

表达式语言引用

搜索Adobe 支持



了解 After Effects 中的表达式和表达式引用，如时间转换方法和矢量数学方法。

使用 After Effects 表达式元素以及标准 JavaScript 元素来编写您的表达式。您可随时使用“表达式语言”菜单将方法和属性插入到表达式中，您还可以随时使用关联器来插入属性。

如果参数描述包含等号 (=) 和一个值（例如 `t=time` 或 `width=.2`），则当您未指定其他值时，该参数将使用随附的默认值。

一些参数描述在方括号中包括一个数字，此数字表示预期的属性或数组的维度。

一些返回值描述在方括号中包括一个数字，此数字指定返回的属性或数组的维度。如果未包括特定维度，则返回的数组的维度取决于输入的维度。

[W3Schools JavaScript 参考网站](#)提供了标准 JavaScript 语言的信息，包括 JavaScript `Math` 和 `String` 对象的页面。

表达式：JavaScript 引擎

After Effects 在表达式求值时，使用 JavaScript 引擎。渲染期间的表达式求值性能比旧 ExtendScript 引擎快 5 倍。

在 Windows 上，After Effects 使用 [V8 开源 JavaScript 引擎](#)。此引擎提供了更加先进的 JavaScript 版本，而 ExtendScript 仅支持第三版 ECMA-262 标准。

在此页面上

[表达式：JavaScript 引擎](#)

[全局对象、属性和方法（表达式引用）](#)

[时间转换方法（表达式引用）](#)

[矢量数学方法（表达式引用）](#)

[随机数方法（表达式引用）](#)

[插值方法（表达式引用）](#)

[颜色转换方法（表达式引用）](#)

[其他数学方法（表达式引用）](#)

[合成属性和方法（表达式引用）](#)

[素材属性和方法（表达式引用）](#)

[图层子对象属性和方法（表达式引用）](#)

[图层常规属性和方法（表达式引用）](#)

[图层属性特性和方法（表达式引用）](#)

要为某个项目选择是用 JavaScript 引擎还是用旧版的 ExtendScript，请在“**项目设置**”对话框中，单击“**表达式**”选项卡，然后更改“**表达式引擎**”选项。使用过往版本 After Effects 保存的项目将默认使用旧版 ExtendScript。

有关 JavaScript 与旧版 ExtendScript 表达式引擎之间特定语法差异的更多信息，请参阅 [JavaScript 和旧版 ExtendScript 表达式引擎之间的语法差异](#)。

JavaScript 引擎带来的改进

- 浮点运算的数学精度更高。
- 可使用来自新版 JavaScript 的数组和字符串方法。
- 还可使用其他新版 JavaScript 对象（类似类型化数组）、使用关键字“let”和“const”，以及大量其他新版 ECMA-262 标准提供的新增内容。

JavaScript 引擎与 ExtendScript 之间的差异

- 在 JavaScript 引擎中，if/else 语句的语法非常严格，且需要根据 JavaScript 标准进行编写。
- 将文本图层的字符索引当作数组进行存取时，使用 `text.sourceText.value[i]` 而不是 `text.sourceText[i]`。
- 已经在 ExtendScript 中过时的老式 snake_case 表达式方法（例如 `this_comp` 和 `to_world`）不受支持。
- 简化的“this(arg)”语法不受支持。

全局对象、属性和方法（表达式引用）

comp(name) 返回类型：合成。参数类型：*name* 是一个字符串。按照名称检索其他合成。

[图层 3D 属性和方法（表达式引用）](#)

[图层空间变换方法（表达式引用）](#)

[摄像机属性和方法（表达式引用）](#)

[光照属性和方法（表达式引用）](#)

[效果属性和方法（表达式引用）](#)

[蒙版属性和方法（表达式引用）](#)

[属性特性和方法（表达式引用）](#)

[项目属性（表达式引用）](#)

[Key 属性和方法（表达式引用）](#)

[MarkerKey 属性（表达式引用）](#)

[MarkerValue.protectedRegion（表达式引用）](#)

[访问有关形状、蒙版和画笔描边的路径点的表达式（表达式引用）](#)

[数据驱动的动画（表达式引用）](#)

[十六进制转 RGB 颜色转换方法（表达式引用）](#)

