenderEngine	当值为 true 时，AE 正在作为渲染引擎运行
settings	可以用脚本来改变 AE 的设置
onError	当 AE 中发生错误时被调用的一个回调函数
exitCode	外部执行脚本时生成的一个数字代号
exitAfterLaunchAndEval	当值为 true 时，在 windows 上用命令行执行一个脚本后 AE 仍保持开启状态
saveProjectOnCrash	当值为 true 时，AE 意外退出时工程会自动保存
memoryInUse	AE 使用的内存
effects	AE 当前可用的效果

activeViewer	当前激活的或最后一次激活的预览面板

方法

方法	描述
newProject()	创建一个新工程
open()	打开一个工程或打开工程对话框
quit()	退出 AE
watchFolder()	开启查看文件夹模式，直到退出模式之前都没有返回值
pauseWatchFolder()	暂停当前查看文件夹进程
endWatchFolder()	结束当前查看文件夹进程
purge()	清理缓存信息的某种目标类型（重复清理选项在编辑菜单栏）
beginUndoGroup()	记录下动作以供之后撤销用
endUndoGroup()	结束记录动作，当一个脚本包含不止一个撤销动作组时才会用到
beginSuppressDialogs()	开始在 UI 界面抑制对话框
endSuppressDialogs()	结束在 UI 界面抑制对话框
setMemoryUsageLimits()	设置内存使用限制，等同于首选项中的内存和缓存面板
setSavePreferencesOnQuit()	设置当 AE 退出时是否保存首选项的改动
activate()	让 AE 主窗口显示在桌面上
scheduleTask()	安排脚本延迟执行
cancelTask()	取消脚本延迟执行命令
parseSwatchFile()	从 Adobe Swatch Exchange(ASE)中导入一个颜色样板

app.activate()

应用对象方法：app.activate()
描述：如果 AE 处在最小化或者图标化状态下，打开 AE 主窗口，并且显示到桌面上。
参数：无
返回：无

app.activeViewer

应用对象属性：app.activeViewer
描述：当前使用的或刚刚使用过的预览窗口对象（合成预览窗口、层预览窗口、素材预览窗口）。如果没有预览窗口就返回 null
类型：预览窗口对象，只读

app.beginSuppressDialogs()

应用对象方法：app.beginSuppressDialogs()
描述：开始在 UI 界面抑制脚本错误对话框，使用 endSuppressDialogs()来恢复显示错误对话框。具体查看 endSuppressDialogs()方法
参数：无
返回：无

app.beginUndoGroup()

应用对象方法：app.beginUndoGroup(undoString)
描述：
标记一个撤销组的开始位置，这个标记允许脚本能够一次性撤销所有撤销组内的动作（使用编辑->撤销/恢复菜单）使用 endUndoGroup()方法来标记组的结束位置。
beginUndoGroup()和 endUndoGroup()能够被嵌套。组内的撤销组成为上一级撤销组的一部分，同时也能够正确地撤销。在这种情况下，组内撤销组的名称会被忽略。
参数：

undoString	这个文本会显示在编辑菜单的撤销命令上（就像是“撤销<undoString>”）

返回：无

app.buildName

应用对象属性：app.buildName
描述：AE 开始运行时的构造名，用于 Adobe 公司内部测试查找修复错误使用
类型：整数，只读

app.cancelTask()

应用对象方法：app.cancelTask(taskID)
描述：从延迟执行的任务计划的队列中移除指定的任务
参数：

taskID	一个用来识别指定任务的整数，由 app.scheduleTask()返回

返回：无

app.effects

应用对象属性：app.effects
描述：应用中可用的效果
类型：数组，每个元素都包含以下属性，只读

displayName	字符串型，在 AE 效果菜单中显示的效果名称
category	字符串型，AE 效果菜单中的分类，不显示在菜单中的自定义效果值为空
matchName	字符串型，每个效果都有一个单独的 matchName，这个名称不会因为 AE 的版本而改变，使用这个值来添加效果

举例：

```
var effectName = app.effects[12].displayName;
```

笔记：effects 后面的数字，代表着效果菜单里的第几个效果，中文版和英文版顺序不同，甚至有的时候因为装了插件的原因也会导致效果顺序的改变。当然这只是一个属性，返回一个值。如果想给某个层添加某个效果，有专门的函数。

app.endUndoGroup()

应用对象方法：app.endUndoGroup()
描述：

标记以 app.beginUndoGroup()开始的撤销组的结束位置。如果想在脚本中使用多个撤销组，你可以在脚本中间使用结束撤销组的方法（而不仅仅是用在脚本结尾处）。
如果你只在脚本中使用一个撤销组，那么就不需要在脚本中使用这个方法，因为在脚本结束后，系统会自动结束这个撤销组。
在没有使用 beginUndoGroup()方法的情况下调用这个方法，会产生一个错误。
参数：无
返回：无

app.endWatchFolder()

应用对象方法：app.endWatchFolder()
描述：结束查看文件夹模式
参数：无
返回：无
具体查看：watchFolder()、parseSwatchFile()、isWatchFolder

app.exitAfterLaunchAndEval

应用对象属性：app.exitAfterLaunchAndEval
描述：
只有在 Windows 平台用命令行执行脚本的时候才会用到这个属性。当用命令行启动 AE 时，-r 或-s 命令行标记让 AE 运行一个脚本（从一个文件或一段字符串中）。
如果这个属性值为 true,AE 会在执行脚本后关闭，如果值为 false，AE 就不会自动关闭。
这个属性只在 Windows 命令行运行 AE 中才生效，Mac OS 中无效。
类型：布尔，读/写

app.exitCode

应用对象属性：app.exitCode
描述：
当从外部执行脚本时生成的一个数字代号（比如从命令行或者 AppleScript）。

- 在 Windows 中，用命令行启动 AE 时会返回这个值（使用 afterfx 或者 afterfx -m 命令），并且有一个脚本用-r 或-s 选项来指定。
- 在 Mac OS 中，每个脚本在 AppleScript 中 DoScript 后的结果会返回这个值。

在 Windows 和 Mac OS 两个平台中，当每个脚本开始计算时值会被设为 0（EXIT_SUCCESS）。当某个脚本执行时发生一个错误的时候，这个脚本可以设置这个值为一个正整数值来标记发生错误。
类型：整数，读/写

举例：

```
app.exitCode = 2;//当值为 2 时，有一个错误产生了，并且关闭 AE
```

app.isoLanguage

应用对象属性：app.isoLanguage

描述：一个字符串，标示着当前运行的 AE 的本地化信息（语言和地区称号）。

注意：$.locale 返回当前操作系统的语言，不是 AE 的语言。

类型：

字符串，只读，一些常见的值包括：

- en_US，英语（美国）
- de_DE，德语（德国）
- es_ES，西班牙语（西班牙）
- fr_FR，法语（法国）
- it_IT，意大利语（意大利）
- ja_JP，日语（日本）
- ko_KR，韩语（韩国）

举例：

```
var lang = app.isoLanguage;
if (lang == "en_US")
        alert("AE 正在使用英语运行。");
else if (lang == "fr_FR")
        alert("AE 正在使用法语运行。");
else
        alert("AE 不是以英语或者法语运行。");
```

app.isRenderEngine

应用对象属性：app.isRenderEngine

描述：如果 AE 正在以渲染引擎运行时返回 true

类型：布尔，只读

app.isWatchFolder

应用对象属性：app.isWatchFolder

描述：如果正在显示查看文件夹对话框或者 AE 正在访问一个文件夹作为渲染输出，那么就返回 true

类型：布尔，只读

app.memoryInUse

应用对象属性：app.memoryInUse
描述：正在 AE 使用的内存量
类型：数字，只读

app.newProject()

应用对象方法：app.newProject()
描述：
在 AE 中创建一个新工程，等同于菜单中 File->New->New Project 命令。
如果当前工程还未保存，AE 会提示用户是否保存当前工程。如果用户点击了取消按钮，新工程就不会被创建并且方法会返回 null。打开新工程之前使用
app.project.close(CloseOptions.DO_NOT_SAVE_CHANGES)来关闭当前工程，具体查看工程对象方法 close()。
参数：无
返回：一个新工程，如果新工程没能被创建则返回 null
举例：

```
app.project.close(CloseOptions.DO_NOT_SAVE_CHANGES);
app.newProject();
```

app.onError

应用对象属性：app.onError
描述：
当发生错误时被调用的回调函数的名称。通过创建一个函数并且分配它一个这个属性，你就可以有计划的对错误做出回应。比如，你可以在渲染时关闭或重启 AE 而错误日志不会记录任何信息。具体查看 RenderQueue render()方法。
这个回调函数只传递错误信息和字符串类型，它不需要返回任何值。
类型：函数名的字符串型，如果没有函数被分配这个方法则返回 null，读/写
举例：

```
function err(errString) {
         alert(errString);
         }
app.onError = err;
```

app.open()

应用对象方法：app.open(file)

描述：打开一个工程

参数：

file	可选，一个工程文件的 ExtendScript 文件对象，如果缺省，这个方法会提示用户选择一个工程文件。

返回：一个新的指定工程的工程对象，如果用户取消了选择文件对话框，则返回 null

举例：

```
var my_file = new File("../my_folder/my_test.aep");
 if (my_file.exists){
    new_project = app.open(my_file);
    if (new_project){
        alert(new_project.file.name);
        }
    }
```

app.parseSwatchFile()

应用对象方法：app.parseSwatchFile(file)

描述：从 Adobe Swatch Exchange(ASE)文件中导入颜色样本数据。

参数：

file	文件的说明，一个 ExtendScript 文件对象

返回：

样本数据，格式如下：

data.majorVersion data.minorVersion	ASE 的版本数字
data.values	样本值的数组
SwatchValue.type	“RGB”、“CMYK”、“LAB”、“Gray”其中一个
SwatchValue.r SwatchValue.g SwatchValue.b	当 type = “RGB”时，颜色值在[0.0~1.0]范围内。 0,0,0 为黑色
SwatchValue.c SwatchValue.m SwatchValue.y SwatchValue.k	当 type = “CMYK”时，颜色值在[0.0~1.0]范围内。 0,0,0,0 为白色
SwatchValue.l SwatchValue.a SwatchValue.b	当 type = “LAB”时，颜色值 L 在[0.0~1.0]范围内，a 和 b 在[-128.0~+128.0]范围内。 0,0,0 为黑色

SwatchValue.value	当 type = “Gray”时，颜色值在[0.0~1.0]范围内。 0.0 为黑色

app.pauseWatchFolder()

应用对象方法：app.pauseWatchFolder(pause)
描述：停止或恢复查找用来定位渲染输出位置的查看文件夹模式。
参数：

pause	值为 true 时为暂停，值为 false 时为恢复

返回：无
具体查看：isWatchFolder、watchFolder()、endWatchFolder()

app.project

应用对象属性：app.project
描述：当前使用的工程，这个应用对象属性本身也是一个对象，而且它的属性和方法很多，会单独分为一个大章节来说明。
类型：工程对象，只读

app.purge()

应用对象方法：app.purge(target)
描述：从内存中清理指定类型的未使用数据。等同于编辑菜单中的内存选项。
参数：

target	从内存中清理的元素类型，主要有以下几种类型 ● PurgeTarget.ALL_CACHES：清除 AE 在内存中的所有缓存数据 ● PurgeTarget.UNDO_CACHES：清理所有撤销组的缓存 ● PurgeTarget.SNAPSHOT_CACHES：清理所有层或合成的快照缓存 ● PurgeTarget.IMAGE_CACHES：清理所有保存的图片数据

返回：无

app.quit()

应用对象方法：app.quit()

描述：退出 AE
参数：无
返回：无

app.saveProjectOnCrash()

应用对象属性：app.saveProjectOnCrash()
描述：当值为 true（默认）时，AE 会试图在因为发生错误导致意外关闭时弹出对话框允许你保存当前工程。值为 false 时，AE 不会弹出对话框并且不会保存直接关闭。
类型：布尔。读/写

app.scheduleTask()

应用对象方法：app.scheduleTask(stringToExecute,delay.repeat)
参数：

stringToExecute	一段要执行的 JavaScript 代码
delay	浮点型，执行代码之前要延迟的毫秒数
repeat	当值为 true 时，重复执行代码，间隔时间为指定的毫秒数。如果值为 false，脚本只会被执行一次。

返回：整数，这个任务的唯一标识符，可以被 app.cacelTask()方法用来取消执行代码。

app.setMemoryUsageLimits()

应用对象方法：app.setMemoryUsageLimits(imageCachePercentage,maximumMemoryPercentage)
描述：设置内存使用限制，等同于首选项中的内存&缓存。对于所有参数的值，如果安装的 RAM 比设置的小，那么值被视为是安装 RAM 的 100%。否则就是 n.的百分数。n.值为：32 位 Windows 中为 2GB，64 位 Windows 中为 4GB，Mac OS 中为 3.5GB。
参数：

imageCachePercentage	浮点数，分配图形缓存的内存百分比
maximumMemoryPercentage	浮点数，能够使用的内存的最大百分比

返回：无

app.setSavePreferencesOnQUit()

应用对象方法：app.setSavePreferenceOnQuit(doSave)
描述：设置或清除当 AE 退出时是否要保存首选项的改动
参数：

daSave	当值为 true 时，AE 退出时保存首选项。false 则相反

返回：无

app.settings

应用对象属性：app.settings
描述：当前导入的设置。同 app.project 一样，会有单独的章节来说明。
类型：设置对象，只读

app.version

应用对象属性：app.version
描述：标识当前运行的 AE 版本号的字符串
类型：字符串，只读
举例：

```
var ver = app.version;
alert("这台电脑上正在运行的 AE 版本是" + ver);
```

app.watchFolder()

应用对象方法：app.watchFolder(folder_object_to_watch)
描述：在指定的文件夹开始一个查看文件夹进程
参数：

folder_object_to_watch	要查看的文件夹的 ExtendScript 文件夹对象

返回：无
举例：

```
var theFolder = new Folder("c:/tool");
app.watchFolder(theFolder);
```

具体查看：endWatchFolder()、parseSwatchFile()、isWatchFolder

媒体项目对象

app.project.item(index)

描述：

媒体项目对象提供了调用导入到 AE 中的音频/视频文件的属性和方法的接口。

- 媒体项目对象是项目对象的子集。所有项目对象的方法和属性都对媒体项目对象适用。
- 媒体项目对象是合成项目对象和素材项目对象的基础类，所以媒体项目对象的属性和方法也同样适用合成项目对象和素材项目对象。

属性：

属性	描述
name	显示在工程面板中的对象名称
width	项目的宽度
height	项目的高度
pixelAspect	项目的像素纵横比
frameRate	项目的帧速率
frameDuration	项目的帧持续时间
duration	项目的总持续时间
useProxy	当值为 true 时，当前项目正在使用一个代理源
proxySource	被当前项目用作代理的素材项目对象
time	项目的当前时间
usedIn	使用这个项目的合成项目对象
hasVideo	当值为 true 时，项目含有视频
hasAudio	当值为 true 时，项目含有音频
footageMissing	当值为 true 时，没有找到这个项目或者这个项目是个占位符

方法

方法	描述
setProxy()	为这个项目设置一个代理
setProxyWithSequence()	设置一个序列作为这个项目的代理
setProxyWithSolid()	设置一个固态层作为这个项目的代理
setProxyWithPlaceholder()	设置一个占位符作为这个项目的代理
setProxyToNone()	移除这个项目的代理

笔记：

项目对象(Item)就目前来看，指的就是工程面板中(Project)的全部素材。有的素材是音视频(AVItem)，有的是文件夹(folderItem)，有的是合成(compItem)，它们各自属性不同，有单独属于自己的属性和方法，但是它们也有共同点，所以有些属性和方法也是可以共用的,因此说他们有着包含关系。

就像第一页的对象模型一样。项目对象(Item)包含了媒体项目对象(AVItem)和文件夹项目对象(folderItem)。媒体项目对象又是合成项目对象(compItem)和素材项目对象(footageItem)的父集。这种不同类型项目的分类只是意义上的分类，实际使用的时候不区分什么类型，统一使用app.item(index)。只是有时在调用一个文件夹的持续时长会返回未定义(undifined)。

index 是指在工程面板中的顺序，从 1 开始。这个顺序只与排序类型有关，与倒序还是顺序无关。

app.project.item().duration

媒体项目对象属性：app.project.item(index).duration

描述：

返回项目持续时间的秒数，静止素材项目持续时间为 0。

- 在合成项目对象中，这个值被链接到合成的持续时间。读/写
- 在素材项目对象中，这个值被链接到真实源(mainSource)的持续时间。只读

类型：在[0.0~10800.0]范围内的浮点型。在合成项目对象中为只读，其他情况为读/写。

app.project.item().footageMissing

媒体项目对象属性：app.project.item(index).footageMissing

描述：当值为 true 时，这个媒体对象是一个占位符，或者代表素材的源文件丢失。这种情况下，使用素材项目源文件对象的 missingFootagePath 可以获取丢失文件的路径。具体查看素材项目对象 mainSource 属性和文件源对象 missingFootagePath 属性。

类型：布尔值，只读

app.project.item().frameDuration

媒体项目对象属性：app.project.item(index).frameDuration
描述：
返回这个媒体项目的而帧长度。这个值是帧速率的倒数。当写入值时，这个值的倒数会自动设为新的帧速率值。
这个属性返回帧速率的倒数，如果这个值不能被 1.0 整除（比如 0.3），或许会和你设置的不一样。因为数的限制，(1/(1/0.3))结果很接近但不是 0.3。
如果这个媒体项目是一个素材项目，这个值会被链接到源素材对象上，只读。可以设置源素材对象的 conformFrameRate 来改变。这个设置同时改变了素材项目的帧速率和帧持续时间。
类型：浮点型，范围[1/99~1.0]，素材项目中为只读，其他情况为读/写。

app.project.item().frameRate

媒体项目对象属性：app.project.item.frameRate
描述：
媒体项目的帧速率，也就是每秒有多少帧。这个值是帧持续时间的倒数。当写入一个值时，这个值的倒数会自动设置为新的帧持续时间的值。

- 在合成项目中，这个值被链接到合成的帧速率，读/写
- 在素材项目中，这个值被链接到真实源素材对象的帧速率上，只读。设置真实源素材的 conformFrameRate 可以改变这个值。这个值同时设置了素材项目的帧速率和帧持续时间。

类型：浮点型，范围[1.0~99.0]，素材项目对象中只读，其他情况读/写

app.project.item().hasAudio

媒体项目对象属性：app.project.item(index).hasAudio
描述：
当值为 true 时，这个媒体项目含有音频。

- 在合成项目中，这个值被链接到合成对象
- 在素材项目中，这个值被链接到真实源对象

类型：布尔，只读

app.project.item().hasVideo

媒体项目对象属性：app.project.item(index).hasVideo

描述：

当值为 true 时，这个媒体项目含有视频。

- 在合成项目中，这个值被链接到合成对象
- 在素材项目中，这个值被链接到真实源对象

类型：布尔，只读

app.project.item().height

媒体项目对象属性：app.project.item(index).height

描述：

项目的像素高

- 在合成项目中，这个值被链接到合成对象，读/写
- 在素材项目中，这个值被链接到真实源对象，只有真实源对象是固态层时是读/写，其他情况只读

类型：整数，范围[1~30000],除了作为 noted 时，其他情况为读/写

app.project.item().name

媒体项目对象属性：app.project.item(index).name

描述：

项目的名称，与工程面板中显示的名称相同

- 在素材项目中，这个值被链接到真实源对象中。如果真实源对象是一个导入的素材，这个值只影响素材在工程面板中显示的名称，不会影响到文件真实的名称。

类型：字符串型，读/写

app.project.item().pixelAspect

媒体项目对象属性：app.project.item(index).pixelAspect

描述：

项目的像素宽高比(PAR)

- 在合成项目中，这个值被链接到合成
- 在素材项目中，这个值被链接到真实源对象

这个值与你设置的值可能会有微小的不一样。下面这个表格比较了 UI 中显示的值和属性返回值

UI 中显示	属性返回值
0.91	1.09401709401709

1	1
1.5	1.5
1.09	1.09401709401709
1.21	1.21212121212121
1.33	1.33333333333333
1.46	1.45868945868946
2	2

类型：浮点型，范围[0.01~100.0]，读/写

app.project.item().proxySource

媒体项目对象属性：app.project.item(index).proxySource
描述：这个素材源开始被用作代理。这个属性为只读，想要改变这个值，可以调用一些媒体项目方法来改变代理源：setProxy()、setProxyWithSequence()、setProxyWithSolid()或者setProxyWithPlaceholder()
类型：素材源对象，只读

app.project.item().setProxy()

媒体项目对象方法：app.project.item(index).setProxy(*file*)
描述：
给当前媒体项目设置一个代理文件。从一个新的文件源对象中导入指定文件，这个文件同时也作为proxySource 属性的值，并且 useProxy 为 true。当使用用户首选项改变值时，这个方法的参数的值也同时会改变。通常情况下如果文件含有一个未标记的 Alpha 通道，并且用户首选项会要求用户做出反应，但是使用这个方法会自动对 Alpha 做出解释，而不需要询问用户。
参数：

file	使用文件对象指定一个文件作为代理使用

返回：无

app.project.item().setProxyToNone()

媒体项目对象方法：app.project.item(index).setProxyToNone
描述：移除这个媒体项目的代理，将 proxuSource 设置为 null，并且将 useProxy 的值设置为

false。
参数：无
返回：无

app.project.item().setProxyWithPlaceholder()

媒体项目对象方法：
app.project.item(index).setProxyWithPlaceholder(*name,width,height,frame-Rate,duration*)
描述：
创建一个指定参数的占位符源，这个值同时也作为 proxySource 属性值，并且 useProxy 为 true。。当使用用户首选项改变值时，这个方法的参数的值也同时会改变。
注意：在 UI 中没有一个直接的操作方法可以让一个占位符作为一个代理。这种情况只会发生在一个代理文件被设置后，文件被移动或删除了。
参数：

name	新对象的名称，字符串型
width,height	占位符的像素宽高，整数型，范围[4~30000]
frameRate	一秒内帧的数量，整数型，范围[1~99]
duration	持续时长，最长为 3 小时。整数型，范围[0.0~10800.0]

返回：无
笔记：按理说 duration 的范围应该属于浮点型，但是原文写的就是整数型。

app.project.item().setProxyWithSequence()

媒体项目对象方法：app.project.item(index).setProxyWithSequence(*file,forceAlphabetical*)
描述：设置一组序列文件作为当前媒体项目的代理。在新文件源对象中导入一组指定的序列文件，这个值同时也作为 proxySource 属性值，并且 useProxy 为 true。当使用用户首选项改变值时，这个方法的参数的值也同时会改变。通常情况下如果序列文件含有一个未标记的 Alpha 通道，并且用户首选项会要求用户做出反应，但是使用这个方法会自动对 Alpha 做出解释，而不需要询问用户。
参数：

file	一组序列文件的第一个文件
forceAlphabetical	当值为 true 时，文件序列以字母顺序排序

返回：无

app.project.item().setProxyWithSolid()

媒体项目对象方法：
app.project.item(index).setProxyWithSolid(*color,name,width,height,pixel-Aspect*)
描述：创建一个指定各个参数的而固态层源对象作为当前项目的代理，这个值同时也作为 proxySource 属性值，并且 useProxy 为 true。。当使用用户首选项改变值时，这个方法的参数的值也同时会改变。
注意：在 UI 中没有一个直接的操作方法可以让一个固态层作为一个代理。这个功能只能使用脚本来实现。
参数：

color	固态层的颜色，三个浮点型值组成的数组，[R,G,B]范围[0.0~1.0]
name	新对象的名称，字符串型
width,height	固态层的像素宽高，整数型，范围[1~30000]
pixelAspect	固态层的像素宽高比，浮点型，范围[0.01~100.0]

返回：无

app.project.item().time

媒体项目对象属性：app.project.item(index).time
描述：
从工程面板点开进行预览的项目的当前时间。这个值是秒数。使用全局函数 timeToCurrentFormat 转换为字符串型的值来表示帧的时间。
mainSource 为静态的素材项目使用这个属性会报错（item.mainSource.isStill 为 true）
类型：浮点型，读/写

app.project.item().usedIn

媒体项目对象属性：app.project.item(index).usedIn
描述：
所有使用这个媒体项目的合成。
注意在检索值之后，这些值已经被储存在数组里了，所以不会自动更新。如果你取得这些值后，又把项目添加到其他合成中，你必须重新检索一次这些值得到包括新合成的数组。
类型：合成项目对象的数组，只读

app.project.item().useProxy

媒体项目对象属性：app.project.item(index).useProxy
描述：值为 true 时，当前项目使用了代理。所有的 SetProxy 方法都会使这个值为 true，通过 SetProxyToNone()方法可以使值变为 false
类型：布尔型，读/写

app.project.item().width

媒体项目对象属性：app.project.item(index).width
描述：
当前项目的像素宽度

- 在合成项目中，这个值被链接到合成。读/写
- 在素材项目中，这个值被链接到真实源对象中，并且当真实源对象为固态层源时，值为读/写，其他情况都为只读

类型：
整数型，范围[1~30000]，读/写，除非为上面标注的情况

媒体层对象

app.project.item(index).layer(*index*)

描述：

媒体层对象提供了一个接口给这些包含媒体项目对象的层（合成层、素材层、固态层、文字层和音频层）。

- 媒体层是层对象的子集。所有层对象的方法和属性和下面列出的这些都可在使用媒体层对象时调用。
- 媒体层是文字层的基础类，所以媒体层对象的所有属性和方法都可以在使用文字层对象时调用。

AE 属性：

不同的层有不同的属性。媒体层有下列这些属性和属性组

```
Marker
Time Remap
Motion Trackers
Masks
Effects
Transform
       Anchor Point
       Position
       Scale
       Orientation
       X Rotation
       Y Rotation
       Rotation
       Opacity
Layer Styles
Geometry Options          //光线追踪 3D
Material Options
       Casts Shadow
       Light Transmission
       Accepts Shadows
       Accepts Lights
       Appears in Reflections    //光线追踪 3D
       Ambient
       Diffuse
       Specular Intensity
       Specular Shininess
       Metal
       Reflection Intensity      //光线追踪 3D
```

Reflection Sharpness //光线追踪 3D

Reflection Rolloff //光线追踪 3D

Transparency //光线追踪 3D

Transparency Rolloff //光线追踪 3D

Index of Reflection //光线追踪 3D

Audio

Audio Levels

举例：

如果工程中第一个项目是合成项目，并且合成项目中第一个层是一个媒体层，那么下面是设置层质量、开始时间和入点的方式。

```
var firstLayer = app.project.item(1).layer(1);
firstLayer.quality = LayerQuality.BEST;
firstLayer.startTime = 1;
firstLayer.inPoint = 2;
```

属性：

属性	描述
source	这个层的源项目
isNameFromSource	值为 true 时，这个层没有被设置层名称，当前名称为源素材名称
height	层的高度
width	层的宽度
audioEnabled	值为 true 时，层的音频处于开启状态
motionBlur	值为 true 时，层的动态模糊已开启
effectsActive	值为 true 时，层的效果被激活
adjustmenLayer	值为 true 时，这是一个调节图层
guideLayer	值为 true 时，这是一个参考图层
threeDLayer	值为 true 时，这是一个 3D 图层
threeDPerChar	值为 true 时，开启文字图层的 3D 文字开关
environmentLayer	值为 true 时，这是一个环境图层
canSetCollapseTransformation	值为 true 时，改变 collapseTransformation 的值是合法的

collapseTransformation	值为 true 时，塌陷转换已开启
frameBlending	值为 true 时，帧混合已开启
frameBlendingType	层的帧混合类型
canSetTimeRemapEnabled	值为 true 时，改变 timeRemapEnabled 的值是合法的
timeRemapEnabled	值为 true 时，当前层已经开启了时间重映射
hasAudio	值为 true 时，当前层包含音频
audioActive	值为 true 时，当前时间层的音频已经激活
blendingMode	层的叠加模式
preserveTransparency	值为 true 时，保持透明已开启
trackMatteType	如果层有一个轨道蒙版，指定的蒙版方式已被应用
isTrackMatte	值为 true 时，当前层作为下层的轨道蒙版
hasTrackMatte	值为 true 时，上层被作为当前层的轨道蒙版
quality	设置的层质量
autoOrient	层自动定向的类型

方法：

方法	描述
audioActiveAtTime()	控制当前层的音频是否在给定的时间激活
calculateTransformFromPoints()	计算当前层中一个指定的点的变化
replaceSource()	改变当前层的源项目
sourceRecAtTime()	检索层的四个边缘
openInViewer()	在一个层预览面板中打开指定层

app.project.item().layer().adjustmentLayer

媒体层对象属性：app.project.item(index).layer(*index*).adjustmentLayer

描述：如果当前层为调节图层则返回 true

类型：布尔型，读/写
笔记：没错，这个属性不仅可以判断当前层是不是调节图层，因为能写入，还能把其他图层变为调节图层。

app.project.item().layer().audioActive

媒体层对象属性：app.project.item(index).layer(*index*).audioActive
描述：
如果当前时间层的音频被开启则返回 true。
如果这个属性的值为 true，那 audioEnable 一定为 true，没有其他的层含有音频可能是因为那些层正在单独显示(solo)，除非当前层也被单独显示着，激活的时间一定是在入点和出点之间。
类型：布尔型，只读

app.project.item().layer().audioActiveAtTime()

媒体层对象方法：app.project.item(index).layer(*index*).audioActiveAtTime(*time*)
描述：
如果当前层的音频在指定时间被开启则返回 true。
如果这个方法返回 true，那 audioEnabled 一定为 true，没有其他的层含有音频可能是因为哪些层正在单独显示(solo)，除非当前层也被单独显示着，激活的时间一定是在入点和出点之间。
类型：布尔型
参数：

time	激活的时间，秒数，浮点型

返回：布尔型

app.project.item().layer().audioEnabled

媒体层对象属性：app.project.item(index).layer(*index*).audioEnabled
描述：当值为 true 时，当前层的音频已开启。这个值对应时间线面板上的音频开关。
类型：布尔型，读/写

app.project.item().layer().autoOrient

媒体层对象属性：app.project.item(index).layer(*index*).autoOrient
描述：当前层执行的自动定向的类型。
类型：
自动定向值的列举，读/写

AutoOrientType.ALONG_PATH	层面向路径的方向
AutoOrientType.CAMERA_OR_POINT_OF_INTEREST	层面向激活的摄像机或摄像机的兴趣点
AutoOrientType.CHARACTERS_TOWARD_CAMERA	每个 3D 文字层的字节都自动面向激活的摄像机
AutoOrientType.NO_AUTO_ORIENT	层不会自动定向

app.project.item().layer().blendingMode

媒体层对象属性：app.project.item(index).layer(*index*).blendingMode

描述：层的叠加模式

类型：一些层叠加模式列举的值，读/写

BlendingMode.ADD
BlendingMode.ALPHA_ADD
BlendingMode.CLASSIC_COLOR_BURN
BlendingMode.CLASSIC_COLOR_DODGE
BlendingMode.CLASSIC_DIFFERENCE
BlendingMode.COLOR
BlendingMode.COLOR_BURN
BlendingMode.COLOR_DODGE
BlendingMode.DANCING_DISSOLVE
BlendingMode.DARKEN
BlendingMode.DARKER_COLOR
BlendingMode.DIFFERENCE
BlendingMode.DISSOLVE
BlendingMode.EXCLUSION
BlendingMode.HARD_LIGHT
BlendingMode.HARD_MIX
BlendingMode.HUE
BlendingMode.LIGHTEN
BlendingMode.LIGHTER_COLOR
BlendingMode.LINEAR_BURN
BlendingMode.LINEAR_DODGE
BlendingMode.LINEAR_LIGHT
BlendingMode.LUMINESCENT_PREMUL
BlendingMode.LUMINOSITY
BlendingMode.MULTIPLY
BlendingMode.NORMAL
BlendingMode.OVERLAY
BlendingMode.PIN_LIGHT
BlendingMode.SATURATION
BlendingMode.SCREEN
BlendingMode.SILHOUETE_ALPHA
BlendingMode.SILHOUETTE_LUMA
BlendingMode.SOFT_LIGHT

BlendingMode.STENCIL_ALPHA
BlendingMode.STENCIL_LUMA
BlendingMode.VIVID_LIGHT

app.project.item().layer().calculateTransformFromPoints()

媒体层对象方法：

app.project.item(index).layer(*index*).calculateTransformFromPoints(*pointTopLeft,pointTopRight,pointBottomRight*)

描述：计算与层中设置的点的变化

参数：

pointTopLeft	左上角的坐标，数组，[x,y,z]
pointTopRight	右上角的坐标，数组，[x,y,z]
pointBottomRight	右下角的坐标，数组，[x,y,z]

返回：变化属性设置的一个对象

举例：

```
var newLayer = comp.layers.add(newFootage);
newLayer.threeDLayer = true;
newLayer.blendingMode = BlendingMode.ALPHA_ADD;
var transform = newLayer.calculateTransformFromPoints(tl, tr, bl);
for(var sel in transform) {
    newLayer.transform[sel].setValue(transform[sel]);
    }
```

笔记：

原手册中写的内容就这么多，还是不知道这个方法到底是计算什么的，三个参数不能是同一值，返回的内容为[object Object]。

而且举例中第一行就会报错，都没有声明 tl、tr、bl 变量就直接用了，for 循环的参数也很奇怪，没有搜到这种用法的。

先放一下，以后用到再做研究。

app.project.item().layer().canSetCollapseTransformation

媒体层对象属性：app.project.item(1).layer(*index*).canSetCollapseTransformation

描述：如果当前层可以合法的改变 collapseTransformation 属性值则返回 true。

类型：布尔型，只读

笔记：*在 collapseTransformation 属性中会介绍到底什么是塌陷转换。*

app.project.item().layer().canSetTimeRemapEnabled

媒体层对象属性：app.project.item(index).layer(*index*).canSetTimeRemapEnabled
描述：如果当前层可以合法的改变 timeRemapEnbled 属性值则返回 true
类型：布尔型，只读

app.project.item().layer().collapseTransformation

媒体层对象属性：app.project.item(index).layer(*index*).collapseTransformation
描述：如果当前层开启了塌陷转换则返回 true。
类型：布尔型，读/写
笔记：
什么是 collapseTransformation（塌陷转换）？
其实就是时间线面板上八个功能开关里的小太阳，中文版 AE 中叫做卷展开关/连续删格化。
此开关也可以理解为还原素材原属性开关，只有在图层为另一个合成项目或.ai 格式的文件时才有用，应用后被嵌套的合成项目质量会提高，渲染时间也会减少，但部分特效和蒙版也将失去作用。
该开关将影响嵌套的合成图像和 AI 文件产生的层，当层是一个合成图像时，该开关为卷展变化开关，可以打开该开关改进图像质量并减少预览和渲染时间，该开关不能使用在应用了遮罩或效果的合成图像上，但是可以建立在调节图层上。当层是一个 AI 文件时，该开关为连续删格化，打开该开关后，AE 在预览和渲染时将会更改分辨率重新计算，因此不论尺寸如何，AE 仅以需要的分辨率显示图像。
有时合成会错位，但是点进去这个合成，里面内容明明是正常的，这时候试试将合成的“小太阳”关闭可能会修复。

app.project.item().layer().effectsActive

媒体层对象属性：app.project.item(index).layer(*index*).effectsActive
描述：如果当前层的效果被激活则返回 true，等同于在时间线面板中的<fx>按钮。
类型：布尔型，读/写

app.project.item().layer().environmentLayer

媒体层对象属性：app.project.item(index).layer(*index*).environmentLayer
描述：如果当前层为一个在光线追踪 3D 合成中的环境层则返回 true。设置这个属性将会自动把当前层转换为 3D 图层（threeDLayer 属性值将变为 true）。
类型：布尔型，读/写

笔记：*这个 environmentLayer（环境层）可以在层右键菜单栏中看到，只有在 3D 渲染器为光线追踪 3D 情况下才能用。至于怎么用，我从来没用过。*

app.project.item().layer().frameBlending

媒体层对象属性：app.project.item(index).layer(*index*).frameBlending
描述：如果当前层的帧混合开启则返回 true。
类型：布尔型，只读

app.project.item().layer().frameBlendingType

媒体层对象属性：app.project.item(index).layer(*index*).frameBlendingType
描述：当前层开启帧混合时的类型
类型：
一些帧混合类型列举的值，读/写
FrameBlendingType.FRAME_MIX
FrameBlendingType.NO_FRAME_BLEND
FrameBlendingType.PIXEL_MOTION

app.project.item().layer().guideLayer

媒体层对象属性：app.project.item(index).layer(*index*).guideLayer
描述：如果当前层为引导层则返回 true。
类型：布尔型，读/写

app.project.item().layer().hasAudio

媒体层对象属性：app.project,item(index).layer(*index*),hasAudio
描述：如果当前层含有音频则返回 true，与层是否开启音频或单独显示无关。
类型：布尔型，只读

app.project.item().layer().hasTrackMatte

媒体层对象属性：app.project.item(index).layer(*index*).hasTrackMatte
描述：如果当前层的上层为轨道蒙版则返回 true。当值为 true 时，当前层的 trackMatteType 属性值控制蒙版的类型。
类型：布尔型，只读

app.project.item().layer().height

媒体层对象属性：app.project.item(index).layer(*index*).height
描述：层的像素高
类型：浮点型，只读

app.project.item().layer().isNameFromSource

媒体层对象属性：app.project.item(index).layer(*index*).isNameFormSource
描述：
如果当前层没有设置层名称则返回 true，但是当前层有源名称。这种情况下，layer.name 与 layer.source.name 的值相同。
如果层设置了层名称，或者层没有源则返回 false。
类型：布尔型，只读

app.project.item().layer().isTrackMatte

媒体层对象属性：app.project.item(index).layer(*index*).isTrackMatte
描述：如果当前层被用作下层的轨道蒙版则返回 true。
类型：布尔型，只读

app.project.item().layer().motionBlur

媒体层对象属性：app.project.item(index).layer(*index*).motionBlur
描述：如果当前的运动模糊开启则返回 true。
类型：布尔型，读/写

app.project.item().layer().openInViewer()

媒体层对象方法：app.project.item(index).layer(*index*).openInViewer()
描述：使指定层在层预览面板打开，或者让层预览面板处于激活状态
参数：无
返回：当前层预览面板的预览器对象，如果不能打开指定层则返回 null。不能打开的情况可能是想要打开文字或形状图层，这两种图层不能在层预览窗口打开。

app.project.item().layer().preserveTransparency

媒体层对象属性：app.project.item(index).layer(*index*).preserveTransparency
描述：如果当前层的保持透明开启则返回 true。
类型：布尔型，读/写
笔记：*保持透明就是轨道蒙版前面的 T 按钮，开启这个按钮，当前图层只显示与下面层叠加的部分。*

app.project.item().layer().quality

媒体层对象属性：app.project.item(index).layer(*index*).quality
描述：当前层显示的质量
类型：
一些层质量列举的值，读/写
LayerQuality.BEST
LayerQuality.DEAFT
LayerQUality.WIREFRAME

app.project.item().layer().replaceSource()

媒体层对象方法：
app.project.item(index).layer(*index*).replaceSource(*newSource,fixExpress-ions*)
描述：给当前层替换源
参数：

newSource	新媒体项目对象源
fixExpressions	值为 true 时，修正新源的表达式，否则为 false。注意这个功能是资源密集型(resource-intensive)的，如果替换很多个素材，只需要在最后的操作执行这个功能。

返回：无

app.project.item().layer().source

媒体层对象属性：app.project.item(index).layer(*index*).source
描述：当前层的媒体项目源。如果当前层为文字层，那么值为 null。使用 AVLayer.replaceSource()可以改变这个值。
类型：媒体项目对象，只读

app.project.item().layer().sourceRectAtTime()

媒体层对象方法：app.project.item(index).layer(*index*).sourceRectAtTime(*timeT,extents*)
描述：在指定时间索引时检索指定层的矩形边界，修正文字图层或形状图层的内容。比如用来使文字与基础线对齐。
参数：

timeT	指定的时间索引，秒数，浮点型
extents	值为 true 时包括实际区域，否则为 false。这个值应用到形状图层上，在必要时可以增大层的边界。

返回：一个有四个属性的 javascript 对象，[top,left,width,height]

app.project.item().layer().threeDLayer

媒体层对象属性：app.project.item(index).layer(*index*).threeDLayer
描述：如果当前层为 3D 图层则返回 true。
类型：布尔型，读/写

app.project.item().layer().threeDPerChar

媒体层对象属性：app.project.item(index).layer(*index*).threeDPerChar
描述：如果当前图层开启了 3D 文字开关则返回 true，3D 文字开关允许字节在文字图层平面上做 3D 动画。只能用于文字图层。
类型：布尔型，读/写

app.project.item().layer().timeRemapEnabled

媒体层对象属性：app.project.item(index).layer(*index*).timeRemapEnabled
描述：如果当前层开启时间重映射则返回 true。
类型：布尔型，读/写

app.project.item().layer().trackMatteType

媒体层对象属性：app.project.item(index).layer(*index*).trackMatteType
描述：指定当前层轨道蒙版的应用方式
类型：
一些轨道蒙版列举的值，读/写
TrackMatteType.ALPHA

TrackMatteType.ALPHA_INVERTED
TrackMatteType.LUMA
TrackMatteType.LUMA_INVERTED
TrackMatteType.NO_TRACK_MATTE

app.project.item().layer().width

媒体层对象属性：app.project.item(index).layer(*index*).width

描述：当前层的像素宽

类型：浮点型，只读

摄像机层对象

app.project.item(index).layer(index)

描述：

摄像机层对象代表合成内的一个摄像机层。使用层收集对象的 addCamera()方法来创建一个摄像机层。它可以被项目的层通过索引号或者字符串名称来收集。

- 摄像机层对象是层对象的子类。所有层对象的方法和属性都可以被摄像机层对象调用。

AE 属性

摄像机层没有另外定义一些属性，但是它有和其他层不同的 AE 属性。下面是一些属性和属性组

Maker
Transform
    Point of Interest
    Position
    Scale
    Orientation
    X Rotation
    Y Rotation
    Rotation
    Opacity
Camera Options
    Zoom
    Depth of Field
    Focus Distance
    Blur Level

收集对象

收集对象把一组对象或值作为一个逻辑组，并且通过索引号提供一个接口，就像一个数组。然而，大多数收集对象是只读的。你不能手动给它们分配对象，它们的内容会根据对象的创建或删除自动更新。

收集对象索引号是从 1 开始的，不是 0。

对象

对象	描述
ItemCollection	所有可以在工程面板找到的项目（导入的外部文件、文件夹、合成等等）
LayerCollection	合成中所有的层
OMCollection	工程中所有的输出模块项目
RQItemCollection	工程中所有的渲染队列项目

属性

length	收集的对象数

方法

[]	通过索引号检索收集的对象，第一个对象的索引是 1

合成项目对象

app.project.item(*index*)
app.project.items[*index*]
描述：
合成项目对象代表着一个合成，允许你控制一个合成并且可以得到关于这个合成的信息。通过工程的项目收集对象来调用这个对象。

- 合成项目对象是媒体项目对象的子类，而媒体项目对象又是项目对象的子类。所有媒体项目对象和项目对象的方法和属性都可以被合成项目对象调用。

举例：
假定工程中第一个项目是一个合成项目，下面的代码显示两个警告框。第一个展示合成项目中的层数量，第二个展示合成项目中最后一个层的名字。

```
var firstComp = app.project.item(1);
alert("层的数量是 " + firstComp.numLayers);
alert("最后一个层的名字是 " + firstComp.layer(firstComp.numLayers).name);
```

笔记：*其实到现在这些对象都没办法直接确定想用的合成、图层之类的，不可能在实际使用中总是用索引号来调用合成图层，一是你不知道用户在工程面板中建立的项目顺序，二是用户随时都可以根据排序规则改变项目顺序。不过有个 app.project.activeItem() 对象来调用当前正在使用的项目，后面应该会有。*
属性：

属性	描述
frameDuration	帧的持续时长
dropFrame	值为 true 时，表示当前合成使用了丢帧时间码
workAreaStart	工作区的开始时间
workAreaDuration	工作区的持续时间
numLayers	合成内层数量
hideShyLayers	值为 true 时，隐藏图层会在时间线面板中可见
motionBlur	值为 true 时，动态模糊在合成中开启
draft3d	值为 true 时，草稿 3D 模式在合成面板中开启

frameBlending	值为 true 时，帧混合在合成中开启
preserveNestedFrameRate	值为 true 时，帧速率在嵌套合成中被保留
preserveNestedResolution	值为 true 时，分辨率在嵌套合成中被保留
bgColor	当前层的背景颜色
activeCamera	当前激活的摄像机层
displayStartTime	改变时间线面板中的开始时间
resolutionFactor	合成面板中 X 和 Y 分辨率被降低采样的因素
shutterAngle	摄像机快门角度
shutterPhase	摄像机快门相位
motionBlurSamplesPerFrame	在经典 3D 层、形状图层和一些效果中，每帧最小动态模糊采样数
motionBlurAdaptiveSampleLimit	2D 层运动最大的运动采样数
layers	当前合成的层
selectedLayers	当前合成选中的层
selectedProperties	当前合成选中的属性
renderer	合成中使用的渲染插件模块
renderers	一组激活的渲染插件模块

方法：

方法	描述
duplicate()	创建并返回当前层的一个复制
layer()	从当前合成调用一个层
openInViewer()	将合成在在合成面板中打开

app.project.item().activeCamera

合成项目对象属性：app.project.item(index).activeCamera

描述：激活的摄像机，就是在可用的摄像机层中最上方的那个。如果合成中不包含可用的摄像机层则返回 null
类型：摄像机层对象，只读

app.project.item().bgColor

合成项目对象属性：app.project.item(index).bgColor
描述：当前合成的背景色。用三个数组值来指定红绿蓝三个颜色通道。
类型：包含三个浮点型值的数组，[R,G,B]，范围[0.0~1.0]，读/写

app.project.item().displayStartTime

合成项目对象属性：app.project.item(index).displayStartTime
描述：设置的合成起始时间，秒数。等同于在合成设置对话框中设置初始帧或开始时间码。
类型：浮点型，范围[0.0~86339.0],读/写

app.project.item().draft3d

合成项目对象属性：app.project.item(index).draft3d
描述：值为 true 时，合成面板中已经开启草图 3D 模式。等同于在合成面板汇总草图 3D 的按钮。
类型：布尔型，读/写

app.project.item().dropFrame

合成项目对象属性：app.project.item(index).dropFrame
描述：值为 true 时，表示合成使用了丢帧时间码。值为 false 时，表示没有丢失时间码。等同于在合成设置对话框中的设置。
类型：布尔型，读/写

app.project.item().duplicate()

合成项目对象方法：app.project.item(index).duplicate()
描述：创建并返回一个合成的复制，这个复制的合成包含和源合成一样的图层。
参数：无
返回：合成项目对象

app.project.item().frameBlending

合成项目对象属性：app.project.item(index).frameBlending
描述：值为 true 时，合成开启了帧混合。等同于合成面板中帧混合按钮。
类型：布尔型，如果值为 true，帧混合已开启，读/写

app.project.item().frameDuration

合成项目对象属性：app.project.item(index).frameDuraion
描述：帧的持续时长，秒数。帧速率的倒数。
类型：浮点型，读/写
笔记：*这个属性在媒体项目对象中也讲到了。*

app.project.item().hideShyLayers

合成项目对象属性：app.project.item(index).hideShyLayers
描述：值为 true 时，只有没有设置隐藏的层在时间线面板显示。值为 false 时，所有层都可见，包括那些设置为隐藏的值（因为还有个隐藏的总开关）。等同于合成面板上隐藏所有层的按钮。
类型：布尔型，读/写

app.project.item().layer()

合成项目对象方法：
app.project.item(index).layer(index)
app.project.item(index).layer(otherLayer,relIndex)
app.project.item(index).layer(name)
描述：返回一个指定名字或索引位置或者相关其他层的索引位置的层对象。
参数：

index	合成中想要调用的层索引号。整数型，范围[1~层数]

-或-

otherLayer	合成中一个层对象。relIndex 值是这个层的索引号加上 relIndex 来定位想要调用的层
relIndex	想要调用的层的位置，与 otherLayer 相关。整数型，范围[1-otherLayer.index,index~numLayers-otherLayer.index]，*numLayers* 为合成中层的数量

-或-

name	想调用的层的名称，字符串型

返回：层对象

app.project.item().layers

合成项目对象属性：app.project.item(index).layers
描述：包含所有当前合成内的层对象的层收集对象。
类型：层收集对象，只读

app.project.item().motionBlur

合成项目对象属性：app.project.item(index).motionBlur
描述：值为 true 时，合成开启了动态模糊。等同于合成面板中的动态模糊按钮。
类型：布尔型，读/写

app.project.item().motionBlurAdaptiveSampleLimit

合成项目对象属性：app.project.item(idnex).motionBlurAdaptiveSampleLimit
描述：2D 层运动最大的运动模糊采样。等同于合成设置窗口中高级标签的自适应采样限制。
类型：整数，范围[16~256]，读/写

app.project.item().motionBlurSamplesPerFrame

合成项目对象属性：app.project.item(index).motionBlurSamplesPerFrame
描述：经典 3D 层、形状图层和一些效果的最小运动模糊采样，等同于合成设置窗口中高级标签的每帧采样设置。
类型：整数，范围[2~64]，读/写

app.project.item().numLayers

合成项目对象属性：app.project.item(index).numLayers
描述：合成内图层的数量。
类型：整数，只读

app.project.item().openInViewer()