适用于: **Adobe After Effects**

footage(name) 返回类型：素材。 参数类型：*name* 是一个字符串。按照名称检索素材项目。

thisComp 返回类型：合成。 表示包含表达式的合成。

thisLayer 返回类型：图层、光照或摄像机。 表示包含表达式的图层。因为 **thisLayer** 是默认对象，所以其使用是可选的。例如，以 **thisLayer.width** 或 **width** 开头的表达式将生成相同的结果。

thisProperty 返回类型：属性。 表示包含表达式的属性。例如，如果您对旋转属性编写表达式，则可使用以 **thisProperty** 开头的表达式来引用旋转属性。

time 返回类型：数值。 表示以秒为单位的合成时间，将以此计算表达式。

colorDepth 返回类型：数值。 返回项目颜色深度值。例如，当每个通道的项目颜色深度为 16 位时，**colorDepth** 返回 16。

posterizeTime(framesPerSecond) 返回类型：数值。 参数类型：*framesPerSecond* 是数值。*framesPerSecond* 值成为其余表达式运行的帧速率。此表达式允许您将属性的帧速率设置为低于合成的速率。例如，以下表达式每秒使用随机值更新一次属性值：
`posterizeTime(1); random()`

value 返回类型：数值、数组或字符串。 表示当前时间包含表达式的属性的值。

时间转换方法（表达式引用）

timeToFrames(t = time + thisComp.displayStartTime, fps = 1.0 / thisComp.frameDuration, isDuration = false)

返回类型：数值。参数类型：*t* 和 *fps* 是数值；*isDuration* 是布尔值。将默认为当前合成时间的 *t* 的值转换为整数数目的帧。每秒的帧数在 *fps* 参数中指定，该参数默认为当前合成的帧速率 ($1.0/\text{thisComp.frameDuration}$)。 *isDuration* 参数默认为 False，如果 *t* 值表示两个时间的差值而非绝对时间则为 True。绝对时间向下舍入到负无穷；持续时间向远离零的方向舍入（向上舍入为正值）。

framesToTime(frames, fps = 1.0 / thisComp.frameDuration) 返回类型：数值。参数类型：*frames* 和 *fps* 是数值。**timeToFrames** 的反向。返回与必需的 *frames* 参数对应的时间。它不必是一个整数。有关 *fps* 参数的说明，请参阅 *timeToFrames*。

timeToTimecode(t = time + thisComp.displayStartTime, timecodeBase = 30, isDuration = false)

返回类型：字符串。参数类型：*t* 和 *timecodeBase* 是数值；*isDuration* 是布尔值。将 *t* 的值转换为表示时间码的字符串。有关 *t* 和 *isDuration* 参数的说明，请参阅 **timeToFrames**。*timecodeBase* 值默认为 30，用于指定一秒内的帧数。

timeToNTSCTimecode(t = time + thisComp.displayStartTime, ntscDropFrame = false, isDuration = false)

返回类型：字符串。参数类型：*t* 是数值，*ntscDropFrame* 和 *isDuration* 是布尔值。将 *t* 转换为表示 NTSC 时间码的字符串。有关 *t* 和 *isDuration* 参数的说明，请参阅 **timeToFrames**。如果 *ntscDropFrame* 为 False（默认值），则结果字符串为 NTSC 未丢帧时间码。如果 *ntscDropFrame* 为 True，则结果字符串为 NTSC 丢帧时间码。

timeToFeetAndFrames(t = time + thisComp.displayStartTime, fps = 1.0 / thisComp.frameDuration, framesPerFoot = 16, isDuration = false)

返回类型：字符串。参数类型：*t*、*fps* 和 *framesPerFoot* 是数值；*isDuration* 是布尔值。将 *t* 的值转换为表示胶片和帧的英尺的字符串。有关 *t*、*fps* 和 *isDuration* 参数的说明，请参阅 **timeToFrames**。*framesPerFoot* 参数指定一英尺胶片中的帧数。它默认为 16，是 35 毫米素材的最常见速率。

timeToCurrentFormat(t = time + thisComp.displayStartTime, fps = 1.0 / thisComp.frameDuration, isDuration = false)

返回类型：字符串。参数类型：*t* 和 *fps* 是数值；*isDuration* 是布尔值。将 *t* 的值转换为表示采用当前项目设置显示格式的时间的字符串。有关所有参数的定义，请参阅 **timeToFrames**。可选的 **ntscDropFrame** 参数已添加到 After Effects CS5.5 和更高版本中的 **timeToCurrentFormat()** 函数。默认值：**ntscDropFrame = thisComp.ntscDropFrame**。