合成项目对象方法：app.project.item().openInViewer()
描述：在合成面板中打开一个合成，并且激活合成面板。

参数：无
返回：合成面板中的查看窗口对象，如果没有合成被打开则返回 null。

app.project.item().preserveNestedFrameRate

合成项目对象属性：app.project.item(index).preserveNestedFrameRate
描述：值为 true 时，当前合成中的预合成帧速率将被保留。等同于合成设置窗口中高级标签内的“预合成或渲染队列中保留帧速率”选项。
类型：布尔型，读/写

app.project.item().preserveNestedResolution

合成项目对象属性：app.project.item(index).preserveNestedResolution
描述：值为 true 时，当前合成中的预合成分辨率将被保留。等同于合成设置窗口中高级标签内的“预合成时保留分辨率”选项。
类型：布尔型，读/写

app.project.item().renderer

合成项目对象属性：app.project.item(index).renderer
描述：当前渲染插件模块正在被当前合成用作渲染，等同于合成设置窗口中高级标签的设置。
类型：字符串型的数组，只读

app.project.item().resolutionFactor

合成项目对象属性：app.project.item(index).resolutionFactor
描述：
渲染当前合成时 X 和 Y 降低分辨率采样因素。
数组的两个值指定采样时跳过多少像素。第一个数控制横向采样，第二个数值控制竖向采样。全分辨率为[1,1]，二分之一分辨率为[2,2]，四分之一分辨率为[4,4]。默认为[1,1]。
类型：两个整数型的数组，范围[1~99],读/写

app.project.item().selectedLayers

合成项目对象属性：app.project.item().selectedLayers
描述：当前合成内所有被选择的图层。这是一个 0 基础数组，即第一个对象的索引号是 0。
类型：层对象的数组，只读

app.project.item().selectedProperties

合成项目对象属性：app.project.item(index).selectedProperties
描述：所有当前合成内被选择的属性（属性和属性组）。第一个属性索引号是 0。
类型：属性和属性组对象的数组，只读

app.project.item().shutterAngle

合成项目对象属性：app.project.item(index).shutterAngle
描述：合成内设置的快门角度。等同于在合成设置窗口中高级标签内的快门角度设置。
类型：整数型，范围[0~720]，读/写

app.project.item().shutterPhase

合成项目对象属性：app.project.item(index).shutterPhase
描述：合成内设置的快门相位。等同于在合成设置窗口中高级标签内的快门相位设置。
类型：整数型，范围[-360~360]，读/写

app.project.item().workAreaDuration

合成项目对象属性：app.project.item(index).workAreaDuration
描述：工作区时长的秒数，这个属性与合成工作区的起始点和结束点时间不同。
类型：浮点型，读/写

app.project.item().workAreaStart

合成项目对象属性：app.project.item(index).wordAreaStart
描述：合成工作区开始的时间，秒数。
类型：浮点型，读/写

文件源对象

app.project.item(*index*).mainSource
app.project.item(*index*).proxySource
描述：
文件源对象描述了素材的来源文件。

- 文件源对象是素材源对象的子类。所有素材源的方法和属性以及下面的这些都可以被文件源对象调用。

属性：

属性	描述
file	定义这个属性的文件
missingFootagePath	丢失的文件路径

方法：

方法	描述
reload()	如果这个文件是素材的真实源，重新导入这个文件。

app.project.item().mainSource.file

文件源对象属性：
app.project.item(*index*).mainSource.file
app.project.item(*index*).proxySource.file
描述：
一个定义属性的 ExtendScript 文件对象中的文件，可以用下面方法改变这个值

- 如果文件源为媒体项目的代理源，调用 setProxy()或者 setProxyWithSequence()
- 如果文件源为素材项目的真实源，调用 replace()或者 replaceWithSwquence()

类型：文件对象，只读

app.project.item().mainSource.file.missingFootagePath

文件源对象属性：
app.project.item(*index*).mainSource.file.missingFootagePath
app.project.item(*index*).proxySource.file.missingFootagePath
描述：丢失文件的路径和文件名。

类型：字符串型，只读

app.project.item().mainSource.file.mainSource.reload()

文件源对象方法：app.project.item(*index*).mainSource.file.mainSource.reload()
描述：重新导入文件。这个只能被真实源文件调用，不可以被代理源调用。
参数：无
返回：无

文件夹项目对象

app.project.FolderItem

描述：文件夹对象等同于你的工程面板中的文件夹。文件夹可以包含各种类型项目（素材、合成、固态层等等），也可以是其他文件夹。

举例：

假定工程中第二个项目是文件夹项目，下面的代码弹出一个警告框显示文件夹内每个一级项目，并且显示项目的名称。

```
var secondItem = app.project.item(2);
if ( !(secondItem instanceof FolderItem) ) {
    alert("错误：第二个项目不是文件夹");
} else {
    for (i = 1; i <= secondItem.numItems; i++) {
        alert("项目顺序" + i + "项目名称 "+ secondItem.item(i).name);
    }
}
```

属性：

属性	描述
items	当前文件夹内包含的内容
numItems	当前文件夹内包含的内容数量

方法：

方法	描述
item()	从文件夹中取得一个项目

app.project.item().item()

文件夹项目对象方法：app.project.item(index).item(*index*)

描述：返回在这个文件夹中指定索引位置的一级项目。注意“一级项目”在这里的意思是在文件夹内第一层项目，不一定非要在工程内部。

参数：

index	整数型，要检索的项目的索引位置。第一个项目索引号是 1

返回：项目对象

app.project.item().items

文件夹对象属性：app.project.item(*index*).items
描述：
一个项目收集对象，包含文件夹内一级的项目对象。
和在工程对象中的项目收集对象不同，这个收集只包含这个文件夹内的一级项目。文件夹内的一级项目与工程内的一级项目不同。只有在根文件夹为一级项目的才同时也是工程内的一级项目。
类型：项目收集对象，只读

app.project.item().numItems

文件夹项目对象属性：app.project.item(*index*).numItems
描述：
在项目收集对象中包含的项目数量（folderItem.items.length）。
如果这个文件夹包含包含其他文件夹，只被当做一个项目计算，不管那个文件夹内包含多少个子项目。
类型：整数型，只读

素材项目对象

app.project.item(*index*)
app.project.items[*index*]
描述：
素材项目对象代表着一个导入进工程的素材项目。它可以通过位置索引号来调用工程中的项目收集对象的内容。

- 素材项目对象是媒体项目对象的子类，而媒体项目对象又是项目对象的子类。所有项目对象和媒体项目对象的方法和属性，加上下面列举这些，都可以被素材项目对象调用。

属性：

属性	描述
file	素材源文件
mainSource	素材项目所有的相关设置

方法：

方法	描述
replace()	用其他的素材文件代替当前素材文件
replaceWithPlaceholder()	用占位符对象代替当前素材文件
replaceWithSequence()	用图像序列代替当前素材文件
replaceWithSolid()	用固态层代替当前素材文件
openInViwer()	将当前素材在素材面板中打开

app.project.item().file

素材项目对象属性：app.project.item(*index*).file
描述：
素材源文件的 ExtendScript 文件对象。
如果素材项目的真实源是一个文件源，而不是代理源，那么这个属性值与 FootageItem.mainSource.file 的值相同，否则为 null。
类型：文件对象，只读

app.project.item().mainSource

素材项目对象属性：app.project.item(index).mainSource
描述：
素材源是指一个包含当前素材项目所有相关设置的对象，包括解释素材对话框的接口。这个属性值是只读的，想要改变这个值，调用一个素材项目的“replace”方法。
查看素材源对象，它有三个类型：

- 固态层源对象
- 文件源对象
- 占位符源对象

如果这是一个文件源对象，并且 footageMissing 值为 true，那么丢失的素材文件路径储存在 FileSource.missingFootagePath 属性中。
类型：素材源对象，只读

app.project.item().openInViewer()

素材项目对象方法：app.project.item(index).openInViewer()
描述：
在素材面板中打开这个素材，并且激活素材面板。
注意，丢失的素材和占位符素材也可以使用这个方法，但是不能使用 UI 界面打开。
参数：无
返回：素材面板的查看对象，如果素材不能被打开则返回 null。

app.project.item().replace()

素材项目对象方法：app,project.item(index).replace(*file*)
描述：
使用指定的文件来代替当前素材项目的源，除了加载文件，这个方法还会为加载文件创建一个新的文件源对象并且将加载文件设置为真实源。在新的源对象中，基于文件的内容设置了名称、宽度、高度、帧持续时间和时长属性。
这个方法保留了前一个真实源对象的解释参数。如果加载文件有一个未标识的 Alpha 通道，这个方法会自动评估 Alpha 解释。
参数：

file	作为素材真实源的文件

app.project.item().replaceWithPlaceholder()

素材项目对象方法：
app.project.item(index).replaceWithPlaceholder(*name,width,height,frame-Rate,duration*)
描述：使用指定的占位符代替素材项目的源。创建一个新的占位符源对象，从参数中设置对象的值，并且将真实源设置为新对象。
参数：

name	占位符的名称，字符串型
width	占位符的像素宽，整数型，范围[4~30000]
height	占位符的像素高，整数型，范围[4~30000]
frameRate	占位符的帧速率，浮点型，范围[1.0~99.0]
duration	占位符的持续时长，秒数，浮点型，范围[0.0~10800.0]

app.project.item().replaceWithSequence()

素材项目对象方法：app.project.item(index).replaceWithSequence(*file,forceAlphabetical*)
描述：
使用指定图像序列代替素材项目的源。除了加载文件，这个方法还会创建一个新的文件源对象,并且将加载文件作为对象的真实源。在新源对象中，基于加载文件内容会设置名称、宽度、高度、帧持续时间、时长属性。
这个方法保留了前一个真实源对象的解释参数。如果加载文件有一个未标识的 Alpha 通道，这个方法会自动评估 Alpha 解释。
参数：

file	作为真实源文件的第一个图片序列
forceAlphabetical	值为 true 时，使用“强制按字母排序”选项

app.project.item().replaceWithSolid()

素材项目对象方法：
app.project.item(index).replaceWithSolid(*color,name,width,height,pixel-Aspect*)
描述：使用指定固态层代替素材项目的源。创建一个新固态层源对象，使用参数中的值作为固态层的值，并且将固态层作为对象的真实源。
参数：

color	固态层的颜色，三个浮点型值的数组，[R,G,B]，范围[0.0~1.0]
name	固态层的名称，字符串型
width	固态层的像素宽，整数型，范围[4~30000]
height	固态层的像素高，整数型，范围[4~30000]
pixelAspect	固态层的像素长宽比，浮点型，范围[0.01~100.0]

素材源对象

app.project.item(index).mainSource
app.project.item(index).proxySource
描述：
素材源对象含有描述一些素材真实源的信息。这些素材源对象通常被用作素材项目的真实源或者合成项目和素材项目的代理源。

- 素材源是固态层源的基础类。所以素材源属性和方法在固态层源对象中仍然可以被调用。

属性：

属性	描述
hasAlpha	值为 true 时，素材或代理含有一个 Alpha 通道
alphaMode	Alpha 通道模式
premulColor	颜色被预乘
invertAlpha	值为 true 时，素材或代理中的 Alpha 通道被反转
isStill	值为 true 时，素材是一个静态图片
fieldSeparationType	场分离类型
highQualityFieldSeparation	场是怎样在非静止素材中被分离的
removePulldown	移除下拉场
loop	图像序列被设置的循环次数
nativeFrameRate	素材的原生帧速率
displayFrameRate	合成中显示的或渲染的有效帧速率
conformFrameRate	素材应该符合的帧速率

方法：

方法	描述
guessAlphaMode()	估算 Alpha 模式的设置
guessPulldown()	估算下拉场类型的设置

app.project.item().mainSource.alphaMode

素材源对象属性：
app.project.item(index).mainSource.alphaMode
app.project.item(index).proxySource.alphaMode
描述：
素材源对象的 alphaMode 属性定义了 alpha 信息在素材中被怎样解释。如果 hasAlpha 为 false，那么这个属性值将会没有意义。
类型：
一些 AlphaMode 列举值，读/写
AlphaMode.IGNORE
AlphaMode.STRAIGHT
AlphaMode.PREMULTIPLIED

app.project.item().mainSource.conformFrameRate

素材源对象属性：
app.project.item(index).mainSource.conformFrameRate
app.project.item(index).proxySource.conformFrameRate
描述：
被代替素材源帧速率的值，如果设置为 0，则使用源帧速率。
如果素材源的 isStill 为 true 则会报错。如果 removePulldown 属性值不是 PulldownPhase.OFF 而属性值设置为 0 则会报错。如果属性值设置为 0 并且 removePulldown 值不为 PulldownPhase.OFF，则自动设置 nativeFrameRate 值。
类型：浮点型，范围[0.0~99.0]，读/写

app.project.item().mainSource.displayFrameRate

素材源对象属性：
app.project.item(index).mainSource.displayFrameRate
app.project.item(index).proxySource.displayFrameRate
描述：
使用或渲染时素材的帧速率。
如果 removePulldown 值为 PulldownPhase.OFF，那么这个属性值与 conformFrameRate（如果不为 0）或 nativeFrameRate（如果 conformFrameRate 值不为 0）相同。如果 removePulldown 属性值不为 PulldownPhase.OFF，这个属性值为 conformFrameRate*0.8 则实际帧速率会每五帧丢一帧。
类型：浮点型，范围[0.0~99.0]，只读

app.project.item().mianSource.fieldSeparationType

素材源对象属性：
app.project.item(index).mainSource.fieldSeparationType
app.project.item(index).proxySource.fieldSeparationType
描述：
场是怎么在非静止素材中被分离的。
如果 isStill 属性值为 true，那么设置这个属性就会报错。如果 removePulldown 属性值不为 PulldownPhase.OFF 那么设置属性值为 FieldSeparationType.OFF 也会报错。
类型：
一些 FieldSeparationType 的列举值，读/写
FieldSeparationType.OFF
FieldSeparationType.UPPER_FIELD_FIRST FieldSeparationType.LOWER_FIELD_FIRST

app.project.item().mianSource.guessAlphaMode()

素材源对象方法：
app.project.item(index).mianSource.guessAlphaMode()
app.project.item(index).proxySource.guessAlphaMode()
描述：
将当前素材源设置 Alpha 模式、premulColor 和反转 Alpha 为最佳估计值。如果 hasAlpha 属性值为 false，那么就不会有变化。
参数：无
返回：无

app.project.item().mainSource.guessPulldown()

素材源对象方法：
app.project.item(index).mainSource.guessPulldown(*method*)
app.project.item(index).proxySource.guessPulldown(*method*)
描述：
为当前素材源设置 fieldSeparationType 和 removePulldown 为最佳评估值。如果 isStill 为 true，那么不会有改变。
参数：

method	这个方法用来做评估值。一个下拉场方法的评估值。 PulldownMethod.PULLDOWN_3_2 PulldownMethod.ADVANCE_24P

返回：无

app.project.item().mainSource.hasAlpha

素材源对象属性：
app.project.item(index).mainSource.hasAlpha
app.project.item(index).proxySource.hasAlpha
描述：
值为 true 时，素材含有 Alpha 通道。在这个情况下，alphaMode、invertAlpha 和 premulColor 属性值有效。值为 false 时，素材的这些属性值没有意义。
类型：布尔型，只读

app.project.item().mainSource.highQualityFieldSeparation

素材源对象属性：
app.project.item(index).mainSource,highQualityFieldSeparation
app.project.item(index).proxySource,highQualityFieldSeparation
描述：
值为 true 时，AE 使用指定的算法去指定怎样执行高质量场分离。
如果 isStill 属性值为 true 或者 fieldSeparationType 属性值为 FieldSeparation.OFF 那么就会报错。
类型：布尔型，读/写

app.project.item().mainSource.isStill

素材源对象属性：
app.project.item(index).mainSource.isStill
app.project.item(index).proxySource.isStill
描述：
素材为静态时值为 true，如果非静态则为 false。
静态素材可能是 JPEG 文件、固态层或 0 时长的占位符。非静态素材可能是视频文件、音频文件、序列或非 0 时长的占位符。
类型：布尔型，只读

app.project.item().mainSource.loop

素材源对象属性：
app.project.item(index).mainSource.loop
app.project.item(index).proxySource.loop
描述：
当在合成中使用素材循环播放的次数。

如果为静态文件则会报错。

类型：整数型，范围[1~9999],默认值为 1，读/写

app.project.item().mainSource.nativeFrameRate

素材源对象属性：

app.project.item(index).mainSource.nativeFrameRate
app.project.item(index).proxySource.nativeFrameRate

描述：

素材的源帧速率。

类型：浮点型，读/写

app.project.item().mainSource.premulColor

素材源对象属性：

app.project.item(index).mainSource.premulColor
app.project.item(index).proxySource.premulColor

描述：

颜色被预乘。这个属性值只有在 alphaMode 为 alphaMode.PREDULTIPLIED 时有效。

类型：三个浮点型的数组，[R，G，B]，范围[0.0~1.0]，读/写

app.project.item().mainSource.isStill

素材源对象属性：

app.project.item(index).mainSource.isStill
app.project.item(index).proxySource.isStill

描述：

在使用场分离中下拉场移除的类型。

如果素材为静态则会报错。当 fieldSeparation 属性值不为 FieldSepatationType.OFF 时尝试设置属性值为 0 会报错。

类型：

一些 PulldownPhase 列举值，读/写

PulldownPhase.RemovePulldown.OFF
PulldownPhase.RemovePulldown.WSSWW
PulldownPhase.RemovePulldown.SSWWW
PulldownPhase.RemovePulldown.SWWWS
PulldownPhase.RemovePulldown.WWWSS
PulldownPhase.RemovePulldown.WWSSW PulldownPhase.RemovePulldown.WSSWW_24P_ADVANCE
PulldownPhase.RemovePulldown.SSWWW_24P_ADVANCE
PulldownPhase.RemovePulldown.SWWWS_24P_ADVANCE

PulldownPhase.RemovePulldown.WWWSS_24P_ADVANCE
PulldownPhase.RemovePulldown.WWSSW_24P_ADVANCE

导入选项对象

new ImportOptions();
new ImportOption(*file*);
描述：
导入选项对象封装了使用 Project.importFile 方法导入文件的选项。
这个构造函数带有可选参数，一个文件的 ExtendScript 文件对象。如果不支持这个可选参数，你必须在使用对象的 imoortFile 方法前明确设置文件属性值，比如：

```
new ImportOptions().file = new File("myfile.psd");
```

属性：

属性	描述
importAs	要被导入的文件类型
sequence	值为 true 时，导入为序列文件而不是单独文件
forceAlphabetical	值为 true 时，“强制字母顺序”设置开启
file	被导入的文件或者被导入的序列的第一个文件

方法：

方法	描述
canImportAs()	限定导入的文件类型

importOptions.canImportAs()

导入选项对象方法：*importOptions*.canImportAs(*type*)
描述：指定文件是否是指定对象类型。如果这个方法返回 true，你可以设置得到的类型作为 importAs 属性值。
参数：

type	可以被导入的文件类型。一些 ImportAsType 列举值。 ImportAsType.COMP ImportAsType.FOOTAGE　　ImportAsType.COMP_CROPPED_LAYERS ImportAsType.PROJECT

返回：布尔型
举例：

```
var io = new ImportOptions(File("c:\\myFile.psd"));
if io.canImportAs(ImportAsType.COMP);
    io.importAs = ImportAsType.COMP;
```

importOptions.file

导入选项对象属性：*importOptions*.file
描述：被导入的文件。如果这个构造函数内设置了一个文件，你可以通过这个属性来访问文件。
类型：ExtendScript 文件对象，读/写

importOptions.forceAlphabetical

导入选项对象属性：*importOptions*.forceAlphabetical
描述：值为 true 时，等同于导入文件时在对话框中勾选“强制字母顺序”选项。
类型：布尔型，读/写

importOptions.importAs

导入选项对象属性：*importOptions*.importAs
描述：导入文件的对象类型。在设置前使用 canImportAs()方法检测导入的文件的对象类型。
类型：
一些 ImportAsType 列举值，读/写
ImportAsType.COMP_CROPPED_LAYERS
ImportAsType.FOOTAGE
ImportAsType.COMP
ImportAsType.PROJECT

importOptions.sequence

导入选项对象属性：*importOptions*.sequence
描述：值为 true 时，导入为一个序列，否则导入单个文件。
类型：布尔型，读/写

项目对象

app.project.item(index)
app.project.item[index]

描述：

项目对象代表一个在工程面板中显示的项目。

第一个项目的索引号是 1。

- 项目对象是媒体项目和文件夹项目的基础类，也是其他各种类型项目的基础类，所有的项目对象属性和方法都可以被这些项目类型调用。

属性：

属性	描述
name	在工程面板中显示的项目对象名称
comment	用作注释的字符串
id	项目的唯一标识符
parentFolder	项目的父文件夹
selected	值为 true 时，当前项目正在被选中
typeName	项目的类型
label	项目的标签颜色

方法：

方法	描述
remove()	从工程中删除指定项目

举例：

这个例子是用作从工程中获取第二个项目并且检查是否是文件夹，然后从文件夹中删除所有没被选中的一级项目，同时检查文件夹中每个项目的父级设置为正确的文件夹。

```
var myFolder = app.project.item(2);
if (myFolder.typeName != "Folder") {
  alert("错误：第二个项目不是文件夹");
} else {
  var numInFolder = myFolder.numItems;
  // 有项目被删除后循环
  for (i = numInFolder; i >= 1; i--) {
    var curItem = myFolder.item(i);
    if (curItem.parentFolder != myFolder) {
      alert("AE 内部错误：父文件夹没有被正确设置");
    } else {
      if (!curItem.selected && curItem.typeName == "Footage") {
        //发现一个未选择的固态层
        curItem.remove();
      }
    }
  }
}
```

笔记：

变量 curItem 为 app.project.item().item()。这个连着两个 item()的情况只有在第一个 item 为文件夹时才可以用，这也是为什么一开始会验证一下第二个项目是不是文件夹。

第二个警告框出现的原因是第二个 item()的父级不是第一个 item()，但是按着语法来说是从第一个 item()中调用的第二个 item()，怎么可能会出现这种错误呢？

app.project.item().comment

项目对象属性：app.project.item(*index*).comment

描述：保存一段内容的字符串，经过任意编码转换后最多可以有 15999B，大约 16KB。这段内容只是为了给用户查看，不影响项目的显示或者行为。

类型：字符串型，读/写

笔记：*这段内容在工程面板的项目的标签中能找到，将工程面板拉长就能看见了，属于项目的标签中的一个。*

app.project.item().id

项目对象属性：app.project.item(index).id

描述：在内部中唯一且持续的标识符，用来在不同会话中标识项目。ID 的值在文件保存或稍后重新打开时都被保留。但是当你将当前工程导入到新的工程后，所有被导入的工程项目都会被重新分配一个新的 ID。这个 ID 不会在 UI 界面中显示。

类型：整数型，只读

app.project.item().label

项目对象属性：app.project.item(index).label
描述：
项目的标签颜色。用数字来代表颜色（0 代表无，1 到 16 分别代表一种颜色）。
脚本不能设置自定义颜色的标签。
类型：整数，范围[0~16]，读/写

app.project.item().name

项目对象属性：app.project.item(index).name
描述：项目在工程面板中显示的名称。
类型：字符串型，读/写

app.project.item().parentFolder

项目对象属性：app.project.item(index).parentFolder
描述：包含这个项目的文件夹（对象）。如果这个项目在工程中的第一层，则返回工程的跟文件夹(app.project.rootFolder)。你可以使用项目收集对象的 addFolder 方法来创建一个新的文件夹，并且设置这个属性值为新文件夹。
类型：文件夹项目对象，读/写
举例：
这段代码在工程面板中创建了一个新的文件夹项目并且将合成移动其内。

```
// 在工程中创建一个新的文件夹项目，名称为“comps”
var compFolder = app.project.items.addFolder("comps");
// 将所有合成移动到新的文件夹内
// folder 合成项目的父文件夹为“comps”文件夹
for (var i = 1; i <= app.project.numItems; i++) {
  if (app.project.item(i) instanceof CompItem)
    app.project.item(i).parentFolder = compFolder;
}
```

app.project.item().remove()

项目对象方法：app.project.item(index).remove()
描述：从项目和项目面板中删除指定项目。如果是文件夹项目，文件夹内包含的所有项目都会被删除。不会在电脑硬盘中删除文件或文件夹。
参数：无
返回：无

app.project.item().selected

项目对象属性：app.project.item(index).selected
描述：值为 true 时，当前项目被选中。可以同时选中多个项目。设置值为 true 来自动选中项目，或设置为 false 来取消选中。
类型：布尔型，读/写

app.project.item().typeName

项目对象属性：app.project.item(index).typeName
描述：用户可读的项目类型，比如“文件夹”、“素材”、“合成”等等。
类型：字符串型，只读

项目收集对象

app.project.items

描述：

项目收集对象代表了项目的收集，项目收集对象属于包含所有在工程中项目的项目对象的工程对象，同时也属于包含所有在文件夹中项目的项目对象的文件夹项目对象。

- 项目收集对象是收集对象的子类。所有手机对象的方法和属性都可以被项目收集对象调用。

方法：

方法	描述
addComp()	创建一个新的合成项目对象并且添加到收集对象中
addFolder()	创建一个新的文件夹项目对象并且添加到收集对象中

笔记：

*这个收集对象可以理解为 **AE** 内部用来存放和记录所有项目的一个对象，不在 **UI** 界面上显示，但是的确存在。*

*至于上面一段绕口的项目的项目对象什么的，原文只是为了说的严谨一些，可以单纯的理解为项目，实际上是指代表项目的对象才能被脚本和 **AE** 操作。*

app.project.items.addComp()

项目收集对象方法：

app.porject.items.addComp(*name,width,height,pixelAspect,duration,frame-Rate*)

描述：

创建一个新的合成。创建并返回一个新的合成项目对象并且添加到收集对象中。

如果项目收集对象属于工程或根文件夹，那么新项目的 parentFolder 属性值为根文件夹。如果项目收集对象属于任意一个其他文件夹，那么新项目的 parentFolder 为那个文件夹。

参数：

name	包含合成名称的字符串
width	合成的像素宽，整数型，范围[4~30000]
height	合成的像素高，整数型，范围[4~30000]
pixelAspect	合成的像素长宽比，浮点型，范围[0.01~10800.0]

duration	合成持续时长，秒数，浮点型，范围[0.0~10800.0]
frameRate	合成的帧速率，浮点型，范围[1.0~99.0]

返回：合成项目对象

app.project.items.addFolder()

项目收集对象方法：app.project.items.addFolder(*name*)

描述：

创建一个新的文件夹。创建并返回一个新的文件夹项目对象并且添加到收集对象中。

如果项目收集对象属于工程或根文件夹，那么新文件夹的 parentFolder 属性值为根文件夹。如果项目收集对象属于任意一个其他文件夹，那么新文件夹的 parentFolder 为那个文件夹。

参数：

name	包含文件夹名称的一段字符

返回：文件夹项目对象

举例：

这段代码在工程面板中创建一个新的文件夹项目并且将合成移动到其内。

```
// 在工程中创建一个新的文件夹项目，名称为“comps”
var compFolder = app.project.items.addFolder("comps");
// 将所有合成移动到新的文件夹内
// folder 合成项目的父文件夹为“comps”文件夹
for (var i = 1; i <= app.project.numItems; i++) {
  if (app.project.item(i) instanceof CompItem)
    app.project.item(i).parentFolder = compFolder;
}
```

关键帧缓动对象

myKey = new KeyframeEase(*speed, influence*)

描述：

关键帧缓动对象封装了层的关键帧缓动设置。关键帧缓动取决于使用属性的 setTemporalEaseAtKey 方法设置的速度和影响值。

这个构造函数创建一个关键帧缓动对象，所有参数都是必须的。

- speed：浮点型，设置速度属性
- influence：浮点型，范围[0.1~100.0]，设置影响属性

举例：

这段代码假定位置属性有超过两个关键帧。

```
var easeIn = new KeyframeEase(0.5, 50);
var easeOut = new KeyframeEase(0.75, 85);
var myPositionProperty = app.project.item(1).layer(1).property("Position")
myPositionProperty.setTemporalEaseAtKey(2, [easeIn], [easeOut]);
```

这段代码设置了缩放，具有两个或三个维度的时间属性。对于 2D 和 3D 属性必须给每个维度设置 easeIn 和 easeOn 值。

```
var easeIn = new KeyframeEase(0.5, 50);
var easeOut = new KeyframeEase(0.75, 85);
var myScaleProperty = app.project.item(1).layer(1).property("Scale")
myScaleProperty.setTemporalEaseAtKey(2, [easeIn, easeIn, easeIn], [easeOut, easeOut,
easeOut]);
```

属性：

属性	描述
speed	设置关键帧的速度值
influnce	设置关键帧的影响值

mykey.influence

关键帧缓动对象属性：*mykey*.influence

描述：关键帧的影响值，等同于关键帧速度对话框中所示。

类型：浮点型，范围[0.1~100.0]，读/写

mykey.speed

关键帧缓动对象属性：mykey.speed
描述：关键帧的速度值。这个属性取决于关键帧的类型，等同于关键帧速度对话框所示。
类型：浮点型，读/写

层对象

app.project.item(index).layer(index)

描述：

层对象提供了在合成内层的接口，也可以通道项目的层收集对象的索引或名称来调用。

- 层对象是摄像机层对象、灯光层对象和媒体层对象的基础类，所以层对象的属性和方法也可以被其他所有类型的层对象调用。

层对象包含了 AE 属性，除了它的 javascript 属性和方法。后面会有一章“属性基础对象”来讲解怎么调用层中的属性。

举例：

如果工程中第一个项目是合成项目，这段代码让合成中的第一个层取消显示并重命名。这或许会关闭合成中的一个图标。

```
var firstLayer = app.project.item(1).layer(1);
firstLayer.enabled = false;
firstLayer.name = "不显示的层";
```

属性：

属性	描述
index	层的索引号
name	层的名称
parent	层的父级
time	层的当前时间
startTime	层的起始时间
stretch	层的时间伸缩百分比
inPoint	层的入点时间
outPoint	层的出点时间
enabled	值为 true 时，层被开启
solo	值为 true 时，层被单独显示
shy	值为 true 时，层被隐藏

locked	值为 true 时，层被锁
hasVideo	值为 true 时，层包含视频
active	值为 true 时，层在当前时间被开启
nullLayer	值为 true 时，这是一个空对象图层
selectedProperties	所有层中被选中的 AE 属性
comment	层的描述内容
containingComp	包含这个图层的合成
isNameSet	值为 true 时，层设置了名字

方法：

方法	描述
remove()	从合成中删除层
moveToBeginning()	将层移动到合成最顶层（使其成为第一个层）
moveToEnd()	将层移动到合成最底层（使其成为最后一个层）
moveAfter()	将层移动到另一个层的下面
moveBefore()	将层移动到另一个层的上面
duplicate()	复制层
copyToComp()	复制层到另一个合成内的顶层
activeAtTime()	层在指定时间是否开启
setParentWithJump()	给层设置一个新父级
applyPreset()	将动画预设设置给层

app.project.item().layer().active

层对象属性：app.project.item(index).layer(index).active

描述：

当值为 true 时，层的视频在当前时间被激活。

如果值为 true，层一定是开启状态，没有其他层被单独显示，除非这个层也被单独显示，并且时间一定在入点和出点时间之间。

在音频层这个属性值总为 false，在媒体层对象中有 audioACtive 属性专门为音频层调用。
类型：布尔型，只读
笔记：*实际上，这个属性值用在固态层之类的也可以。这个属性的实际用途应该就是检测图层有没¥有开启，跟*
系，而音频层之所以不能使用这个属性值，是因为音频层压根就没有那个小眼睛。

app.project.item().layer().activeAtTime()

层对象方法：app.project.item(index).layer(index).activeAtTime(*time*)
描述：如果层在指定时间被激活则返回 true。如果返回 true，层一定处于开启状态，没有其他层被单独显示，除非这个层也被单独显示，并且时间一定在入点和出点时间之间。
参数：

time	秒数，浮点型

返回：布尔型

app.project.item().layer().applyPreset()

层对象方法：app.proejct.item(index).layer(index).applyPreset(*presetName*)
描述：将指定的动画预设添加到层。预先保存的动画预设文件安装在预设文件夹，并且用户可以通过 UI 界面创建新的动画预设。
参数：

presetName	动画预设文件

返回：无

app.project.item().layer().comment

层对象属性：app,project.item(index).layer(index).comment
描述：层的描述内容。
类型：字符串型，读/写

app.project.item().layer().containingComp

层对象属性：app.project.item(index).layer(index).containingComp
描述：包含这个层的合成。
类型：合成项目对象，只读

app.project.item().layer().copyToComp()

层对象方法：app.project.item(index).layer(index).copyToComp(*intoComp*)
描述：
复制这个层到指定的合成。原始图层保持不变。创建一个新的层对象，属性值与复制层相同，在目标合成项目对象中预先将这个新对象给层收集对象。使用 intoComp(1)来检索复制的层。
复制进一个层会改变之前合成中层的索引号。这与在 UI 界面中复制和粘贴层操作一样。
参数：

intoComp	目标合成，一个合成项目对象

返回：无

app.project.item().layer().duplicate()

层对象方法：app.project.item(index).layer(index).duplicate()
描述：复制这个层。创建一个新的层对象，属性值与源图层相同。等同于在 UI 界面中使用编辑->复制操作。
参数：无
返回：无

app.project.item().layer().enabled

层对象属性：app.project.item(index).layer(index).enabled
描述：值为 true 时，层被开启，否则为 false。等同于时间线面板中层前端的视频开关。
类型：布尔型，读/写
笔记：*视频开关就是那个小眼睛开关。*

app.project.item().layer().hasVideo

层对象属性：app.project.item(index).layer(index).hasVideo
描述：值为 true 时，层在时间线面板有视频开关（小眼睛图标），否则为 false。
类型：布尔型，只读
笔记：*不管这个小眼睛开关有没有开启都会返回 true，所以这个属性只能来检测是不是音频层。*

app.project.item().layer().index

层对象属性：app.project.item(index).layer(index).index

描述：层的位置索引号
类型：整数型，范围[1~层数]，只读

app.project.item().layer().inPoint

层对象属性：app.project.item(index).layer(index).inPoint
描述：层在合成中的“入”点时间。
类型：浮点型，范围[-10800.0~10800.0]（正负三个小时），读/写

app.project.item().layer().isNameSet

层对象属性：app.project.item(index).layer(index).isNameSet
描述：如果层被指定了名称则返回 true。
类型：布尔型，只读
笔记：*层有源名称和层名称，源名称就是新建层时的命名，层名称则是在时间线面板上的命名，同一个层在不同位置或合成中可以有不同的名字，源名称则只为同一个。*

app.project.item().layer().locked

层对象属性：app.project.item(index).layer(index).locked
描述：值为 true 时，层被锁住，否则为 false。等同于层面板中的锁开关。
类型：布尔型，读/写

app.project.item().layer().moveAfter()

层对象方法：app.project.item(index).layer(index).moveAfter(*layer*)
描述：将层移动到指定层的下面。
参数：

layer	目标层，相同合成内的层对象

返回：无

app.project.item().layer().moveBefore()

层对象方法：app.project.item(index).layer(index).moveBefore(*layer*)
描述：将层移动到指定层的上面
参数：

layer	目标层，相同合成内的层对象

返回：无

app.project.item().layer().moveToBeginning

层对象方法：app.project.item(index).layer(index).moveToBegining()
描述：将层移动到合成的最顶部（使其成为第一个层）。
参数：无
返回：无

app.project.item().layer().moveToEnd

层对象方法：app.project.item(index).layer(index).moveToEnd()
描述：将层移动到合成的最底部（使其成为最后一个层）。
参数：无
返回：无

app.project.item().layer().name

层对象属性：app.project.item(index).layer(index).name
描述：层的名称。默认下与层源名称（不能在层面板中被更改）相同，但是设置之后就不一样了。
类型：字符串型，读/写

app.project.item().layer().nullLayer

层对象属性：app.project.item(index).layer(index).nullLayer
描述：值为 true 时，层为空对象，否则返回 false。
类型：布尔型，只读

app.project.item().layer().outPoint

层对象属性：app.project.item(index).layer(index).outPoint
描述：层在合成中的“出”点时间。
类型：浮点型，范围[-10800.0~10800.0]（正负三小时），读/写

app.project.item().layer().parent

层对象属性：app.project.item(index).layer(index).parent
描述：
层的父级，可以为 null。
设置父级后默认会计算父级的变换值然后产生偏移值，偏移值会被用来平衡子级层。比如一个父级旋转属性为 30°，设置子级后，子级的旋转属性为-30°。
使用 setParentWithJump 方法来设置设置父级后不产生偏移值。
类型：层对象或 null，读/写

app.project.item().layer().remove()

层对象方法：app.project.item(index).layer(index).remove
描述：从合成中删除指定的层。
参数：无
返回：无

app.project.item().layer().selectedProperties

层对象属性：app.project.item(index).layer(index).selectedProperties
描述：一个包含层当前所有被选中的属性和属性组的数组。
类型：属性基础对象的数组，只读

app.project.item().layer().setParentWithJump()

层对象方法：app.project.item(index).layer(index).setParentWithJump(*newParent*)
描述：
设置当前层的父级为指定的层，并且子级层没有变换属性改变。或许会跳过子级层的旋转、透明度或缩放，因为子级层的变换属性与之前相同。
如果不想跳过子级层属性，直接使用 parent 属性。这种情况下，会计算偏移值改变子级的变换。
参数：

newParent	可选，相同合成中的层对象。如果没有指定，会设置父级为 None

返回：无

app.project.item().layer().shy

层对象属性：app.project.item(index).layer(index).shy
描述：值为 true 时，如果合成的“隐藏层”选项开启，层会在层面板中被隐藏。
类型：布尔型，读/写

app.project.item().layer().solo

层对象属性：app.project.item(index).layer(index).solo
描述：值为 true 时，层被单独显示，否则为 false。
类型：布尔型，读/写

app.project.item().layer().startTime

层对象属性：app.project.item(index).layer(index).startTime
描述：层的开始时间，以合成时间为标准（秒数）。
类型：浮点型，范围[-10800.0~10800.0]（正负三个小时），读/写

app.project.item().layer().stretch

层对象属性：app.project.item(index).layer(index).stretch
描述：层的时间伸缩，以百分比为单位。100 代表没有伸缩。值在 0 和 1 之间会被设置为 1，在-1 和 0 之间（不包括 0）会设置为-1。
类型：浮点型，范围[-9900.0~9900.0]，读/写

app.project.item().layer().time

层对象属性：app.project.item(index).layer(index).time
描述：层当前时间，以合成中时间为标准（秒数）。
类型：浮点型，只读

层收集对象

app.project.item(index).layers

描述：

层收集对象代表了一系列层。层收集对象属于包含了合成中所有层的层对象的合成项目对象。收集对象的方法允许你使用这些层。

- 层收集对象是收集对象的子类。所有收集对象的方法和属性都可以被层收集对象调用。

举例：

假定工程面板中第一个项目是合成项目，第二个项目是媒体项目，这个例子展示了在合成项目对象中层收集对象的层数，将工程面板中的媒体项目作为新层添加到合成中，并且显示新的层数。

```
var firstComp = app.project.item(1);
var layerCollection = firstComp.layers;
alert("层数之前是" + layerCollection.length);
var anAVItem = app.project.item(2); layerCollection.add(anAVItem);
alert("层数现在是" + layerCollection.length);
```

方法：

方法	描述
add()	创建一个新的媒体层并添加到收集对象中
addNull()	创建一个新的空对象层并添加到收集对象中
addSolid()	创建一个新的层和固态层源的素材项目并添加到收集对象中
addText()	创建一个新的点文本层并添加到收集对象中
addBoxText()	创建一个新的段落（框）文本层并添加到收集对象中
addCamera()	创建一个新的摄像机层并添加到收集对象中
addLight()	创建一个新的灯光层并添加到收集对象中
addShape()	创建一个新的形状图层并添加到收集对象中
byName()	用指定的名称检索层对象
precompose()	预合成指定的层

app.project.item().layers.add()

层收集对象：app.project.item(index).layers.add(*item,duration*)
描述：
将指定的项目作为媒体层对象添加到合成中，并且添加到收集对象中。
创建的层默认为合成开始时间。
这个方法会在项目不能作为层添加到合成项目中时生成一个 Exception。
参数：

item	被添加的项目的媒体项目对象
duration	可选，静止图层的时间长度，浮点型。只有项目为静止素材时才可用。在视频、序列和音频上无效。 如果支持，最好设置新层的时长，否则会按着用户偏好自动设置。默认下，会与合成项目的时长相同。

返回：无

app.project.item().layers.addBoxText()

层收集对象方法：app.project.item(index).layers.addBoxText(*sourceText*)
描述：
创建一个新的段落（框）文本层并且将新的文本层对象添加到收集对象中。
使用 addText()创建点文本层。
参数：

sourceText	可选，包含新层文本内容的字符串或者文本文档对象，具体查看“文本文档对象”。

返回：文本层对象
笔记：*这个方法应该是写错了，落下一个参数。这个参数是含有两个浮点数的数组。有且只能有两个浮点数项目。第一个数组项目定义了段落框的宽，第二个数组项目定义了段落框的高。这个数组参数在源文本参数之前。*

app.project.item().layers.addCamera()

层收集对象方法：app.project.item(index).layers.addCamera(*name,centerPoint*)
描述：
创建一个新的摄像机层并添加到收集对象中。
创建的层默认为合成开始时间。
参数：

name	包含新层名称的字符串
centerPoint	摄像机的中心点，浮点型数组[x,y]。用来设置摄像机兴趣点属性的初始值，Z 轴默认为 0。

返回：摄像机层对象

app.project.item().layers.addLight()

层收集对象方法：app.project.item(index).layers.addLight(*name,centerPoint*)

描述：

创建一个新的灯光层并且添加到收集对象中。

创建的层默认为合成开始时间。

参数：

name	包含新层名称的字符串
centerPoint	灯光层的中心点，浮点型数组[x,y]。用来设置摄像机兴趣点属性的初始值。

app.project.item().layers.addNull()

层收集对象方法：app.project.item(index).layers.addNull(*duration*)

描述：创建一个新的空对象层并添加到收集对象中。等同于选择图层->新建->空对象操作。

参数：

duration	可选，静止图层的时间长度，浮点型。只有项目为静止素材时才可用。在视频、序列和音频上无效。 如果支持，最好设置新层的时长，否则会按着用户偏好自动设置。默认下，会与合成项目的时长相同。

返回：媒体层对象

app.project.item().layers.addShape()

层收集对象：app.project.item(index).layers.addShape()

描述：

创建一个新的空形状图层的形状图层对象。使用形状图层对象来添加属性，比如形状、填充、描边和路径等等。

等同于使用图形工具，图像工具会自动添加填充、描边等等的矢量组。

参数：无

返回：形状图层对象

app.project.item().layers.addSolid()

层收集对象方法：app.project.item(index).layers.addSolid(*color,name,width,height,pixelAspect,duration*)

描述：创建一个指定参数的新固态层源对象，将新的固态层源对象作为素材项目对象的真实源属性值，并且添加到工程的素材项目中。创建一个新的媒体层对象，作为新媒体项目的源，并添加到收集属性中。

参数：

color	固态层的颜色，三个浮点型值的数组，[R,G,B]，范围[0.0~1.0]
name	包含固态层名称的字符串
width	固态层的像素宽，整数型，范围[4~30000]
height	固态层的像素高，整数型，范围[4~30000]
pixelAspect	固态层的像素长宽比，浮点型，范围[0.01~100.0]
duration	可选，静止图层的时间长度，浮点型。只有项目为静止素材时才可用。在视频、序列和音频上无效。 如果支持，最好设置新层的时长，否则会按着用户偏好自动设置。默认下，会与合成项目的时长相同。

返回：媒体层对象

app.project.item().layers.addText()

层收集对象方法：app.project.item(index).layers.addText(*sourceText*)

描述：

创建一个新的点文本层并且添加将新的文本层对象到此收集对象中。

使用 addBoxText()方法创建段落（框）文本图层。

参数：

sourceText	可选，包含新层文本内容的字符串或者文本文档对象，具体查看“文本文档对象”。

返回：文本层对象

app.project.item().layers.precompose()

层收集对象方法：app.project.item(index).layers.precompose(*layerIndicies,name,moveAllAttributes*)

描述：创建一个新的合成项目对象并将指定的层移动到它的层收集对象中。这个方法将删除此收集对象中的一些层并添加一个新的合成项目对象。

参数：

layerIndices	要预合成的层的位置索引，整数型的数组
name	新合成项目对象的名称
moveAllAttributes	可选。当值为 true 时（默认），在新合成中保留所有属性。 等同于预合成时在对话框选择“移动所有属性到新的合成”。 如果 layerIndices 数组只有一个项目，你可以设置这个参数为 false。 等同于在预合成对话框中选择“保留所有属性”

返回：合成项目对象

app.project.item().layers.byName()

层收集对象：app.project.item(index).layers.byName(*name*)

描述：返回在此收集对象中指定名称找到的第一个层，没有找到则返回 Null。

参数：

name	包含名称的字符串

返回：层对象或 null

灯光层对象

app.project.item(index).layer(index)

描述：

灯光层对象代表一个合成内的灯光层。使用层收集对象的 addLight 方法创建。它可以通道项目的层收集对象通过索引号或名称访问。

- 灯光层对象是层对象的子类。所有层对象的方法和属性都可以被灯光层对象调用。

AE 属性：

灯光层没有额外定义属性，但是有一些与其他类型层不同的 AE 属性。它有以下这些属性和属性组。

```
Marker
Transform
        Point of Interest
        Position
        Scale
        Orientation
        X Rotation
        Y Rotation
        Rotation
        Opacity
Light Options
        Intensity
        Color
        Cone Angle
        Cone Feather
        Casts Shadows
        Shadow Darkness
        Shadow Diffusion
```

属性：

属性	描述
lightType	灯光的类型

app.project.item().layer().lightType

灯光层对象属性：app.project.item(index).layer(index).lightType

描述：

对于灯光层，定义了灯光的类型。

对非灯光层尝试设置这个属性会报错。

类型：

一些列举的灯光类型值，读/写

LightType.PARALLEL
LightType.SPOT
LightType.POINT
LightType.AMBIENT

标记对象

new MarkerValue(*comment*,*chapter*,*url*,*frameTarget*,*cuePointName*,*params*)

描述：

标记对象代表一个层标记，这个标记可能包含相关注释、可选的章节参考点、网页链接、在层中具有特定点的 Flash 视频提示点等等。使用这个构造函数创建标记，除了 comment 都是可选的。所有参数都是字符串，等同于标记对象返回的属性值，除了 params，这是一个包含一对参数名称-参数值的数组，可以通过 getParameters()和 setParameters()访问。脚本可以设置任意数量的参数，参数的顺序不反应软件中显示的顺序。

为了连接标记和层，在层的标记属性中设置标记对象：

```
layerObject.property("Marker").setValueAtTime(time, markerValueObject);
```

属性：

属性	描述
comment	在连接层上的注释
duration	标记的时长
chapter	连接层的章节链接参考点
cuePointName	Flash 提示点名称
eventCuePoint	Flash 提示点是否用于事件或导航。
url	在连接层上的网页 URL
frameTarget	通过 URL 指定的网页内的制定帧目标

方法：

方法	描述
getParameters()	检索连接标记值的关键值对
setParameters()	设置连接标记值的关键值对

举例：

- 在 2 秒处设置标记，注明“淡入淡出”

```
var myMarker = new MarkerValue("淡入淡出");
myLayer.property("Marker").setValueAtTime(2, myMarker);
```

- 从指定标记中取回注释值

```
var commentOfFirstMarker =
app.project.item(1).layer(1).property("Marker").keyValue(1).comment;
var commentOfMarkerAtTime4 =
  app.project.item(1).layer(1).property("Marker").valueAtTime(4.0, true).comment
var markerProperty = app.project.item(1).layer(1).property("Marker");
var markerValueAtTimeClosestToTime4 =
  markerProperty.keyValue(markerProperty.nearestKeyIndex(4.0));
var commentOfMarkerClosestToTime4 = markerValueAtTimeClosestToTime4.comment;
```

笔记：

除了第一个注释参数，后面那些都不是直接在时间线面板上显示的，也就是说剩下的参数几乎没有什么用。需要双击标记打开标记面板才能看到,那些参数并不是为了工程中用到的，而是给输出的视频带一个标记，支持这些标记的视频播放器可以相对应的做出反应。