注意：

如果您需要对素材中的时间码外观进行更多控制，请使用 **timeToCurrentFormat** 方法或其他 **timeTo** 方法生成时间码，而非使用时间码或编号效果。创建文本图层，向源文本属性添加表达式，并在表达式字段中输入 **timeToCurrentFormat()**。使用此方法，您可以对时间码文本进行格式设置以及动画制作。此外，时间码使用当前的项目设置定义的同一直显示样式。

矢量数学方法（表达式引用）

矢量数学函数是对数组进行运算的全局方法，将其视为数学矢量。与内置 JavaScript 方法（例如 **Math.sin**）不同，这些方法不与 **Math** 前缀一起使用。除非另有说明，否则矢量数学方法对维度的要求是宽松的并返回属于最大输入数组对象的维度的值，用零填充缺失的元素。例如，表达式 **add([10, 20], [1, 2, 3])** 返回 **[11, 22, 3]**。

[JJ Gifford 的网站](#)提供了演示如何将简单的几何图形和三角函数与表达式结合使用的说明和示例。

add(vec1, vec2) 返回类型：数组。参数类型：vec1 和 vec2 是数组。添加两个矢量。

sub(vec1, vec2) 返回类型：数组。参数类型：vec1 和 vec2 是数组。减去两个矢量。

mul(vec, amount) 返回类型：数组。参数类型：vec 是数组，amount 是数值。将矢量的每个元素与数量相乘。

div(vec, amount) 返回类型：数组。参数类型：vec 是数组，amount 是数值。用矢量的每个元素除以数量。

clamp(value, limit1, limit2) 返回类型：数值或数组。参数类型：value、limit1 和 limit2 是数值或数组。value 的每个组件的值都限定为介于 limit1 和 limit2 相应值的值之间。

dot(vec1, vec2) 返回类型：数值。参数类型：vec1 和 vec2 是数组。返回矢量参数的点（内）积。

cross(vec1, vec2) 返回类型：数组 [2 或 3]。参数类型：vec1 和 vec2 是数组 [2 或 3]。返回 vec1 和 vec2 的矢量叉积。有关更多信息，请参阅数学参考或 JavaScript 指南。

normalize(vec) 返回类型：数组。参数类型：vec 是数组。标准化矢量以使其长度为 1.0。使用 **normalize** 方法是执行运算 **div(vec, length(vec))** 的简便方法。

length(vec) 返回类型：数值。参数类型：vec 是数组。返回矢量 vec. 的长度

length(point1, point2)

返回类型：数值。参数类型：*point1* 和 *point2* 是数组。返回两点之间的距离。*point2* 参数是可选的。例如，`length(point1, point2)` 与 `length(sub(point1, point2))` 一样。例如，将此表达式添加到摄像机的焦距属性中，从而将焦平面锁定到摄像机的目标点，以便目标点对准焦点：`length(position, pointOfInterest)`

lookAt(fromPoint, atPoint) 返回类型：数组 [3]。参数类型：*fromPoint* 和 *atPoint* 是数组 [3]。参数 *fromPoint* 是您要定向的图层的世界空间中的位置。参数 *atPoint* 是您要将图层指向的世界空间中的点。返回值可用作“方向”属性的表达式，使图层点的 z 轴指向 *atPoint*。此方法对摄像机和光照特别有用。如果您对摄像机使用此表达式，请关闭自动方向。例如，聚光灯的方向属性的以下表达式会将光点指向同一合成中的 1 号图层的锚点：`lookAt(position, thisComp.layer(1).position)`

随机数方法（表达式引用）

注意：

用于随机地改变属性值的摆动方法属于属性特性和方法类别。有关更多信息，请参阅[“属性”特性和方法（表达式引用）](#)。

seedRandom(offset, timeless=false) 返回类型：无。参数类型：*offset* 是数值，*timeless* 是布尔值。**random** 和 **gaussRandom** 方法使用控制数字序列的种子值。默认情况下，种子计算为唯一图层标识符的函数、图层中的属性、当前时间以及位移值 0。调用 **seedRandom** 以将位移设为 0 之外的某个值，从而创建其他随机序列。对 *timeless* 参数使用 **true** 以便不使用当前时间作为随机种子的输入。对 **timeless** 参数使用 **true** 使您能够生成一个随机数值，该值不会随计算时间而改变。*offset* 值（而非 *timeless* 值）还用于控制 **wiggle** 函数的初始值。例如，不透明度属性的以下表达式可将不透明度值设为不随