第一个代码中的 myLayer 这里做了简略处理，缺少一个变量赋值：

var myLayer = app.project.item(index).layer(index);

app.project.item().layer().property().keyValue().chapter

标记对象属性：

app.project.item(index).layer(index).property(“Marker”).keyValue(index).chapter

描述：标记的章节链接。章节链接创建一个章节跳跃点，在 QuickTime 播放器或其它支持章节标记的播放器中可以直接跳跃到设置的时间点。

类型：字符串，读/写

app.project.item().layer().property().keyValue().comment

标记对象属性：

app.project.item(index).layer(index).property(“Marker”).keyValue(index).comment

描述：标记的注释。这个注释显示在时间线面板靠近层标记的位置。

类型：字符串，读/写

app.project.item().layer().property().keyValue().cuePointName

标记对象属性：

app.project.item(index).layer(index).property(“Marker”).keyValue(index).cuePointName

描述：Flash 提示点，与标记对话框中显示的一致。

类型：字符串，读/写

app.project.item().layer().property().keyValue().duration

标记对象属性：
app.project.item(index).layer(index).property(“Marker”).keyValue(index).duration
描述：标记的时长，秒数。时长以时间线面板上标记后的短柄来显示。
类型：浮点型，读/写

app.project.item().layer().property().keyValue().eventCuePoint

标记对象属性：
app.project.item(index).layer(index).property(“Marker”).keyValue(index).eventCuePoint
描述：值为 true 时，Flash 提示点为一个事件，否则为导航。
类型：字符串，读/写

app.project.item().layer().property().keyValue().frameTarget

标记对象属性：
app.project.item(index).layer(index).property(“Marker”).keyValue(index).frameTarget
描述：标记的帧目标。与 URL 值绑定在一起，这个属性在网页内指定一个帧。
类型：字符串，读/写

app.project.item().layer().property().keyValue().getParameters()

标记对象方法：
app.project.item(index).layer(index).property(“Marker”).keyValue(index).getParameters()
描述：从连接这个标记值的提示点中返回 Flash 提示点参数的参数名称和参数值对。
参数：无
返回：一组属性的对象，匹配每个参数名称和参数值。

app.project.item().layer().property().keyValue().setParameters()

标记对象方法：
app.project.item(index).layer(index).property(“Marker”).keyValue(index).setParameters(*keyValuePairs*)
描述：连接 Flash 提示点参数的一对参数名称-参数值。提示点可以有任意多的参数，但是通过 UI 界面只能添加三个，使用这个方法可以添加不止三个。
参数：

keyValuePairs	包含作为属性和值的参数名称-参数值的对象。调用对象的 toString()方法来分配给每个属性字符串值作为名称。

返回：无

举例：

```
var mv = new MarkerValue("My Marker");
var parms = new Object; parms.timeToBlink = 1;
parms.assignMe = "A string"
mv.setParameters(parms);
myLayer.property("Marker").setValueAtTime(2, mv);
```

app.project.item().layer().property().keyValue().url

标记对象属性：

app.project.item(index).layer(index).property(“Marker”).keyValue(index).url

描述：标记的 URL 值。这个 URL 自动连接到一个网页。

类型：字符串型，读/写

蒙版属性组对象

app.project.item(index).layer(index).mask
描述：
蒙版属性组对象封装了层中蒙版的属性。

- 蒙版属性组对象是属性组对象的子类。所有属性基础和属性组对象的方法和属性都可以被蒙版属性组调用。

属性：

属性	描述
maskMode	蒙版模式
inverted	值为 true 时，蒙版被反转
rotoBezier	值为 true 时，使用自动曲线创建新蒙版
maskMotionBlur	蒙版应用的动态模糊方式
locked	值为 true 时，蒙版被锁
color	UI 界面中蒙版的边缘线颜色
maskFeatherFalloff	蒙版的羽化模式

app.project.item().layer().mask().color

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).color
描述：用来在 UI 界面中画蒙版时的边缘线颜色（合成面板、层面板和时间线面板）。
类型：三个浮点型数组，[R,G,B]，范围[0.0~1.0]，读/写

app.project.item().layer().mask().inverted

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).inverted
描述：值为 true 时，蒙版被反转，否则为 false。
类型：布尔型，读/写

app.project.item().layer().mask().locked

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).locked
描述：值为 true 时，蒙版被锁并且在 UI 界面中不能被编辑，否则为 false。
类型：布尔型，读/写

app.project.item().layer().mask().maskFeatherFalloff

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).maskFeatherFalloff
描述：蒙版的羽化模式。等同于选择图层->蒙版->羽化设置。
类型：
一些列举的蒙版羽化值，读/写
MaskFeatherFalloff.FFO_LINEAR
MaskFeatherFalloff.FFO_SMOOTH

app.project.item().layer().mask().maskMode

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).maskMode
描述：蒙版的类型
类型：
一些列举的蒙版类型值，读/写
MaskMode.NONE
MaskMode.ADD
MaskMode.SUBTRACT
MaskMode.INTERSECT
MaskMode.LIGHTEN
MaskMode.DARKEN
MaskMode.DIFFERENCE

app.project.item().layer().mask().maskMotionBlur

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).maskMotionBlur
描述：蒙版应用动态模糊方式
类型：
一些列举的蒙版动态模糊模式的值，读/写
MaskMotionBlur.SAME_AS_LAYER
MaskMotionBlur.ON
MaskMotionBlur,OFF

app.project.item().layer().mask().rotoBezier

蒙版属性组对象属性：app.project.item(index).layer(index).mask(index).rotoBezier
描述：值为 true 时，蒙版开启了使用自动曲线创建新蒙版模式，否则为 false。
类型：布尔型，读/写

输出模块收集对象

app.project.renderQueue.items.outputModules

描述：

输出模块收集对象包含了渲染序列中所有的输出模块。这个收集对象提供了输出模块对象的接口，但是不提供附加功能。第一个输出模块对象在收集对象中的索引位置为 1。

- 输出模块收集对象是收集对象的子类。所有收集对象的方法和属性都可以被输出模块收集对象调用。

输出模块对象

app.project.renderQueue.item(index).outputModule(index)
描述：渲染序列项目的一个输出模块对象生成一个渲染操作的单个文件或序列，并且包含被渲染文件的属性和方法。
属性：

属性	描述
file	被渲染文件的输出路径和名称
postRenderAction	渲染后要执行的动作
name	输出模块在 UI 界面中显示的名称
templates	输出模块的所有模板
includeSourceXMP	值为 true 时，写下所有输出文件的源素材 XMP 元数据

方法：

方法	描述
remove()	从渲染队列项目的列表中删除指定的输出模块
saveAsTemplate()	保存一个新的输出模块模板
applyTemplate()	应用一个输出模块模板

app.project.renderQueue.item().outputModule().applyTemplate()

输出模块对象方法：
app.project.renderQueue.item(index).outputModule(index).applyTemplate(t-emplateName)
描述：应用指定的输出模块模板
参数：

templateName	被应用的模板的名称

返回：无

app.project.renderQueue.item().outputModule().file

输出模块对象属性：app.project.renderQueue.item(index).outputModule(index).file
描述：被渲染的输出模块的文件的 ExtendScript 文件对象。
类型：ExtendScript 文件对象，读/写

app.project.renderQueue.item().outputModule().includeSourceXMP

输出模块对象属性：
app.project.renderQueue.item(index).outputModule(index).includeSourceXMP
描述：值为 true 时，将所有源素材 XMP 元数据写入到输出文件，等同于在输出模块设置对话框中的包括源 XMP 元数据选项。
类型：布尔型，读/写

app.project.renderQueue.item().outputModule().name

输出模块对象属性：app.project.renderQueue.item(index).outputModule(index).name
描述：输出模块的名称，与 UI 界面上显示的相同。
类型：字符串型，只读

app.project.renderQueue.item().outputModule().postRenderAction

输出模块对象属性：
app.project.renderQueue.item(index).outputModule(index).postRenderAction
描述：渲染结束后要执行的动作。
类型：
一些列举的渲染后动作值，读/写
postRenderAction.NONE
postRenderAction.IMPORT
postRenderAction.IMPORT_AND_REPLACE_USAGE
postRenderAction.SET_PROXY

app.project.renderQueue.item().outputModule().remove()

输出模块对象方法：app.project.renderQueue.item(index).outputModule(index).renove()
描述：从收集对象中删除指定的输出模块对象。
参数：无
返回：无

app.project.renerQueue.item().outputModule().savaAsTemplate()

输出模块对象方法：
app.project.renderQueue.item(index).outputModule(index).saveAsTemplate(*name*)
描述：保存输出模块为模板并添加到模板数组中。
参数：

name	新模板的名称，字符串型

返回：无

app.project.renderQueue.item().outputModule().templates

输出模块对象属性：app.project.renderQueue.item(index).outputModule(index).templates
描述：AE 中本地安装的所有输出模块模板的名字。
类型：字符串的数组，只读

占位符源对象

app.project.item(index).mainSource
app.project.item(index).proxySource

描述：

占位符源对象描述了占位符的素材源。

占位符源对象是素材源对象的子类。所有的素材源对象的方法和属性都可以被占位符源对象调用。

占位符源没有额外的方法或属性。

工程对象

app.project

描述：

工程对象代表一个 AE 工程。对象属性提供了工程内指定对象的接口，比如导入文件或素材和合成，再比如时间码等等的工程设置。对象方法可以导入素材、创建固态层、合成和文件夹、保存更改等等。

属性：

属性	描述
file	当前打开的工程文件
rootFolder	包含工程所有内容的文件夹，等同于工程面板
items	工程内项目的数量
activeItem	当前激活的项目
bitsPerChannel	工程当前的颜色深度
transparencyGridThumbnails	值为 true 时，缩略图预览使用幻灯片式棋盘模式
numItems	工程内包含的全部项目数
selection	工程面板中被选中的全部项目
renderQueue	工程的渲染序列
timeDisplayType	时间显示的风格，等同于在工程设置对话框中时间显示风格选项
footageTimecodeDisplayStartType	工程设置对话框中的素材开始时间的设置，当时间码被选择作为时间显示风格时可以使用
framesUseFeetFrames	在工程设置对话框中的使用英尺数+帧数的设置
feetFramesFilmType	在工程设置对话框中的使用英尺数+帧数菜单的设置
framesCountType	在工程设置对话框中的帧数菜单设置
displayStartFrame	显示工程时开始编号的帧

linearBlending	值为 true 时，工程中使用线性混合
xmpPacket	工程的 XMP 元数据

方法：

方法	描述
item()	从工程中检索一个项目
consolidateFootage()	整合工程中所有素材
removeUnusedFootage()	从工程中删除未使用的素材
reduceProject()	减少项目
close()	使用正常保存选项关闭工程
save()	保存工程
saveWithDialog()	显示保存对话框
importPlaceholder()	导入一个占位符到工程中
importFile()	导入一个文件到工程中
importFileWithDialog()	显示导入文件对话框
showWindow()	显示或隐藏工程面板
autoFixExpressions	自动替换所有表达式中的文本

app.project.activeItem

工程对象属性：app.project.activeItem

描述：当前正在被激活的项目，如果没有项目被选中或多个项目被选中都会返回 null。

类型：项目对象或 null，只读

app.project.autoFixExpressions()

工程对象方法：app.project.autoFixExpressions(*oldText*,*newText*)

描述：自动替换工程中出错的表达式文本。

参数：

oldText	要替换的文本

newText	新文本

返回：无

app.project.bitsPerChannel

工程对象属性：app.project.bitsPerChannel
描述：当前工程的颜色深度，8 位、16 位或 32 位。
类型：整数型，8、16 或 32，读/写

app.project.close()

工程对象方法：app.project.close(*closeOptions*)
描述：自动使用保存更改选项关闭工程，或提示用户保存更改还是不保存更改。
参数：

closeOptions	执行关闭时的动作。一些列举的关闭选项值： CloseOptions.DO_NOT_SAVE_CHANGES:不保存直接关闭 CloseOptions.PROMPT_TO_SAVE_CHANGES:关闭前提示是否保存修改 CloseOptions.SAVE_CHANGES：关闭时自动保存

返回：布尔型，成功返回 true。如果文件没有被提前保存并且用户选择提示框的取消选项则返回 false。

app.project.consolidateFootage()

工程对象方法：app.project.consolidateFootage()
描述：整合工程内所有素材。等同于文件->整个所有素材命令。
参数：无
返回：整数型，被删除的素材总数

app.project.displayStartFrame

工程对象属性：app.project.displayStartFrame
描述：在工程设置对话框中设置帧数为 1 或 0 的备用方式，等同于将 framesCountType 属性值设为 FramesCountType.FC_START_0 或 FramesCountType.FC_START_1。
类型：整数型，0 或 1，读/写

app.project.feetFrameFilmType

工程对象属性：app.project.feetFrameFilmType

描述：

在工程设置对话框中使用英尺数+帧数菜单设置。

使用这个属性代替已经过时的 timecodeFilmType 属性。

类型：

一些列举的英尺帧影片类型，读/写

FeetFramesFilmType.MM16
FeetFramesFilmType.MM35

app.project.file

工程对象属性：app.project.file

描述：包含当前打开的工程文件的 ExtendScript 文件对象。

类型：文件对象，如果程序没有保存则返回 null，只读

app.project.footageTimecodeDisplayStartType

工程对象属性：app.project.footageTimecodeDisplayStartType

描述：工程设置对话框中的素材开始时间设置，当时间码被选择作为时间显示样式时可用。

类型：

一些列举的素材时间码显示开始类型，读/写

FootageTimecodeDisplayStartType.FTCS_START_0
FootageTimecodeDisplayStartType.FTCS_USE_SOURCE_MEDIA

app.project.framesCountType

工程对象属性：app.project.framesCountType

描述：工程设置对话框中帧数菜单设置。

类型：

一些列举的帧数类型值，读/写

FramesCountType.FC_START_1
FramesCountType.FC_START_0
FramesCountType.FC_TIMECODE_CONVERSION

注意：设置属性值为 FramesCountType.FC_TIMECODE_CONVERSION 会重置 displayStart-Frame 属性值为 0。

app.project.framesUseFeetFrames

工程对象属性：app.project.framesUseFeetFrames
描述：工程设置对话框中的使用英尺数+帧数设置。如果使用英尺数+帧数则返回 true，使用帧数则返回 false。
类型：布尔型，读/写

app.project.importFile()

工程对象方法：app.project.importFile(importOptions)
描述：在指定的导入选项对象中导入指定文件。等同于文件->导入文件命令。从文件中创建一个新的素材项目对象，并且将其添加到工程的 items 数组中。
参数：

importOptions	在指定的导入选项对象中导入指定文件。具体查看 “导入选项对象”

返回：素材项目对象
举例：

```
app.project.importFile(new ImportOptions(File("案例.psd")))
```

app.project.imporFileWithDialog()

工程对象方法：app.project.importFileWithDialog()
描述：显示导入文件对话框。等同于文件->导入->文件命令。
返回：导入文件时创建的项目对象的数组，如果用户取消对话框则返回 null

app.project.importPlaceholder()

工程对象方法：app.project.importPlaceholder(*name,width,height,frameRate,duration*)
描述：创建并返回一个新的占位符项目对象，并添加到工程的 items 数组中。等同于文件->导入->占位符命令。
参数：

name	包含占位符名称的字符串
width	占位符的像素宽，整数型，范围[4~30000]
height	占位符的像素高，整数型，范围[4~30000]
frameRate	占位符的帧速率，浮点型，范围[1.0~99.0]

duration	占位符的时长，秒数，浮点型，范围[0.0~10800.0]

返回：占位符项目对象

app.project.item()

工程对象方法：app.project.item(index)

描述：检索指定索引位置的项目。

参数：

index	项目的索引位置，整数型，第一个索引号是 1

返回：项目对象

app.project.items

工程对象属性：app.project.items

描述：工程内所有的项目。

类型：项目收集对象，只读

app.project.linearBlending

工程对象属性：app.project.linearBlending

描述：如果用户在工程中使用了线性混合则返回 true，否则为 false。

类型：布尔型，读/写

app.project.numItems

工程对象属性：app.project.numItems

描述：工程内包含的项目总数，包括文件夹和所有类型的素材。

类型：整数型，只读

举例：

```
n = app.project.numItems;
alert("工程内一共有" + n + "个项目")
```

app.project.reduceProject()

工程对象方法：app.project.reduceProject(array_of_items)

描述：删除工程总所有未指定保留的项目。等同于文件->减少项目命令。

参数：

array_of_items	包含保留的项目对象的数组

返回：整数型，被删除项目的总数。

举例：

```
var theItems = new Array();
theItems[theItems.length] = app.project.item(1);
theItems[theItems.length] = app.project.item(3);
app.project.reduceProject(theItems);
```

笔记：*因为 theItems 是数组，所以给两次变量等于添加一个数组的项目。*

app.project.removeUnusedFootage()

工程对象方法：app.project.removeUnusedFootage()

描述：删除工程内所有未使用的素材。等同于文件->删除未使用素材命令。

参数：无

返回：整数型，被删除的素材项目对象的总数

app.project.renderQueue

工程对象属性：app.project.renderQueue

描述：工程的渲染队列。

类型：渲染队列对象，只读

app.project.rootFolder

工程对象属性：app.project.rootFolder

描述：包含工程内容的根文件夹，这是一个工程面板中包含所有项目的虚拟文件夹，但是工程面板中项目没有在此文件夹之外的。

类型：文件夹项目对象，只读

app.project.save()

工程对象方法：app.project.save(*file*)

描述：保存工程。等同于文件->保存->保存为命令。如果当前工程还未保存过并且没有指定文件路径，那么就提示用户选择保存路径和文件名称。将工程保存为新文件会忽略提示。

参数：

file	可选，要保存的文件的 ExtendScript 文件对象

返回：无

app.project.saveWithDialog()

工程对象方法：app.project.saveWithDialog()
描述：显示保存对话框。用户可以命名文件、选择保存路径和保存文件，或者选择取消退出对话框。
参数：无
返回：布尔型，如果工程被保存返回 true。

app.project.selection

工程对象属性：app.project.selection
描述：工程面板中所有被选中的项目，按着工程面板中显示的顺序。
类型：项目对象的数据，只读

app.project.showWindow()

工程对象方法：app.project.showWindow(*doShow*)
描述：显示或隐藏工程面板。
参数：

doShow	值为 true 时，显示工程面板。值为 false 时，隐藏工程面板。

app.project.timeDisplayType

工程对象属性：app.project.timeDisplayType
描述：时间显示样式，等同于工程设置对话框中时间显示样式设置。
类型：
一些列举的时间显示样式的值，读/写
TimeDisplayType.FRAMES
TimeDisplayType.TIMECODE

app.project.transparencyGridThumbnails

工程对象属性：app.project.transparencyGridThumbnails
描述：值为 true 时，缩略图视图使用幻灯片式棋盘模式。
类型：布尔型，读/写

app.project.xmpPacket

工程对象属性：app.project.xmpPacket

描述：工程的 XMP 元数据，储存为 RDF（基于 XML）。更多 XMP 的信息，可以查看《JavsScript 工具指南》。

类型：字符串，读/写

举例：

下面的代码存取当前工程的 XMP 元数据，并且修改工程元数据字段标签。

```
var proj = app.project;
// 导入 XMP 库作为 ExtendScript 外部对象
if (ExternalObject.AdobeXMPScript == undefined) {
  ExternalObject.AdobeXMPScript = new
  ExternalObject('lib:AdobeXMPScript');
}
var mdata = new XMPMeta(app.project.xmpPacket); // 取的工程的 XMP 元数据
// 更新工程元数据标签的值
var schemaNS = XMPMeta.getNamespaceURI("xmp");
var propName = "xmp:Label";
try {
  mdata.setProperty(schemaNS, propName, "超级最终版");
} catch (e) {
  alert(e);
}
app.project.xmpPacket = mdata.serialize();
```

属性对象

app.project.item(index).layer(index).propertySpec

描述：

属性对象包含一个层中指定的 AE 属性的值、关键帧和表达式信息。AE 属性是在单独层内的效果、遮罩或变换等的值，通常是动态的。怎么样才能存取属性，具体查看“属性基础对象”和“属性组 property()方法”

- 属性对象是属性基础对象的子类，所有属性基础对象的方法和属性都可以被属性对象调用。

注意：JavaScript 对象的属性(properties)在此手册中叫做属性(attributes)，为了避免和 AE 定义的属性(property)混淆。

属性：

属性	描述
propertyValueType	当前属性中储存的值的类型
value	当前时间属性的值
hasMin	值位 true 时，有一个允许的最小值
hasMax	值位 true 时，有一个允许的最大值
minValue	允许的最小值
maxValue	允许的最大值
isSpatial	值为 true 时，属性定义了一个空间值
canVaryOverTime	值为 true 时，属性可以添加关键帧
isTimeVarying	值为 true 时，属性含有关键帧或表达式能够改变属性值
numKeys	该属性的关键帧数
unitsText	在添加表达式的值内单元的一段文本描述
expression	该属性的表达式
canSetExpression	值为 true 时，可以通过脚本设置表达式
expressionEnabled	值为 true 时，表达式被用来生成属性的值

expressionError	最近一次执行计算的脚本发生的错误，如果有的话
selectedKeys	当前属性所有被选中的关键帧
propertyIndex	当前属性的位置索引号
dimensionsSeparated	值为 true 时，则该属性有多维属性
isSeparationFollower	值为 true 时，该属性为多维属性其中的一个分离属性
isSeparationLeader	值为 true 时，该属性是多维属性且可以被分离
separationDimension	多维属性中的分离属性数量
separationLeader	该分离属性的源多维属性

方法：

方法	属性
valueAtTime()	在指定时间该属性的值
setValue()	给该属性设定一个固定值
setValueAtTime()	给该属性创建一个关键帧
setValueAtTimes()	给该属性创建一系列关键帧
setValueAtKey()	检索一个关键帧并给该属性的关键帧设置一个值
nearestKeyIndex()	检索一个离指定时间最近的一个关键帧
keyTime()	检索当条件发生时的时间
keyValue()	检索当条件发生时间的关键帧值
addKey()	在指定时间给该属性添加新的关键帧
removeKey()	从该属性中移除一个关键帧
isInterpolationTypeValid()	值为 true 时，该属性可以插值
setInterpolationTypeAtKey()	设置关键帧的差值类型
keyInInterpolationType()	检索关键帧“入”插值类型
keyOutInterpolationType()	检索关键帧“出”插值类型
setSpatialTangentsAtKey()	设置该关键帧“入”和“出”的空间切线

keyInSpatialTangent()	检索关键帧的“入”空间切线
keyOutSpatialTangent()	检索关键帧的“出”空间切线
setTemporalEaseAtKey()	设置关键帧“入”和“出”的时间缓动
keyInTemporalEase()	设置关键帧的“入”时间缓动
keyOutTemporalEase()	设置关键帧的“出”时间缓动
setTemporalContinuousAtKey()	设置关键帧是否有时间线性插值
keyTemporalContinuous()	检索关键帧是否有时间线性插值
setTemporalAutoBezierAtKey()	设置关键帧是否有时间自动贝塞尔插值
keyTemporalAutoBezier()	检索关键帧是否有时间自动贝塞尔插值
setSpatialContinuousAtKey()	设置关键帧是否有空间线性插值
keySpatialContinuous()	检索关键帧是否有空间线性插值
setSpatialAutoBezierAtKey()	设置关键帧是否有空间自动贝塞尔插值
keySpatialAutoBezier()	检索关键帧是否有空间自动贝塞尔插值
setRovingAtKey()	设置是否漂浮关键帧
keyRoving()	检索是否漂浮关键帧
setSelectedAtKey()	设置关键帧是否被选中
keySelected()	检索关键帧是否被选中
getSeparationFollower()	对于一个分离属性，在多维属性中检索指定的分离属性

举例：

检索并设置透明度的值

```
var myProperty = myLayer.opacity;
//透明度属性值类型为一维属性，并且为浮点型
myProperty.setValue(50); // 设置透明度值为50%
// 变量myOpacity 为浮点型
var myOpacity = myProperty.value;
```

检索并设置位置的值

```
var myProperty = myLayer.position;
//位置属性值类型为ThreeD_SPATIAL(三维_空间)，并且为三个浮点型的数组
myProperty.setValue([10.0, 30.0, 0.0]);
// 变量myPosition 为三个浮点型的数组
```

```
var myPosition = myProperty.value;
```

将蒙版形状的值由关闭改为开启

```
var myMask = mylayer.mask(1);
var myProperty = myMask.maskPath;
myShape = myProperty.value;
myShape.closed = false;
myProperty.setValue(myShape);
```

检索指定时间点的颜色值。颜色值为四个浮点型项目的数组，[r,g,b,opacity]。这段脚本设置了 4 秒时灯光颜色的红通道值为 2 秒时的一半。

```
var myProperty = myLight.color;
var colorValue = myProperty.valueAtTime(2,true);
colorValue[0] = 0.5 * colorValue[0];
myProperty.setValueAtTime(4,colorValue);
```

检索 3.5 秒时的表达式计算出的缩放值，预期值为[10,50]

```
var myProperty = myLayer.scale;
// preExpression 值为false 意味着属性值为表达式计算的结果
var scaleValue = myProperty.valueAtTime(3.5, false);
if (scaleValue[0] == 10 && scaleValue[1] == 50) {
  alert("哦耶！");
} else {
  alert("糟糕！");
}
```

给旋转设置一个 0 到 90 的关键帧并返回。这个动画持续 10 秒，中间关键帧在 5 秒处。旋转属性为一维值。

```
myProperty = myLayer.rotation;
myProperty.setValueAtTime(0, 0);
myProperty.setValueAtTime(5, 90);
myProperty.setValueAtTime(10, 0);
```

改变某些源文本的前三个关键帧的值。

```
myProperty = myTextLayer.sourceText;
if (myProperty.numKeys < 3) {
  alert("错误，不足三个关键帧。");
} else {
  myProperty.setValueAtKey(1, new TextDocument("key number 1"));
  myProperty.setValueAtKey(2, new TextDocument("key number 2"));
  myProperty.setValueAtKey(3, new TextDocument("key number 3"));
}
```

使用简便语法给位置、缩放、颜色和源文字设置值。

```
// 这两个是等价的，第二个自动填充默认值为0
myLayer.position.setValue([20, 30, 0]);
myLayer.position.setValue([20, 30]);
// 这两个是等价的，第二个自动填充默认值为100
myLayer.scale.setValue([50, 50, 100]);
myLayer.scale.setValue([50, 50]);
// 这两个是等价的，第二个自动填充默认值为1.0
myLight.color.setValue([.8, .3, .1, 1.0]);
myLight.color.setValue([.8, .3, .1]);
// 这两个是等价的，第二个创建一个字段文档
```

```
myTextLayer.sourceText.setValue(new TextDocument("foo"));
myTextLayer.sourceText.setValue("foo");
```

app.project.item().layer().propertySpec.addKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.addKey(*time*)
描述：在指定时间给指定属性添加一个新的关键帧或标记，并且返回新关键帧的索引号。
参数：

time	添加关键帧的时间，秒数，浮点型，合成的开始时间为 0

返回：整数型，关键帧或标记的索引。

app.project.item().layer().propertySpec.canSetExpression

属性对象属性：app.project.item(index).layer(index).*propertySpec*.canSetExpression
描述：值为 true 时，指定的属性可以通过脚本设置表达式，具体查看 “属性 expression 属性”。
类型：布尔型，只读

app.project.item().layer().propertySpec.canVaryOverTime

属性对象属性：app.project.item(index).layer(index).*propertySpec*.canVaryOverTime
描述：值为 true 时，指定的属性可以随着时间而变化，也就是说该属性可以添加关键帧或表达式
类型：布尔型，只读

app.project.item().layer().propertySpec.dimensionsSeparated

属性对象属性：app.project.item(index).layer(index).*propertySpec*.dimensionsSeparated
描述：值为 true 时，属性的维度被多个分离属性表示。比如，层的位置属性由 X 位置和 Y 位置属性表示，那么位置属性的的 dimensionsSeparated 属性为 true。
注意：这个属性只有当 isSeparationLeader 属性为 true 时才能设置。
类型：布尔型，读/写
笔记：*这个属性写入 true 还是 false 没发现有什么形式上的区别。*

app.project.item().layer().propertySpec.expression

属性对象属性：app.project.item(index).layer(index).*propertySpec*.expression
描述：

指定属性的表达式。只有当指定属性的 canSetExpression 值为 true 时才可以写入。当给该属性指定值时，这个字符串值会被评估。

- 如果字符串包含一个有效的表达式，expressionEnabled 属性值变为 true。
- 如果字符串不包含有效的表达式，会生成一个错误，expressionEnable 属性值为 false。
- 如果设置一个空字符串，expression 属性值会变为 false，但是不会生成错误。

类型：字符串，读/写，若果指定属性的 canSetExpression 值为 true。

笔记：*不管字符串内容是什么，都会写入属性的表达式中，只是错误的表达式会在预览面板中报错*

app.project.item().layer().propertySpec.expressionEnabled

属性对象属性：app.project.item(index).layer(index).*propertySpec*.expressionEnabled

描述：值为 true 时，指定的属性使用表达式来生成值。值为 false 时，该属性正在使用关键帧信息或静态值。这个属性只有在指定属性的 canSetExpression 属性值为 true 并且 expression 属性值包含一个有效的表达式字符串时才可以被设置为 true。

类型：布尔型，读/写

app.project.item().layer().propertySpec.expressionError

属性对象属性：app.project.item(index).layer(index).*propertySpec*.expressionError

描述：最近一次设置 expression 属性值生成的错误，如果有的话。如果没有表达式或表达式不包含错误，那么属性值为空字符串（”” ）。

类型：字符串型，只读

app.project.item().layer().propertySpec.getSeparationFollower()

属性对象方法：

app,project.item(index).layer(index).*propertySpec*.getSeparationFollower(dim)

描述：检索多维属性指定的分离属性。比如，可以使用这个方法来存取位置属性的 X 位置和 Y 位置属性。

注意：这个属性只能在 isSeparationLeader 属性值为 true 时被应用。

参数：

dim	分离属性的维度，开始于 0

返回：属性对象，如果该属性不是多维属性或没有指定维数则返回错误。

app.project.item().layer().propertySpec.hasMax

属性对象属性：app.project.item(index).layer(index).*propertySpec*.hasMax

描述：值为 true 时，指定的属性有一个允许的最大值，否则为 false。
类型：布尔型，只读。

app.project.item().layer().propertySpec.hasMin

属性对象属性：app.project.item(index).layer(index).*propertySpec*.hasMin
描述：值为 true 时，指定的属性有一个允许的最小值，否则为 false。
类型：布尔型，只读。

app.project.item().layer().propertySpec.isInterpolationTypeValid()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.isInterpolationType-Valid(*type*)
描述：如果该属性可以使用指定的关键帧插值类型插值，则返回 true。
参数：

type	一些列举的关键帧插值类型的值 KeyframeInterpolationType.LINEAR KeyframeInterpolationType.BEZIER KeyframeInterpolationType.HOLD

返回：布尔型

app.project.item().layer().propertySpec.isSeparationFollower

属性对象属性：app.project.item(index).layer(index).propertySpec.isSeparationFollower
描述：值为 true 时，该属性代表着多维属性其中的一个分离属性。比如，X 位置的该属性值为 true。
注意：原始的、统一的、多维属性叫做 “separation leader”，新的、分离的、单一维的属性叫做 “separation followers”。
类型：布尔型，只读

app.project.item().layer().propertySpec.isSeparationLeader

属性对象属性：app.project.item(index).layer(index).*propertySpec*.isSeparationLeader
描述：值为 true 时，该属性为多维并且能够分离。比如，位置属性的该属性值为 true。
类型：布尔型，只读

app.project.item().layer().propertySpec.isSpatial

属性对象属性：app.project.item(index).layer(index).*propertySpec*.isSpatial
描述：值为 true 时，该属性定义了一个空间值。比如位置和效果的点控制。
类型：布尔型，只读

app.project.item().layer().propertySpec.isTimeVarying

属性对象属性：app.project.item(index).layer(index).*propertySpec*.isTimeVarying
描述：值为 true 时，指定属性随时间而变化，这意味着该属性有关键帧或可用的表达式。当该属性值为 true 时，canVaryOverTime 属性值一定为 true。
类型：布尔型，只读

app.project.item().layer().propertySpec.keyInterpolationType()

属性对象属性：
app.project.item(index).layer(index).*propertySpec*.keyInterpolationType(keyIndex)
描述：返回指定关键帧的“入”插值类型。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：
一些列举的关键帧插值类型，
KeyframeInterpolationType.LINEAR
KeyframeInterpolationType.BEZIER
KeyframeInterpolationType.HOLD

app.project.item().layer().propertySpec.keyInSpatialTangent()

属性对象方法：
app.project.item(index).layer(index).*propertySpec*.keyInSatialTangent(*keyIndex*)
描述：如果指定的属性是空间性质的（即值类型为 TwoD_SPATIAL 或 ThreeD_SPATIAL），返回指定关键帧“入”的空间切线。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：
浮点型项目的数组：

- 如果属性值类型为 isPropertyValueType.TwoD_SPATIAL，则数组包含两个浮点型项目。
- 如果属性值类型为 isPropertyValueType.ThreeD_SPATIAL,则数组包含三个浮点型值。
- 如果属性值类型不为这两个值，会发生一个意外错误。

app.project.item().layer().propertySpec.keyInTemporalEase()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyInTemporalEase(*key-Index*)
描述：返回指定关键帧的“入”时间缓动。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：
关键帧缓动对象的数组

- 如果属性值类型为 isPropertyValueType.TwoD_SPATIAL，则数组包含两个对象。
- 如果属性值类型为 isPropertyValueType.ThreeD_SPATIAL,则数组包含三个对象。
- 其他任何值类型，数组只包含一个对象

app.project.item().layer().propertySpec.keyOutInterpolationType()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyOutInterpolation-Type(keyIndex)
描述：返回指定关键帧的“出”插值类型。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：
一些列举的关键帧插值类型，
KeyframeInterpolationType.LINEAR
KeyframeInterpolationType.BEZIER
KeyframeInterpolationType.HOLD

app.project.item().layer().propertySpec.keyOutSpatialTangent()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyOutSpatial-Tangent(keyIndex)
描述：如果指定的属性是空间性质的（即值类型为 TwoD_SPATIAL 或 ThreeD_SPATIAL），返回指定关键帧“出”的空间切线。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：

浮点型项目的数组：

- 如果属性值类型为 isPropertyValueType.TwoD_SPATIAL，则数组包含两个浮点型项目。
- 如果属性值类型为 isPropertyValueType.ThreeD_SPATIAL,则数组包含三个浮点型值。
- 如果属性值类型不为这两个值，会发生一个意外错误。

app.project.item().layer().propertySpec.keyOutTemporalEase()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyOutTemporalEase(*keyIndex*)

描述：返回指定关键帧的“出”时间缓动。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：

关键帧缓动对象的数组

- 如果属性值类型为 isPropertyValueType.TwoD_SPATIAL，则数组包含两个对象。
- 如果属性值类型为 isPropertyValueType.ThreeD_SPATIAL,则数组包含三个对象。
- 其他任何值类型，数组只包含一个对象

app.project.item().layer().propertySpec.keyRoving()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyRoving(*keyIndex*)

描述：

如果指定关键帧是漂浮关键帧则返回 true。属性中第一个和最后一个关键帧不能被漂浮。如果强行设置，这个操作会被忽略，并且 keyRoving()持续返回 false。

如果该属性的值类型不是 TwoD_SPATIAL，也不是 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keySelected()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keySelected(*keyIndex*)

描述：如果指定的关键帧被选中则返回 true。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keySpatialAutoBezier()

属性对象方法：app.project.item().layer().*propertySpec*.keySpatialAutoBezier(*keyIndex*)

描述：

如果指定的关键帧有空间自动贝塞尔插值则返回 true。(只有 keySpatialContinuous(*keyIndex*)返回 true 时这个类型的插值才会影响到该关键帧。)

如果属性值的类型不为 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keySpatialContinuous()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keySpatial-Continuous(*keyIndex*)

描述：

如果指定的关键帧有空间线性插值则返回 true。

如果属性值的类型不为 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keyTemporalAutoBezier()

属性对象方法：

app.project.item(index).layer(index).*propertySpec*.keyTemporalAutoBezier(*keyIndex*)

描述：

如果指定的关键帧有时间自动贝塞尔插值则返回 true。

如果该关键帧插值类型在 keyInterpolationType(*keyIndex*)和 keyOutInterpolationType(*keyIndex*)都是 KeyframeInterpolationType.BEZIER 时，时间自动贝塞尔插值才会在该关键帧上生效。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keyTemporalContinuous()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.keyTemporal-Continuous(*keyIndex*)

描述：

如果指定的关键帧有时间线性插值则返回 true。

如果该关键帧插值类型在 keyInterpolationType(*keyIndex*)和 keyOutInterpolationType(*keyIndex*)都是 KeyframeInterpolationType.BEZIER 时，时间自动贝塞尔插值才会在该关键帧上生效。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：布尔型

app.project.item().layer().propertySpec.keyTime()

属性对象方法：

app.project.item(index).layer(index).*propertySpec*.keyTime(*keyIndex*)
app.project.item(index).layer(index).*propertySpec*.keyTime(*markerComment*)

描述：

检索指定的关键帧或标记并返回它们所在的时间点。

如果没有找到与参数匹配的关键帧或标记，该方法产生一个意外错误并显示一致命错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

markerComment	标记的内容（具体查看”标记值对象内容属性”）

返回：浮点型值

app.project.item().layer().propertySpec.keyValue()

属性对象方法：
app.project.item(index).layer(index).*propertySpec*.keyValue(*keyIndex*)
app.project.item(index).layer(index).*propertySpec*.keyValue(*markerComment*)
描述：
检索指定的关键帧或标记并返回它们当前值。
如果没有找到与参数匹配的关键帧或标记，该方法产生一个意外错误并显示一致命错误。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

markerComment	标记的内容（具体查看”标记值对象内容属性”）

返回：关键帧返回浮点型值，标记返回标记值对象

app.project.item().layer().propertySpec.maxValue

属性对象属性：app.project.item(index).layer(index).*propertySpec*.maxValue
描述：指定属性允许的最大值。如果 hasMax 属性值为 false，发生一个意外错误，并产生一个致命错误。
类型：浮点型，只读

app.project.item().layer().propertySpec.minValue

属性对象属性：app.project.item(index).layer(index).*propertySpec*.minValue
描述：指定属性允许的最小值。如果 hasMin 属性值为 false，发生一个意外错误，并产生一个致命错误。
类型：浮点型，只读

app.project.item().layer().propertySpec.nearestKeyIndex()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.nearestKeyIndex(*time*)
描述：返回与指定时间相邻最近的关键帧索引号。
参数：

time	指定的时间，秒数，合成的开始时间是 0

返回：整数型

app.project.item().layer().propertySpec.numKeys

属性对象属性：app.project.item(index).layer(index).*propertySpec*.numKeys
描述：指定属性的关键帧数，如果属性没有关键帧值为 0。
类型：整数型，只读

app.project.item().layer().propertySpec.propertyIndex

属性对象属性：app.project.item(index).layer(index).*propertySpec*.propertyIndex
描述：指定属性的位置索引号。第一个属性的位置索引号是 1。
类型：整数型，只读

app.project.item().layer().propertySpec.propertyValueType

属性对象属性：app.project.item(index).layer(index).propertySpec.propertyValueType
描述：
指定属性的值类型。PropertyValueType 包含每种属性中存储或检索的类型值。每种不同类型的值有不同的结构。所有属性对象存储的值类型都包含在这个属性值中。
比如，一个 3D 空间属性（类似层的位置属性）储存了一个三个浮点型项目的数组。通过这个数组给位置属性设置值，就像这样：

```
mylayer.property("position").setValue([10,20,0]);
```

作为对比，形状属性（类似层的蒙版形状）储存了一个形状对象。通过这个形状对象给形状属性设置值，就像这样：

```
var myShape = new Shape();
myShape.vertices = [[0,0],[0,100],[100,100],[100,0]];
var myMask = mylayer.property("ADBE Mask Parade").property(1);
myMask.property("ADBE Mask Shape").setValue(myShape);
```

类型：
一些列举的属性值类型，读/写

PropertyValueType.NO_VALUE	没有储存值
PropertyValueType.ThreeD_SPATIAL	3 个浮点型位置值的数组。比如，锚点值可能是 [10.0,20.2,0.0]
PropertyValueType.ThreeD	3 个浮点型量化值的数组。比如，缩放的值可能是 [100.0,20.2,0.0]
PropertyValueType.TwoD_SPATIAL	2 个浮点型位置值的数组。比如，锚点值可能是 [10.0,20.2]
PropertyValueType.TwoD	2 个浮点型量化值的数组。比如，缩放的值可能是 [100.0,20.2]

PropertyValueType.OneD	浮点型值
PropertyValueType.COLOR	4 个浮点型值的数组，范围[0.0~1.0]。比如，[0.8,0.3,0.1,1.0]
PropertyValueType.CUSTOM_VALUE	自定义属性值，比如色阶效果的直方图属性
PropertyValueType.MARKER	标记值对象，具体查看“标记对象”
PropertyValueType.LAYER_INDEX	整数，0 意味着没有层
PropertyValueType.MASK_INDEX	整数，0 意味着没有蒙版
PropertyValueType.SHAPE	形状对象，具体查看“形状对象”
PropertyValueType.TEXT_DOCUMENT	文本文档对象，具体查看“文本文档对象”

app.project.item().layer().propertySpec.removeKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.removeKey(*keyIndex*)
描述：
删除指定属性的指定关键帧。如果指定的索引号不存在，产生一个意外错误并显示一个致命错误。当关键帧被删除时，其他索引号也会相应改变。如果要删除多个关键帧，必须从索引号大的关键帧开始依次降低删除，确保剩下的关键帧索引号不会影响到删除操作。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号

返回：无

app.project.item().layer().propertySpec.selectedKeys

属性对象属性：app.project.item(index).layer(index).*propertySpec*.selectedKeys
描述：指定的属性的所有被选中的关键帧的索引号。如果没有关键帧被选中或者属性没有关键帧，返回一个空数组。
类型：整数型的数组，只读

app.project.item().layer().propertySpec.separationDimension

属性对象属性：app.project.item(index).layer(index).*propertySpec*.separationDimension
描述：对于一个分离属性，维数代表了在多维属性中的位置。第一个维数开始于 0。比如，Y 位置属性的 separtionDimension 属性值为 1，X 位置属性的值为 0。

类型：整数，只读

app.project.item().layer().propertySpec.separationLeader

属性对象属性：app.project.item(index).layer(index).*propertySpec*.separationLeader
描述：该分离属性的源多维属性。比如，如果当前属性是 Y 位置，该属性的值为位置属性。
类型：属性对象，只读

app.project.item().layer().propertySpec.setInterpolationTypeAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setInterpolationType-AtKey(*keyIndex*, *inType*,*outType*)
描述：给指定关键帧设置“入”和“出”的插值类型。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
inType	入点插值类型。一些列举的关键帧插值类型 KeyframeInterpolationType.LINEAR KeyframeInterpolationType.BEZIER KeyframeInterpolationType.HOLD
outType	可选，出点插值类型。如果没有设置则默认与入点插值类型相同。一些列举的关键帧插值类型 KeyframeInterpolationType.LINEAR KeyframeInterpolationType.BEZIER KeyframeInterpolationType.HOLD

返回：无

app.project.item().layer().propertySpec.setRovingAtKey()

属性对象方法：app.project.item(index).layer(index).propertySpec.setRovingAt-Key(*keyIndex*,*newVal*)
描述：
给指定关键帧设置浮动开启或关闭。属性的第一个和最后一个关键帧不能设置浮动。如果强行设置，操作会被忽略，并且 keyRoving()会持续返回 false。
如果属性值类型不是 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。
参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newVal	true 为开启浮动，false 为关闭浮动

返回：无

app.project.item().layer().propertySpec.setSelectedAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setSelectedAt-Key(*keyIndex*,*onOff*)

描述：选中或取消选中指定的关键帧。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
onOff	true 为选中关键帧，false 为取消选中

返回：无

app.project.item().layer().propertySpec.setSpatialAutoBezierAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setSpatialAutoBezier-AtKey(*keyIndex*,*newVal*)

描述：

给指定的关键帧设置开启空间自动贝塞尔插值或关闭。

如果属性值类型不是 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newVal	true 为开启空间自动贝塞尔插值，false 为关闭

返回：无

app.project.item().layer().propertySpec.setSpatialContinuousAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setSpatialContinuous-AtKey(*keyIndex*,*newVal*)

描述：

给指定的关键帧设置开启空间线性插值或关闭。

如果属性值类型不是 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newVal	true 为开启空间线性插值，false 为关闭

返回：无

app.project.item().layer().propertySpec.setSpatialTangentsAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setSpatialTangents-AtKey(*keyIndex*, *inTangent*, *outTangent*)

描述：

给指定的关键帧设置入点和出点的切线矢量。

如果属性值类型不是 TwoD_SPATIAL 和 ThreeD_SPATIAL，会产生一个意外错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
inTangent	入点切线矢量，2 个或 3 个浮点型的数组 ● 如果属性值是 PropertyValueType.TwoD_SPATIAL，该数组包含两个值。 ● 如果属性值是 PropertyValueType.ThreeD_SPATIAL，该数组包含三个值。
outTangent	可选，出点切线矢量，如果没有设置则默认与入点相同，2 个或 3 个浮点型的数组 ● 如果属性值是 PropertyValueType.TwoD_SPATIAL，该数组包含两个值。 ● 如果属性值是 PropertyValueType.ThreeD_SPATIAL，该数组包含三个值。

返回：无

app.project.item().layer().propertySpec.setTemporalAutoBezierAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setTemporalAutoBezier-AtKey(*keyIndex*, *newVal*)

描述：

给指定的关键帧设置开启时间自动贝塞尔插值或关闭。当开启时，只有在 keySpatialContinuous(*keyIndex*)为 true 时才会有效。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newVal	true 为开启时间自动贝塞尔插值，false 为关闭

返回：无

app.project.item().layer().propertySpec.setTemporalEaseAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setTemporalEase-AtKey(*keyIndex*, *inTemporalEase*, *outTemporalEase*)

描述：给指定的关键帧设置入点和出点的时间缓动。具体查看标记对象。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
inTemporalEase	入点时间缓动，1 个、2 个或 3 个浮点型的数组 ● 如果属性值是 PropertyValueType.TwoD_SPATIAL，该数组包含两个对象。 ● 如果属性值是 PropertyValueType.ThreeD_SPATIAL，该数组包含三个对象。 ● 如果属性值为其他值，该数组包含一个对象。
outTemporalEase	可选，出点时间缓动，如果没有设置则默认与入点相同，1 个、2 个或 3 个浮点型的数组 ● 如果属性值是 PropertyValueType.TwoD_SPATIAL，该数组包含两个值。 ● 如果属性值是 PropertyValueType.ThreeD_SPATIAL，该数组包含三个值。 ● 如果属性值为其他值，该数组包含一个对象。

返回：无

app.project.item().layer().propertySpec.setValue()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setValue(*newValue*)

描述：

给没有关键帧的属性设置一个静态值。

如果指定的属性有关键帧，这个方法会产生一个意外错误并显示一个致命错误。给带关键帧的属性设置值，使用 setValueAtTime()和 setValueAtKey()方法。

参数：

newValue	给属性设置的关键帧值。

返回：无

app.project.item().layer().propertySpec.setValueAtKey()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setValue-AtKey(keyIndex,*newValue*)

描述：

检索一个指定的关键帧并设置值。

如果指定的属性没有关键帧或没有指定索引号的关键帧，该方法会产生一个意外错误并显示一个致命错误。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newValue	给属性设置的关键帧值。

返回：无

app.project.item().layer().propertySpec.setValueAtTime()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setValue-AtTime(keyIndex,*newValue*)

描述：

在指定时间给一个关键帧设置一个值。如果在指定的时间不存在关键帧，则给指定的属性创建一个新关键帧。

参数：

keyIndex	关键帧的索引号，整数型，范围[1~关键帧总数]，addKey 和 nearestKeyIndex 方法可以返回关键帧索引号
newValue	给属性设置的关键帧值。

返回：无

app.project.item().layer().propertySpec.setValueAtTimes()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.setValue-AtTimes(times,*newValue*)

描述：

在指定时间组给一个关键帧设置一个值。如果在指定的时间不存在关键帧，则给指定的属性创建一个新关键帧。

参数：

times	时间的数组，每个时间都是浮点型值。合成开始时间为 0

newValue	给属性设置的关键帧值

返回：无

app.project.item().layer().proeprtySpec.value

属性对象属性：app.project.item(index).layer(index).*proeprtySpec*.value

描述：

当前时间指定属性的值。

- 如果 expressionEnabled 值为 true，返回表达式计算后的值。
- 如果当前时间有关键帧，返回关键帧的值。
- 否则，返回静态值。

返回值的类型取决于属性值类型。

类型：符合属性的值类型，只读

app.project.item().layer().propertySpec.valueAtTime()

属性对象方法：app.project.item(index).layer(index).*propertySpec*.value-AtTime(*time,preExpression*)

描述：

指定时间的计算表达式前后的属性值。

注意返回值的类型是不确定的，可以是不同的类型，主要取决于被计算的属性。

参数：

time	指定的时间，秒数，浮点型值，合成开始时间为 0
preExpression	如果该属性有表达式并开启，返回指定时间没有计算表达式的值。如果表达式关闭，返回指定时间计算表达式的值。 如果该属性没有表达式则会忽略。