时间而改变的随机值：`seedRandom(123456, true); random() * 100`此示例中的乘以 100 会将 `random` 方法返回的范围 0–1 内的值转换为范围 0–100 内的数；此范围通常对不透明度属性（具有 0% 到 100% 不等的值）更有用。

random() 返回类型：数值。返回范围 0–1 内的随机数。在 After Effects CC 和 CS6 中，当图层 ID 相互靠近时，`random()` 的行为会变得更随机。`wiggle()` 表达式不受影响。

random(maxValOrArray) 返回类型：数值或数组。参数类型：*maxValOrArray* 是数值或数组。如果 *maxValOrArray* 是数值，则此方法会返回范围 0 到 *maxValOrArray* 的数值。如果 *maxValOrArray* 是数组，则此方法会返回与 *maxValOrArray* 维度相同的数组，每个组件的范围为 0 到 *maxValOrArray* 的相应组件。

random(minValOrArray, maxValOrArray) 返回类型：数值或数组。参数类型：*minValOrArray* 和 *maxValOrArray* 是数值或数组。如果 *minValOrArray* 和 *maxValOrArray* 是数值，则此方法会返回范围 *minValOrArray* 到 *maxValOrArray* 内的数值。如果参数是数组，则此方法会返回与维度更大的数组维度相同的数组，每个组件处于 *minValOrArray* 的相应组件到 *maxValOrArray* 的相应组件这一范围中。例如，表达式 `random([100, 200], [300, 400])` 返回其第一个值在范围 100–300 内且其第二个值在范围 200–400 内的数组。如果两个输入数组的维度不匹配，则将用零填充较短数组的更高维度的值。

gaussRandom() 返回类型：数值。返回随机数。结果有一个高斯（钟形）分布。大约 90% 的结果都处于范围 0–1 内，其余 10% 在此范围之外。

gaussRandom(maxValOrArray) 返回类型：数值或数组。参数类型：*maxValOrArray* 是数值或数组。当 *maxValOrArray* 是数值时，此方法会返回一个随机数。大约 90% 的结果都处于 0 到 *maxValOrArray* 范围内，其余 10% 在此范围之外。当 *maxValOrArray* 是数组时，此方法会返回一个随机值数组，维度与 *maxValOrArray* 相同。90% 的值都处在 0 到 *maxValOrArray* 范围内，其余 10% 在此范围之外。结果有一个高斯（钟形）分布。

gaussRandom(minValOrArray, maxValOrArray) 返回类型：数值或数组。参数类型：*minValOrArray* 和 *maxValOrArray* 是数值或数组。如果 *minValOrArray* 和 *maxValOrArray* 是数值，则此方法会返回随机数。大约 90% 的结果都介于 *minValOrArray* 到 *maxValOrArray* 范围内，其余 10% 在此范围之外。如果这些参数是数组，则此方法会返回与具有较大维度的参数维度相同的随机数数组。对于每个组件，大约 90% 的结果都处于 *minValOrArray* 的相应组件到 *maxValOrArray* 的相应组件范围内，其余 10% 在此范围之外。结果有一个高斯（钟形）分布。

noise(valOrArray) 返回类型：数值。参数类型：*valOrArray* 是数值或数组 [2 或 3]。返回范围 -1 到 1 中的数值。噪声实际上不是随机的；它基于柏林噪声，这意味着相邻的两个输入值的返回值往往也是相邻的。此类噪声在您需要看似随机且相差不会很大的数值序列时（在对任何明显随机的自然运动进行动画制作时通常就如此）非常有用。示例：

```
rotation + 360*noise(time)
```

插值方法（表达式引用）

对于所有“插值”方法，参数 *t* 通常是 **time** 或 **value**，但它也可以采用其他值。如果 *t* 是 **time**，则值之间的插值会在持续时间内发生。如果 *t* 是 **value**，则表达式会将一系列值映射到新系列值。

有关插值方法的其他说明和示例，请访问 [JJ Gifford 的网站](#)。

Chris 和 Trish Meyer 在 [ProVideo Coalition 网站](#)上的一篇文章中提供了这些方法的附加信息和示例。

Ian Haigh 在 [After Effects Scripts 网站](#)上提供了一个脚本，您可将该脚本用于将高级插值方法表达式（例如回弹）轻松应用于属性。