返回：符合属性的值类型

属性基础对象

app.project.item().layer().*perpertySpec*

描述：

通过层中的属性的名字访问该属性，使用各种表达式语法，如同通过 AE 参数控制。比如，下面是在效果组中访问属性的所有方法。

```
var effect1 = app.project.item(1).layer(1).effect("Add Grain")("Viewing Mode");
var effect1again = app.project.item(1).layer(1).effect.addGrain.viewingMode;
var effect1againtoo = app.project.item(1).layer(1)("Effects").addGrain.viewingMode;
var effect1againtoo2 = app.project.item(1).layer(1)("Effects")("Add Grain")("Viewing Mode");
```

具体查看“属性组对象 property()方法”。

- 属性基础对象是属性对象和属性组对象的基础类，属性基础对象的属性和方法可以被属性对象和属性组对象调用。具体查看“属性对象”和“属性组对象”

引用失效：

当发生某些事，比如改变对象，引用会失效，脚本引用已改变的对象会产生一个致命错误。在简单的情况下这很容易理解。比如，如果删除一个对象，对于已删除对象的引用会产生一个警告“无效对象”：

```
var layer1 = app.project.item(1).layer(1);
layer1.remove();
alert(layer1.name); // 对已删除的对象引用无效
```

同样的，如果引用一个已经删除的 AE 属性对象也会产生一个警告：

```
var layer1 = app.project.item(1).layer(1);
var layer1position = layer1.transform.position;
layer1.remove();
alert(layer1position.value); // 已删除对象的属性引用无效
```

一个稍微复杂的情况是一个属性从一个属性组中被删除。在这个情况下，如果之后引用组中该项目或其他项目，AE 会产生一个“无效对象”致命错误，因为它们的索引位置改变了。比如：

```
var effect1 = app.project.item(1).layer(1).effect(1);
var effect2 = app.project.item(1).layer(1).effect(2);
var effect2param = app.project.item(1).layer(1).effect(2).blendWithOriginal;
effect1.remove();
alert(effect2.name); // 无效引用，因为组的索引位置改变了
```

属性：

属性	描述
name	属性的名字

matchName	属性的特殊名字，用作创建一个唯一的命名路径
propertyIndex	在其父类内该属性的索引
propertyDepth	在该属性和容纳层之间的父类的层级数
propertyType	该属性的类型
parentProperty	该属性的当前父类
isModified	值为 true 时，属性在创建后值已经被修改过
canSetEnabled	值为 true 时，在 UI 界面中该属性前显示一个小眼睛（或 fx）图标
enabled	值为 true 时，该属性已开启
active	值为 true 时，该属性被激活
elided	值为 true 时，该属性没有显示在 UI 界面中
isEffect	值为 true 时，该属性是一个效果
isMask	值为 true 时，该属性是一个蒙版
selected	值为 true 时，该属性被选中

方法：

方法	描述
propertyGroup()	取得该属性的父类
remove()	从工程中删除
moveTo()	在其父类内部移动该属性到一个新位置
duplicate()	复制这个属性对象

app.project.item().layer().porpertySpec.active

属性基础对象属性：app.project.item(index).layer(index).*porpertySpec*.active
描述：值为 true 时，该属性被激活。对于一个层，等同于小眼睛图标的设置，如果当前时间在层的入点和出点之间。对于一个效果和所有属性，等同于 enabled 属性，除非那个效果或属性是只读的。
类型：布尔型，只读

app.project.item().layer().propertySpec.matchName

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.matchName
描述：属性的一个特殊名称，用来创建一个唯一的命名路径。这个匹配名称不会显示，但是你可以在脚本中引用它。所有的属性都有一个唯一的匹配名称。匹配名称不会随着版本变化导致的显示名称（name 属性值）变化和任何 AE 的变化而变化。与显示名称不同，匹配名称不是局部化的。
并且索引组可能不会有显示名称，但是一定会有一个匹配名称。(索引组有 PropertyType.INDEXED_GROUP 类型，具体查看”属性基础对象的 propertyType 属性”)。
类型：字符串型，只读

app.project.item().layer().propertySpec.moveTo()

属性基础对象方法：app.project.item(index).layer(index).*propertySpec*.moveTo(*newIndex*)
描述：
将该属性移动到父属性组内的另一个位置。
这个方法只有对子索引组有效，如果不是子索引组或索引值无效，这个方法会生成一个意外错误并显示一个致命错误。索引组有 PropertyType.INDEXED_GROUP 类型，具体查看”属性基础对象的 propertyType 属性”)。
注意：使用这个方法对同一索引组内的其他已经被引用的子索无效。比如，如果在一个层内有三个效果，每个效果指定了不同的变量，移动一个效果会使得所有变量的引用都无效。你需要重新给他们指定变量。
参数：

newIndex	整数型，在该属性的组内的新索引位置。

返回：无

app.project.item().layer().propertySpec.name

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.name
描述：
属性的显示名称。
如果该属性不是索引组的子类那么设置名称会发生一个致命错误。(即，这是一个类型为 PropertyType.INDEXED_GROUP 的属性组)。
类型：字符串型，读/写，如果不是索引组的子类，为只读。

app.project.item().layer().propertySpec.parentProperty

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.parentProperty
描述：该属性的当前父类属性组，如果该属性的父类属性组时一个图层则返回 null。
类型：属性组对象或 null，只读

app.project.item().layer().propertySpec.propertyDepth

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.propertyDepth
描述：在该属性和容纳层之间的父类属性组的层数。图层的值为 0。
类型：整数型，只读

app.project.item().layer().propertySpec.propertyGroup()

属性基础对象方法：
app.project.item(index).layer(index).*propertySpec*.propertyGroup()
app.project.item(index).layer(index).*propertySpec*.propertyGroup(*countUp*)
描述：取得父子级中指定层级的属性的上级属性的属性组对象。
参数：

countUp	可选。在父子级中的上层层级数。整数型，范围[1~propertyDepth]，默认为 1，取得当前的父级。

返回：属性组对象，如果父级为容纳层则返回 null

app.project.item().layer().propertySpec.propertyIndex

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.propertyIndex
描述：如果该属性是索引组的子级（即，这是一个类型为 PropertyType.INDEXED_GROUP 的属性组），则返回该属性在其父级属性组内的位置索引。
类型：整数型，只读

app.project.item().layer().propertySpec.propertyType

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.propertyType
描述：该属性的类型。
类型：
一些列举的属性类型值，读/写

PropertyType.PROPERTY	一个单独的属性，比如位置

PropertyType.INDEXED_GROUP	一个子级含有可编辑名称和索引的属性组。效果和蒙版都是这个类型。比如，图层的蒙版属性通过索引号引用一个单独蒙版的变量数。
PropertyType.NAMED_GROUP	一个不可以编辑子级名称的属性组。图层就是这个类型。

app.project.item().layer().propertySpec.remove()

属性基础对象方法：app.project.item(index).layer(index).propertySpec.remove()
描述：
将该属性从它的父级属性组删除。如果该属性就是一个属性组，会同时删除它的子属性。
此方法只有对索引组的子级有效。如果不是或索引值无效，此方法会产生一个意外错误并显示一个致命错误。(索引组类型为 PropertyType.INDEXED_GROUP)
此方法可以被文字动画属性调用(即，任何已经添加到文字上的动画)。
参数：无
返回：无

app.project.item().layer().propertySpec.selected

属性基础对象属性：app.project.item(index).layer(index).*propertySpec*.selected
描述：
值为 true 时，该属性被选中。设置为 true 则选中该属性，设置为否则取消选中。
为大量的属性设置选中可能会降低系统性能。使用合成对象或层对象的 selectedProperties 属性来查看全系列的合成或层选中属性。
类型：布尔型，读/写

属性组对象

app.project.item(index).layer(index).*propertyGroupSpec*

描述：

属性组对象代表了属性的一个组。它可以包含属性对象和其他属性组对象。属性组可以嵌套父子层级结构，比如自上而下是层对象和单一属性对象，或者第三个蒙版的蒙版羽化属性。使用属性基础对象属性和方法横贯组层级结构。

关于访问属性和属性组的例子，具体查看“属性基础对象”。

- 属性组对象是属性基础对象的子类。所有属性基础对象的属性和方法都可以被属性组对象调用
- 属性组对象时蒙版属性组对象的基础类。属性组对象的属性和方法都可以被蒙版属性对象调用

属性：

属性	描述
numProperties	组内索引化属性的数量

方法：

方法	描述
property()	取得一个组或一个子属性
canAddProperty()	检索一个属性是否可以被添加进组
addProperty()	向组中添加一个属性

app.project.item().layer().propertyGroupSpec.addProperty()

属性组对象方法：app.project.item(index).layer(index).*propertyGroupSpec*.addProperty()

描述：

创建并返回一个指定名称的属性基础对象，并添加到该组内。

通常情况下，只能添加属性到索引化组（一个类型为 PropertyType.INDEXED_GROUP 的属性组）。只有在文字动画属性情况下例外，可以添加属性到名称组（一个类型为 PropertyType.NAMED_GROUP 的属性组）。

如果该方法不能通过指定名称创建属性，会生成一个意外错误。在使用该方法之前调用 canAddProperty 还检查是否能够在属性组内添加指定属性。

参数：

name	被添加的属性的显示名称或匹配名称。 支持的名称有下面这些： ● 任意一个可以通过 UI 添加的属性的匹配名称。比如，“ADBE Mask Atom”、“ADBE Paint Atom”、“ADBE Text Position”。 ● 添加到 ADBE Mask Parade 时：“ADBE Mask Atom”、“Mask” ● 添加到 ADBE Effect Parade 时，任意一个效果的匹配名称，比如 “ADBE Bulge”、“ADBE Glo2”、“APC Vegas”。 ● 任意效果的显示名称，比如 “Bulge”、“Glow”、“Vegas”。 ● 对于文本动画，“ADBE Text Animator” ● 对于选择器，范围选择器名称为 “ADBE Text Selector”，摆动选择器名称为 “ADBE Text Wiggly Selector”，表达式选择器名称为 “ADBE Text Expressible Selector”

返回：属性基础对象

app.project.item().layer().propertyGroupSpec.canAddProperty()

属性组对象方法：app.project.item(index).layer(index).*propertyGroupSpec*.canAdd-Property(*name*)

描述：

如果指定名称的属性可以添加到该属性组则返回 true。比如，只能将蒙版添加到蒙版组。唯一合法的输入参数时 “mask” 或 “ADBE Mask Atom”。

```
maskGroup.canAddProperty("mask"); //返回true
maskGroup.canAddProperty("ADBE Mask Atom"); //返回true
maskGroup.canAddProperty("blend"); // 返回false
```

参数：

name	被检查的属性的显示名称或匹配名称。

返回：布尔型

app.project.item().layer().proeprtyGroupSpec.numProperties

属性组对象属性：app.project.item(index).layer(index).*proeprtyGroupSpec*.numProperties

描述：

在该属性组内的索引化属性的数量。

对于层，该方法返回值为 3，相当于在层内索引化组的数量：蒙版、效果、动态追踪组。然而，层可能也会有其他只有名称可见的属性。

类型：整数型，只读

app.project.item().layer().propertyGroupSpec.proeprty()

属性组对象方法：

app.project.item(index).layer(index).*propertyGroupSpec*.proeprty(*index*)

app.project.item(index).layer(index).*propertyGroupSpec*.proeprty(*name*)

描述：

检索并返回该组内指定名称或索引号的子属性。

指定名称可以使用与表达式同样的用法。下面这些用法都可以使用并效果相同：

```
mylayer.position
mylayer("position")
mylayer.property("position")
mylayer(1)
mylayer.property(1)
```

一些层的属性，比如位置，只能通过名称访问。

当使用名称检索多层之下的属性时，必须不止一次调用此方法。比如，下面调用了两层之下属性，并返回在蒙版组内第一个蒙版。

```
myLayer.property("ADBE Masks").property(1)
```

参数：

index	子属性在的索引号，在这里是一个索引化属性组。整数型，范围[0~numProperties]

name	子属性的名称，它可能是： ● 任意匹配名称 ● 任何用“括号风格”语法表达的名称，包括显示名称或紧凑的英语名称 ● 任何用“中间层风格”语法表达的名称 更多支持的属性名称，查看下面的表格

返回：属性基础对象，如果指定名称的子属性没有找到则返回 null。

通过名称访问属性：

访问任意层	● “ADBE Mask Parade”或“Masks” ● “ADBE Effect Parade”或“Effects” ● “ADBE MTrackers”或“Motion Trackers”
访问任意媒体层	● “Anchor Point”或“anchorPoint” ● “Position”或“position” ● “Scale”或“scale” ● “Rotation”或“rotation” ● “Z Rotation”或“zRotation”或“Rotation Z”或“rotationZ” ● “Opacity”或“opactity” ● “Marker”或“marker”

访问任意没有静止源的媒体层	● "Time Remap" 或 "timeRemapEnabled"
访问带有音频通道的媒体层	● "Audio Levels" 或 "audioLevels"
访问摄像机层	● "Zoom" 或 "zoom" ● "Depth of Field" 或 "depthOfField" ● "Focus Distance" 或 "focusDistance" ● "Aperture" 或 "aperture" ● "Blur Level" 或 "blueLevel"
访问灯光层	● "Intensity" 或 "intensity" ● "Color" 或 "color" ● "Cone Angle" 或 "coneAngle" ● "Cone Feather" 或 "coneFeather" ● "Shadow Darkness" 或 "shadowDarkness" ● "Shadow Diffusion" 或 "shadowDiffusion" ● "Casts Shadows" 或 "castsShadow"
访问 3D 层	● "Accepts Shadows" 或 "acceptsShadow" ● "Accepts Lights" 或 "acceptsLights" ● "Ambient" 或 "ambient" ● "Diffuse" 或 "diffuse" ● "Specular" 或 "specular"（这是镜面强度属性） ● "Shininess" 或 "shininess"（这是镜面反光度） ● "Casts Shadows" 或 "castsShadows" ● "Light Transmission" 或 "lightTransmission" ● "Metal" 或 "metal"
访问摄像机层、灯光层或 3D 层	● "X Rotation" 或 "xRotation" 或 "Rotation X" 或 "rotationX" ● "Y Rotation" 或 "yRotation" 或 "Rotation Y" 或 "rotationY" ● "Orientation" 或 "orientation"
访问文本层	● "Source Text" 或 "sourceText" 或 "Text" 或 "text"
访问属性组 "ADBE Mask Parade"	● "ADBE Mask Atom"
访问属性组 "ADBE Mask Atom"	● "ADBE Mask Shape" 或 "maskShape" 或 "maskPath" ● "ADBE Mask Feather" 或 "maskFeather" ● "ADBE Mask Opacity" 或 "maskOpacity" ● "ADBE Mask Offset" 或 "maskOffset"

举例：

1. 如果一个名为“myLayer”的图层有一个方框模糊效果，可以使用下面任意一种方式检索：

```
myLayer.property("Effects").property("Box Blur");
myLayer.property("Effects").property("boxBlur");
myLayer.property("Effects").property("ADBE Box Blur");
```

2. 如果一个名为“myLayer”的图层有一个名为“Mask 1”的蒙版，可以使用下面的方式检索：

```
myLayer.property("Masks").property("Mask 1");
```

3. 从凸出效果中检索凸出中心值，可以使用下面任意一种方式：

```
myLayer.property("Effects").property("Bulge").property("Bulge Center");
myLayer.property("Effects").property("Bulge").property("bulgeCenter");
```

渲染队列对象

app.project.renderQueue

描述：

渲染队列对象代表了渲染自动化过程，可通过某一 AE 工程的渲染队列面板使用的数据和功能。属性提供了渲染队列的项目和渲染状态的访问接口。方法可以开始、暂停、停止渲染进程。

渲染队列项目对象提供了要渲染项目的设定的接口。具体查看 “渲染队列项目对象”。

属性：

属性	描述
rendering	值为 true 时，正在渲染。
numItems	渲染队列项目的总数量
items	渲染队列中项目的收集

方法：

方法	描述
showWindow()	显示或隐藏渲染队列面板
render()	开始渲染进程，渲染未完成之前不返回参数
pauseRendering()	暂停或重新开始渲染
stopRendering()	停止渲染
item()	在收集数组中取得一个渲染队列项目

app.project.renderQueue.item()

渲染队列对象方法：app.project.renderQueue.item(*index*)

描述：在项目收集对象中取得指定的项目。

参数：

index	项目的位置索引。整数型，范围[0~numItems]

返回：渲染队列项目对象

app.project.renderQueue.items

渲染队列对象属性：app.project.renderQueue.items
描述：包含渲染队列所有项目的收集对象。具体查看 “渲染队列项目对象”
类型：渲染队列项目收集对象，只读

app.project.renderQueue.numItems

渲染队列对象属性：app.project.renderQueue.numItems
描述：渲染队列中所有项目的总数。
类型：整数型，只读

app.project.renderQueue.pauseRendering()

渲染队列对象方法：app.project.renderQueue.pauseRendering(*pause*)
描述：暂停当前渲染进程，或继续暂停的渲染进程。与在渲染队列窗口点击暂停操作相同。可以从 onStatusChanged 或 onError 回调函数调用该方法。具体查看 “渲染队列项目 onStatusChanged 属性” 和 “应用对象 onError 属性”。
参数：

pause	true 为暂停当前渲染进程，false 为继续暂停的进程

返回：无

app.project.renderQueue.render()

渲染队列对象方法：app.project.renderQueue.render()
描述：
开始渲染进程。等同于在渲染队列面板上点击渲染按钮。这个方法在渲染完成之前不会返回值。使用 onError 和 onStatusChanged 回调函数调用 pauseRendering()和 stopRendering()方法来暂停或停止渲染进程。

- 为了在渲染进程中出现错误作出响应，定义了一个回调函数 app.onError，具体查看 “app.onError”。
- 为了在渲染进程中某一项目的环境发生改变作出响应，定义了回调函数 RenderQueueItem.onStatusChanged,具体查看 “渲染队列项目 onStatusChanged 属性”。

参数：无
返回：无

app.project,renderQueue,rendering

渲染队列对象属性：app.project,renderQueue,rendering
描述：值为 true 时，渲染进程正在进行或暂停，值为 false 时，渲染停止。
类型：布尔型，只读

app.project.renderQueue.stopRendering()

渲染队列对象方法：app.project.renderQueue.stopRendering()
描述：停止渲染进程。等同于在渲染时点击渲染队列面板的停止按钮。可以从 onStatusChanged 或 onError 回调函数中调用该方法。
参数：无
返回：无

渲染队列项目对象

app.project.renderQueue.item(index)

描述：渲染队列项目对象代表了渲染队列中某一个项目。该对象提供了被渲染项目设置的访问接口。使用渲染队列项目收集对象将一个合成添加到渲染队列中来创建该对象。具体查看 “渲染队列项目收集 add()方法”。

属性：

属性	描述
numOutputModules	输出模块分配项目的总数
render	值为 true 时，当队列开始时开始渲染此项目
startTime	该项目开始渲染的时间
elapsedSeconds	该项目当前渲染时长
timeSpanStart	被渲染的合成开始时间
timeSpanDuration	被渲染的合成持续时长
skipFrames	渲染该项目时跳过的帧数
comp	该项目要被渲染的合成
outputModules	该项目的输出模块收集对象
templates	一系列渲染设置模板
status	该项目的当前渲染状态
onStatusChanged	当该项目的环境改变时渲染进程调用的一个回调函数
logType	该项目的记录类型

方法：

方法	描述
outputModule()	取得该项目的输出模块
remove()	从渲染队列中删除该项目

saveAsTemplate()	保存为新的渲染设置模板
applyTemplate()	应用渲染设置模板
duplicate()	复制该项目

app.project.renderQueue.item().applyTemplate()

渲染队列项目对象方法：app.project.renderQueue.item(index).applyTemplate(*templateName*)
描述：给该项目应用一个渲染设置模板。
参数：

templateName	包含要使用的模板名称的字符串

返回：无

app.project.renderQueue.item().comp

渲染队列项目对象属性：app.project.renderQueue.item(index).comp
描述：被该渲染队列项目渲染的合成。要删除这个合成，比如删除该渲染队列并创建一个新的。
类型：合成项目对象，只读

app.project.renderQueue.item().duplicate()

渲染队列项目对象方法：app.project.renderQueue.item(*index*).duplicate()
描述：创建一个该项目的复制并添加到此渲染队列中。
注意：复制一个状态为 "Done" 的项目，新的项目状态为 "Queued"。
参数：无
返回：渲染队列项目对象

app.project.renderQueue.item().elapsedSeconds

渲染队列项目对象属性：app.project.renderQueue.item(index).elapsedSeconds
描述：该项目已渲染的秒数。
类型：整数型，如果项目还没渲染则返回 null，只读

app.project.renderQueue.item().logType

渲染队列项目对象属性：app.project.renderQueue.item(index).logType

描述：该项目的日志类型，选择项目渲染时哪些事件需要被记录。

类型：

一些列举的日志类型值，读/写

LogType.ERRORS_ONLY
LogType.ERRORS_AND_SETTINGS
LogType.ERRORS_AND_PER_FRAME_INFO

app.project.renderQueue.item().numOutputModules

渲染队列项目对象属性：app.project.renderQueue.item(index).numOutputModules

描述：分配给该项目的输出模块的数量。

类型：整数型，只读

app.project.renderQueue.item().onStatusChanged

渲染队列项目对象属性：app.project.renderQueue.item(index).onStatusChanged

描述：

当 RenderQueueItem.status 属性改变时调用的回调函数的名称。

你不能在渲染进行或暂停过程中对渲染队列项目或应用做任何改动。然而,可以使用此回调函数暂停或停止渲染进程。参考查看 “渲染队列 pauseRendering()方法” 和 “渲染队列 stopRendering()方法”。

也可以查看 “应用对象 onError 属性”。

类型：包含函数名称的字符串，如果没有分配函数则返回 null。

举例：

```
function myStatusChanged() {
  alert(app.project.renderQueue.item(1).status)
}
app.project.renderQueue.item(1).onStatusChanged = myStatusChanged();
app.project.renderQueue.item(1).render = false; //改变状态并显示对话框
```

app.project.renderQueue.item().outputModules

渲染队列项目对象属性：app.project.renderQueue.item(index).outputModules

描述：该项目的输出模块收集对象。

类型：输出模块收集对象，只读

app.project.renderQueue.item().outputModule()

渲染队列项目对象方法：app.project.renderQueue.item(index).outputModule(index)

描述：取得指定索引位置的输出模块。

参数：

index	输出模块的位置索引。整数型，范围[1~numOutputModules]

返回：输出模块对象

app.project.renderQueue.item().remove

渲染队列项目对象方法：app.project.renderQueue.item(index).remove

描述：从渲染队列中删除该项目。

参数：无

返回：无

app.project.renderQueue.item().render

渲染队列项目对象方法：app.project.renderQueue.item(index).render

描述：值为 true 时，当渲染队列开始则该项目会被渲染。当设置值为 true 时，RenderQueueItem.status 的值设置为 RQItemStatus.QUEUED。当设置为 false 时，status 值被设置为 RQItemStatus.UNQUEUED。

类型：布尔型，读/写

app.project.renderQueue.item().saveAsTemolate()

渲染队列项目对象方法：app.project.renderQueue.item(index).saveAsTemolate(*name*)

描述：将该项目的当前渲染设置保存为新模板并指定名称。

参数：

name	包含新模板名称的字符串。

返回：无

app.project.renderQueue.item().skipFrames

渲染队列项目对象属性：app.project.renderQueue.item(index).skipFrames

描述：

当渲染项目时跳过的帧数量。使用该属性来做测试渲染，相比全部渲染更快。

值为 0 表示不跳过帧，结果与渲染所有帧相同。值为 1 表示每隔一帧跳过一次，与 “rendering on twos” 操作相同。更高的值跳过的帧数更多。

总时间长度保持不变。比如，如果跳过的值为 1，一个序列输出一半帧，每帧为两倍持续时长。

类型：整数，范围[0~99]，读/写

app.project.renderQueue.item().status

渲染队列项目对象属性：app.project.renderQueue.item(index).status
描述：该项目当前渲染状态。
类型：
一些列举的渲染项目状态值，只读

RQItemStatus.WILL_CONTINUE	渲染进程被暂停
RQItemStatus.NEEDS_OUTPUT	项目缺少有效的输出路径
RQItemStatus.UNQUEUED	项目被列在渲染队列面板中，但是合成还没有准备好渲染
RQItemStatus.QUEUED	合成准备渲染
RQItemStatus.RENDERING	合成正在渲染
RQItemStatus.USER_STOPPED	渲染进程被用户或脚本停止
RQItemStatus.ERR_STOPPED	渲染进程因为致命错误而停止
RQItemStatus.DONE	渲染完成

app.project.renderQueue.item().templates

渲染队列项目对象属性：app.project.renderQueue.item(index).templates
描述：该项目中所有可用的渲染设置模板的名称，参考查看 “渲染队列项目对象 saveAsTemplate() 方法”。
类型：字符串型的数组，只读

app.project.renderQueue.item().timeSpanDuration

渲染队列项目对象属性：app.project.renderQueue.item(index).timeSpanDuration
描述：被渲染的合成的时长秒数。这个持续时长取决于结束时间减去开始时间。设置该属性值等同于在渲染设置对话框中设置自定义结束时间。
类型：浮点型，读/写

app.project.renderQueue.item().timeSpanStart

渲染队列项目对象属性：app.project.renderQueue.item(index).timeSpanStart
描述：合成在渲染开始时的时间，秒数。设置该属性值等同于在渲染设置对话框中设置自定义开始时间。
类型：浮点型，读/写

渲染队列项目收集对象

app.project.renderQueue.items

描述：

渲染队列项目收集对象包含了工程中所有渲染队列的项目，如同渲染队列面板中显示的一样。该收集对象提供了渲染队列项目对象的访问接口，并且允许你从合成中创建项目对象。收集对象中第一个渲染队列项目对象索引位置为 1。参考查看 “渲染队列项目对象”。

- 渲染队列项目收集对象是收集对象的子类。所有收集对象的方法和属性都可以被渲染队列项目收集对象调用。参考查看 “收集对象”。

方法：

方法	描述
add()	添加一个合成到渲染队列

app.project.renderQueue.items.add()

渲染队列项目收集对象方法：app.project.renderQueue.items.add(*comp*)

描述：添加一个合成到渲染队列中，并创建一个渲染队列项目。

参数：

comp	要被添加的合成的合成项目对象

返回：渲染队列项目对象

设置对象

描述：

设置对象提供了一个非常方便的方法来管理脚本的设置。该设置对象被保存 zaiAE 首选项文件中并在应用会话之间保持持久性。设置被文件内的章节和键名识别，每个键名都与一个值相关联。在首选项文件中，章节名称用括号和引号括起来，键名列在章节名称下并用引号括起来。所有的值都是字符串。

你可以使用该对象创建新的设置，也可以访问已存在的设置。

方法：

方法	描述
saveSetting()	保存设置的默认值
getSetting()	检索一个设置值
haveSetting()	检索是否分配了指定的设置

app.settings.getSetting()

设置对象方法：app.*settings*.getSetting(*sectionName*,*keyName*)

描述：从首选项文件中检索一个脚本首选项项目值。

参数：

sectionName	一个包含设置章节名称的字符串
keyName	一个包含设置项目键名的字符串

返回：字符串

举例：

如果在 "Eraser - Paint Settings" 章节中保存了 "Aligned Clone" 的键名，你可以使用下面这段代码检索该值：

```
var n = app.settings.getSetting("Eraser - Paint Settings", "Aligned Clone");
alert("这个设置是" + n);
```

app.settings.haveSetting()

设置对象方法：app.*settings*.haveSetting(*sectionName*,*keyName*)

描述：当指定的脚本首选项项目存在并有一个值则返回 true。

参数：

sectionName	一个包含设置章节名称的字符串
keyName	一个包含设置项目键名的字符串

返回：布尔型

app.settings.saveSetting()

设置对象方法：app.*settings*.saveSetting(*sectionName,keyName,value*)

描述：给脚本首选项项目设置一个默认值

参数：

sectionName	一个包含设置章节名称的字符串
keyName	一个包含设置项目键名的字符串
value	一个包含新值的字符串

返回：无

形状对象

app.project.item(index).layer(index).property(index).property(“mask-Shape”).value
描述：
形状对象封装了形状图层中图层的描述信息，或者是一个蒙版的边缘线形状。这是 AE 属性“蒙版路径”的值，同时也是形状图层的“路径”值。使用 new Shape()构造函数创建一个新的空的形状对象，再设置各个属性去定义该形状。
形状有一些锚点或者说顶点，每个锚点都有一对方向手柄或者说切线顶点。一个切线顶点（在非贝塞尔曲线蒙版中）决定了线绘制或从锚点出发的方向。形状中每个顶点都有一个相关联的入点切线矢量和出点切线矢量。
切线值是一对与相关顶点有关的一对 x,y 坐标。比如，一个[-1,-1]的切线位于顶点的上方和左侧，并且有 45°斜率，无论顶点的实际位置在哪。手柄越长影响越大。比如，一个入点形状片段停留在[-1,-1]入点切线比在[-2,-2]入点切线离顶点更近，即使它们朝向顶点同一个方向。
如果形状不是闭合的，第一个顶点的 inTangent 和最后一个点的 outTangent 会被忽略。如果形状是闭合的，这两个向量指定了最后连接点片段的方向手柄从最后顶点引出并回到第一个顶点处。
自动曲线蒙版会自动计算形状的切线。(参考查看”蒙版属性组 rotoBezier 属性”)。如果形状使用了自动曲线蒙版，切线值会被忽略。这意味着，对于自动曲线蒙版，你只能为构造的形状设置 vertices 属性，并且设置 inTangents 和 outTangents 都为 null。当你访问新形状时，它的切线值会被自动计算出的切线值代替。
对于闭合蒙版形状，可变宽度的蒙版羽化点可以存在于蒙版路径的任何地方。羽化点是蒙版路径属性的一部分。通过蒙版路径显示的片段的数值来引用指定的羽化点。
注意：羽化点在蒙版中以数组列举，顺序和创建顺序一致。
举例：
创建一个方形蒙版。
这个方形是有 4 个顶点的封闭形状。连接直线片段的 inTangents 和 outTangents 的值为 0，不需要明确设置，这是默认值。

```
var myShape = new Shape();
myShape.vertices = [[0,0], [0,100], [100,100], [100,0]];
myShape.closed = true;
```

创建一个 U 型蒙版
一个“U”型非闭合形状，与方形有一样的 4 个顶点。

```
var myShape = new Shape();
myShape.vertices = [[0,0], [0,100], [100,100], [100,0]];
myShape.closed = false;
```

创建一个椭圆形。

有 4 个顶点的封闭图形，并带有 inTangent 和 outTangent 值。

```
var myShape = new Shape();
myShape.vertices = [[300,50],[200,150],[300,250],[400,150]];
myShape.inTangents = [[55.23,0],[0,-55.23],[-55.23,0],[0,55.23]];
myShape.outTangents = [[-55.23,0],[0,55.23],[55.23,0],[0,-55.23]];
myShape.closed = true;
```

创建一个方形蒙版并带有两个羽化点。

一个带有两个羽化点的大正方形，一个羽化点靠近第二个蒙版片段（下底边）的左端，半径为 30px，另一个在第三个蒙版片段（右边）中间，半径为 100px。

```
var myShape = new Shape();
myShape.vertices = [
  [100, 100],
  [100, 400],
  [400, 400],
  [400, 100]
]; // 片段逆时针绘制
myShape.closed = true;
myShape.featherSegLocs = [1, 2]; // 片段开始数是 0，所以第二个片段是 1
myShape.featherRelSegLocs = [0.15, 0.5]; // 0.15 靠近正方形的左下角
myShape.featherRadii = [30, 100]; // 第二个羽化点（在右边片段上）有一个较大的半径
```

属性：

属性	描述
closed	值为 true 时，形状时一个封闭曲线
vertices	形状的锚点
inTangents	切向量进入形状顶点
outTangents	从形状顶点引出的切向量
featherSegLocs	蒙版路径片段（在顶点之间的蒙版路径章节），包含每个羽化点
featherRelSegLocs	蒙版路径片段上的羽化点的相对位置
featherRadii	每个羽化点的羽化数（半径）
featherInterps	每个羽化点的羽化半径插值类型
featherTensions	每个羽化点的羽化张力
featherTypes	每个羽化点的方向（向内或向外）
featherRelCornerAngles	在蒙版路径拐角处的一段曲线外部羽化范围的每个点中，任意两个点的相对角度

shapeObject.value.closed

形状对象属性：*shapeObject*.value.closed
描述：值为 true 时，第一个和最后一个顶点相连接，该形状是一个封闭曲线。值为 false 时，没有绘制封闭片段。
类型：布尔型，读/写

shapeObject.value.featherRadii

形状对象属性：*shapeObject*.value.featherRadii
描述：包含每个羽化点半径的数组（羽化数）；内部羽化点值为负数。
注意：存储在数组中的值按着羽化点创建的顺序排序。
类型：浮点型的数组，读/写

shapeObject.value.featherRelCornerAngles

形状对象属性：*shapeObject*.value.featherRelCornerAngles
描述：一个在蒙版路径拐角处的曲线外部羽化范围点中任意每两个点的相对角度百分比的数组。不在拐角处的角度值为 0%。
注意：存储在数组中的值按着羽化点创建的顺序排序。
类型：浮点型百分比（0 到 100）的数组，读/写

shapeObject.value.featherRelSegLocs

形状对象属性：*shapeObject*.value.featherRelSegLocs
描述：一个包含在其蒙版路径片段（顶点之间的蒙版路径的章节，开始数为 0）上每个羽化点相对位置的数组，从 0 到 1。
要移动一个羽化点到另一个蒙版路径片段上，首先要改变 featherSegLocs 的值，然后再改变该属性值。
注意：存储在数组中的值按着羽化点创建的顺序排序。
类型：浮点型的数组，范围[0 到 1]，读/写

shapeObject.value.featherSegLocs

形状对象属性：*shapeObject*.value.featherSegLocs
描述：一个包含每个羽化点的蒙版路径片段数（在顶点之间的蒙版路径章节，开始数为 0）的数组。

通过改变它的片段数（该属性）或可选的，它的 featherRelSegLocs 属性值来移动一个羽化点到另一个片段。

类型：整数的数组，读/写

举例：

```
// 假定一个长方形封闭蒙版（片段数 0~3）有 3 个蒙版羽化点
// 移动全部 3 个羽化点到第一个蒙版片段
// 从蒙版中取得形状对象，假定这是层中第一个蒙版
var my_maskShape = layer.mask(1).property("ADBE Mask Shape").value;
// 检查蒙版羽化点在哪个位置
// 注意：这里储存的顺序为添加时的顺序
var where_are_myMaskFeatherPoints = my_maskShape.featherSegLocs;
// 移动全部 3 个羽化点到第 1 个蒙版片段（数字 0）
my_maskShape.featherSegLocs = [0, 0, 0];
// 更新该蒙版路径
layer.mask(1).property("ADBE Mask Shape").setValue(my_maskShape);
```

shapeObject.value.featherTensions

形状对象属性：*shapeObject*.value.featherTensions

描述：一个包含每个羽化点的张力数的数组，从 0（0%张力）到 1（100%张力）。

注意：存储在数组中的值按着羽化点创建的顺序排序。

类型：浮点型（0 到 1）的数组，读/写

shapeObject.value.featherTypes

形状对象属性：*shapeObject*.value.featherTypes

描述：一个包含每个羽化点方向的数组，0（向外羽化点）或 1（向内羽化点）的其中一个。

注意：你不可以在羽化点创建后去改变其方向。存储在数组中的值按着羽化点创建的顺序排序。

类型：整数型（0 或 1）的数组，读/写

shapeObject.value.inTangents

形状对象属性：*shapeObject*.value.inTangents

描述：

与形状顶点相关的入点切向量或方向把手。使用两个浮点型值的数组指定每个向量，并且将该向量收集到与顶点数组等长的数组内。

每个切线值默认为[0,0]。当蒙版形状不是自动贝塞尔曲线时，该结果导致了直线片段。

如果形状使用了自动贝塞尔曲线蒙版，所有切线值都会被忽略并且切线会被自动计算值。

类型：一对浮点型数组的数组，读/写

shapeObject.value.outTangents

形状对象属性：*shapeObject*.value.outTangents
描述：
与形状顶点相关的出点切向量或方向把手。使用两个浮点型值的数组指定每个向量，并且将该向量收集到与顶点数组等长的数组内。
每个切线值默认为[0,0]。当蒙版形状不是自动贝塞尔曲线时，该结果导致了直线片段。
如果形状使用了自动贝塞尔曲线蒙版，所有切线值都会被忽略并且切线会被自动计算值。
类型：一对浮点型数组的数组，读/写

shapeObject.value.vertices

形状对象属性：*shapeObject*.value.vertices
描述：该形状的锚点。使用两个浮点型值的数组指定每个点，并且将这对点收集到全部的点数组中。比如：

```
myShape.vertices = [[0,0], [0,1], [1,1], [1,0]];
```

类型：一对浮点型数组，读/写

形状图层对象

app.project.item(index).layer(index)

描述：

形状图层对象代表着合成内一个形状图层。使用图层收集对象的 addShape()方法创建该对象。参考查看 “层收集对象 addShape()方法”。它可以在一个项目的层收集对象中使用索引号或名称字符串来访问。

- 形状图层对象时媒体层对象的子类，媒体层对象是层对象的子类。所有媒体层和层对象的方法和属性都可以被形状图层调用。参考查看 “层对象” 和 “媒体层对象”。

固态层源对象

app.project.item(index).mainSource
app.project.item(index).proxySource
描述：

固态层源对象代表了一个固态-纯色层素材源。

- 固态层源时素材源的子类。所有素材源的方法和属性都可以被固态层源调用。参考查看“素材源对象”。

属性：

属性	描述
color	固态层的颜色

solidSource.color

固态层源对象属性：solidSource.color

描述：固态层的颜色，用红色、绿色、蓝色值表示。

类型：三个浮点型值的数组。[R,G,B]，范围[0.0~1.0]，读/写

系统对象

system

描述：

系统对象提供了用户系统属性的访问接口，比如用户名、操作系统名称、操作系统版本等等。这些都可以通过 system 全局变量取得。

举例：

```
alert ("你的操作系统是" + system.osName + "，运行的版本是" + system.osVersion);
confirm("你是：  " + system.userName + "，运行在 " + system.machineName + "。");
```

属性：

属性	描述
userName	当前用户名
machineName	计算机的名称
osName	操作系统的名称
osVersion	操作系统的版本号

方法：

方法	描述
callSystem()	在系统的命令行中执行一个命令

system.callSystem()

系统对象方法：system.callSystem(*cmdLineToExecute*)

描述：

执行一个系统的命令，就像在操作系统的命令行中输入一样。返回系统在命令行中相应的任意输出，如果有的话。

在 Windows 中，你可以在 cmd.exe 命令中使用/c 开关调用命令，运行转换引号 (¥”…¥”) 中的命令。如比，下面检索了当前时间并将其显示给用户：

```
var timeStr = system.callSystem("cmd.exe /c \"time /t\"");
alert("当前时间是" + timeStr);
```

参数：

cmdLineToExecute	包含一段命令和它的参数的字符串

返回：命令的输出

system.machineName

系统对象属性：system.machineName
描述：正在运行 AE 的计算机的名称。
类型：字符串型，只读

system.osName

系统对象属性：system.osName
描述：正在运行 AE 的操作系统的名称。
注意：对于 Windows 7，该属性会返回一个空值，使用$.os 代替。
类型：字符串型，只读

system.osVersion

系统对象属性：system.osVersion
描述：当前本地操作系统的版本。
类型：字符串型，只读

system.userName

系统对象属性：system.userName
描述：当前登录到该系统的用户名称。
类型：字符串型，只读

文本文档对象

new TextDocument(docText)
app.project.item(indedx).layer(index).property(“Source Text”).value

描述：

文本文档对象储存了文本层的源文本属性的值。使用该构造函数创建对象并将这段字符串封装。

举例：

这段代码设置了一些源文本的值并以警告框的形式显示新值：

```
var myTextDocument = new TextDocument("幸福蛋糕");
myTextLayer.property("Source Text").setValue(myTextDocument);
alert(myTextLayer.property("Source Text").value);
```

这段代码设置了随着时间显示不同文本值的关键帧：

```
var textProp = myTextLayer.property("Source Text");
textProp.setValueAtTime(0, new TextDocument("幸福"));
textProp.setValueAtTime(.33, new TextDocument("蛋糕"));
textProp.setValueAtTime(.66, new TextDocument("很"));
textProp.setValueAtTime(1, new TextDocument("好吃！"));
```

这段代码设置了一些文本的字符和段落的设置：

```
var textProp = myTextLayer.property("Source Text");
var textDocument = textProp.value;
myString = "节日快乐！";
textDocument.resetCharStyle();
textDocument.fontSize = 60;
textDocument.fillColor = [1, 0, 0];
textDocument.strokeColor = [0, 1, 0];
textDocument.strokeWidth = 2;
textDocument.font = "TimesNewRomanPSMT";
textDocument.strokeOverFill = true;
textDocument.applyStroke = true;
textDocument.applyFill = true;
textDocument.text = myString;
textDocument.justification = ParagraphJustification.CENTER_JUSTIFY;
textDocument.tracking = 50;
textProp.setValue(textDocument);
```

属性：

属性	描述
text	该文本层的源文本值
font	该文本层的字体，通过它的 PostScript 名称
fontSize	该文本层的字体像素大小

applyFill	值为 true 时，该文本层显示填充
applyStroke	值为 true 时，该文本层显示描边
fillColor	该文本层填充的颜色
strokeColor	该文本层的描边颜色
strokeOverFill	指定文本层填充和描边的渲染顺序
strokeWidth	该文本层的描边粗细
justification	文本层的段落对齐
tracking	该文本层的字符间隔
pointText	值为 true 时，该文本层时点（无边界）文本
boxText	值为 true 时，该文本层是段落（有边界）文本
boxTextSize	对于段落文本，段落边界的像素尺寸

方法：

方法	描述
resetCharStyle()	在字符面板中恢复默认的字符设置
resetParagraphStyle()	在段落面板中恢复默认的段落设置

textDocument.applyFill

文本文档对象属性：*textDocument*.applyFill
描述：值为 true 时，该文本图层显示填充。访问 fillColor 属性取得实际颜色。值为 false 时，只显示描边。
类型：布尔型，读/写

textDocument.applyStroke

文本文档对象属性：*textDocument*.applyStroke
描述：值为 true 时，该文本层显示描边。访问 strokeColor 属性来取得实际颜色，访问 strokeWidth 属性取得粗细。值为 false 时，只显示填充。
类型：布尔型，读/写

textDocument.boxText

文本文档对象属性：*textDocument*.boxText
描述：如果一个文本层是段落文本层则返回 true，否则返回 false。
类型：布尔型，只读

textDocument.boxTextSize

文本文档对象属性：*textDocument*.boxTextSize
描述：段落文本层的尺寸，一个像素尺寸[宽,高]的数组。
类型：两个整数型（最小值为 1）的数组，读/写

textDocument.fillColor

文本文档对象属性：*textDocument*.fillColor
描述：
该文本层的填充颜色，一个[r,g,b]浮点型值的数组。比如，在 8 位颜色深度工程中，255 的红色值为是 1.0，在 32 位颜色深度工程中，一个过亮的蓝色值可能是 3.2。
注意：如果该文本层给每个字符设置了不同的填充颜色，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
类型：浮点型值的[r,g,b]数组，读/写

textDocument.font

文本文档对象属性：*textDocument*.font
描述：
该文本层通过其 PostScript 名称指定的字体。
注意：如果该文本层给每个字符设置了不同的字体，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
类型：字符串，读/写

textDocument.fontSize

文本文档对象属性：*textDocument*.fontSize
描述：
该文本层的字体像素尺寸。

注意：如果该文本层给每个字符设置了不同的字体尺寸，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。

类型：浮点型，范围[0.1~1296]，读/写

textDocument.justification

文本文档对象属性：*textDocument*.justification

描述：该文本层的段落对齐方式。

类型：

一些列举的段落对齐值，只读

ParagraphJustification.LEFT_JUSTIFY
ParagraphJustification.RIGHT_JUSTIFY ParagraphJustification.CENTER_JUSTIFY
ParagraphJustification.FULL_JUSTIFY_LASTLINE_LEFT
ParagraphJustification.FULL_JUSTIFY_LASTLINE_RIGHT
ParagraphJustification.FULL_JUSTIFY_LASTLINE_CENTER
ParagraphJustification.FULL_JUSTIFY_LASTLINE_FULL

textDocument.pointText

文本文档对象属性：*textDocument*.pointText

描述：如果一个文本层为点文本层则返回 true，否则为 false。

类型：布尔型，只读

textDocument.resetCharStyle()

文本文档对象方法：*textDocument*.resetCharStyle()

描述：重置字符面板中默认的文本字符的规格参数。

参数：无

返回：无

textDocument.resetParagraphStyle()

文本文档对象方法：*textDocument*.resetParagraphStyle()

描述：重置段落面板中默认的文本段落的规格参数。

参数：无

返回：无

textDocument.strokeColor

文本文档对象属性：*textDocument*.strokeColor
描述：
该文本层的描边颜色，一个[r,g,b]浮点型值的数组。比如，在 8 位颜色深度工程中，255 的红色值为是 1.0，在 32 位颜色深度工程中，一个过亮的蓝色值可能是 3.2。
注意：如果该文本层给每个字符设置了不同的填充颜色，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
类型：浮点型值的[r,g,b]数组，读/写

textDocument.strokeOverFill

文本文档对象属性：*textDocument*.strokeOverFill
描述：
指定文本层填充和描边的渲染顺序。值为 true 时，描边在填充之上显示。
注意：如果该文本层给每个字符设置了不同的渲染顺序，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
类型：布尔型，读/写

textDocument.strokeWidth

文本文档对象属性：*textDocument*.strokeWidth
描述：该文本层描边的像素粗细。
注意：如果该文本层给每个字符设置了不同的描边像素粗细，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
返回：浮点型值（0 到 1000），读/写

textDocument.text

文本文档对象属性：*textDocument*.text
描述：该文本层的源文本属性的文本值。
类型：字符串型，读/写

textDocument.tracking

文本文档对象属性：*textDocument*.tracking
描述：该文本层的字符间距。

注意：如果该文本层给每个字符设置了不同的字符边距，此属性返回第一个字符的设置。同时，如果你改变了值，它会重置所有在文本层中的字符。
类型：浮点型，读/写

文本层对象

app.project.item(index).layer(index)

描述：

文本层对象代表了合成中一个文本层。使用层收集对象的 addText()方法创建该对象。参考查看 “层收集对象 addText()方法”。该对象可通过项目的层收集对象指定索引号或名称字符串访问。

- 文本层对象是媒体层对象的子类，媒体层对象也是层对象的子类。所有媒体层和层对象的方法和属性都可以被文本层对象访问。查看 “层对象” 和 “媒体层对象”。

AE 属性：

文本层对象没有额外定义属性，但是除了继承媒体层属性之外，有下列这些 AE 属性和属性组：

```
Text
        Source Text
        Path Options
                Path
                Reverse Path
                Perpendicular To Path
                Force Alignment
                First Margin
                Last Margin
        More Options
                Anchor Point Grouping
                Grouping Alignment
                Fill & Stroke
                Inter-Charater Blending
        Animators
```

不再使用的 **AE** 属性和对象属性：

继承于媒体层的 Time Remap 和 Motion Trackers 属性在文本层中不能应用，还有一些媒体层对象属性不能使用：

```
canSetTimeRemapEnabled
timeRemapEnabled
trackMatteType
isTrackMatte
hasTrackMatte
```

查看窗口对象

app.activeViewer

描述：

查看窗口对象代表一个合成、图层或素材面板。

举例：

这是最大化当前激活查看窗口面板，并且如果包含一个合成则显示查看窗口的类型：

```
var activeViewer = app.activeViewer;
activeViewer.maximized = true;
if (activeViewer.type == ViewerType.VIEWER_COMPOSITION)
alert("合成面板已被激活！");
```

属性：

属性	描述
type	查看窗口的内容类型。
active	值为 true 时，查看窗口被激活。
maximized	值为 true 时，查看窗口正处于最大化。

方法：

属性	描述
setActive()	将查看窗口移动到桌面顶层并激活

viewer.active

查看窗口对象属性：*viewer*.active

描述：值为 true 时，查看窗口被激活并移动到桌面顶层。

类型：布尔型，只读

viewer.maximized

查看窗口对象属性：*viewer*.maximized

描述：值为 true 时，查看窗口最大化。

类型：布尔型，读/写

viewer.setActive()

查看窗口对象方法：*viewer*.setActive()
描述：将查看窗口移动到桌面顶层并激活。调用这个方法会将查看窗口对象的 active 属性值设置为 true。
参数：无
返回：布尔型

viewer.type

查看窗口对象属性：*viewer*.type
描述：查看窗口面板的内容。
类型：
一些列举的查看窗口类型值，只读
ViewerType.VIEWER_COMPOSITION
ViewerType.VIEWER_LAYER
ViewerType.VIEWER_FOOTAGE