Andrew Devis 在 Creative COW 网站上提供了[两个视频教程](#)，其中详细介绍了如何结合使用**线性**表达式方法和“将音频转换为关键帧”命令。

linear(t, tMin, tMax, value1, value2) 返回类型：数值或数组。参数类型：t、tMin 和 tMax 是数值，value1 和 value2 是数值或数组。当 $t \leq tMin$ 时返回 value1。当 $t \geq tMax$ 时返回 value2。当 $tMin < t < tMax$ 时，返回 value1 和 value2 之间的线性插值。例如，不透明度属性的以下表达式可导致不透明度值在 0 秒到 6 秒的时间内从 20% 线性渐变为 80%：
`linear(time, 0, 6, 20, 80)` 此方法（像所有“插值”方法一样）还可用于从一系列值转换为其他系列值。例如，不透明度属性的以下表达式可将不透明度值从范围 0%-100% 转换为范围 20%-80%：
`linear(value, 0, 100, 20, 80)`

linear(t, value1, value2) 返回类型：数值或数组。参数类型：t 是数值，value1 和 value2 是数值或数组。当 t 介于 0 到 1 时返回从 value1 到 value2 进行线性插值的值。当 $t \leq 0$ 时返回 value1。当 $t \geq 1$ 时返回 value2。

ease(t, value1, value2) 返回类型：数值或数组。参数类型：t 是数值，value1 和 value2 是数值或数组。与具有相同参数的 **linear** 类似，只不过插值渐进和渐出以使开始点和结束点的速度为 0。此方法会产生一个非常流畅的动画。

ease(t, tMin, tMax, value1, value2) 返回类型：数值或数组。参数类型：t、tMin 和 tMax 是数值，value1 和 value2 是数值或数组。与具有相同参数的 **linear** 类似，只不过插值渐进和渐出以使开始点和结束点的速度为 0。此方法会产生一个非常流畅的动画。

easeIn(t, value1, value2) 返回类型：数值或数组。参数类型：t 是数值，value1 和 value2 是数值或数组。类似于 **ease**，只不过切线仅在 value1 一侧为 0 且插值在 value2 一侧是线性的。

easeIn(t, tMin, tMax, value1, value2) 返回类型：数值或数组。参数类型：t、tMin 和 tMax 是数值，value1 和 value2 是数值或数组。类似于 **ease**，只不过切线仅在 tMin 一侧为 0 且插值在 tMax 一侧是线性的。

easeOut(t, value1, value2) 返回类型：数值或数组。参数类型：*t* 是数值，*value1* 和 *value2* 是数值或数组。类似于 **ease**，只不过切线仅在 *value2* 一侧为 0 且插值在 *value1* 一侧是线性的。

easeOut(t, tMin, tMax, value1, value2) 返回类型：数值或数组。参数类型：*t*、*tMin* 和 *tMax* 是数值，*value1* 和 *value2* 是数值或数组。类似于 **ease**，只不过切线仅在 *tMax* 一侧为 0 且插值在 *tMin* 一侧是线性的。

颜色转换方法（表达式引用）

Harry Frank 在其 [graymachine 网站](#)上提供了一个视频教程，其中演示了如何使用这些颜色转换方法来更改无线电波效果产生的波形的颜色。

rgbToHsl(rgbaArray) 返回类型：数组 [4]。参数类型：*rgbaArray* 是数组 [4]。将 RGBA 空间中的颜色转换为 HSLA 空间。输入是标准化的红色、绿色、蓝色和 Alpha 通道值数组，全部介于 0.0 到 1.0 范围内。生成的值是色相、饱和度、亮度以及 Alpha 通道值数组，同样介于 0.0 到 1.0 范围内。示例：
`rgbToHsl.effect("Change Color")`
`("Color To Change")`

hslToRgb(hslaArray) 返回类型：数组 [4]。参数类型：*hslaArray* 是数组 [4]。将 HSLA 空间中的颜色转换为 RGBA 空间。此转换与 **rgbToHsl** 方法执行的转换相反。

其他数学方法（表达式引用）

degreesToRadians(degrees) 返回类型：数值。参数类型：*degrees* 是数值。将度转换为弧度。

radiansToDegrees(radians) 返回类型：数值。 参数类型：*radians* 是数值。将弧度转换为度。

合成属性和方法（表达式引用）

layer(index) 返回类型：图层、光照或摄像机。 参数类型：*index* 是数值。按照编号（“时间轴”面板中的顺序）检索图层。示例：`thisComp.layer(3)`

layer(name) 返回类型：图层、光照或摄像机。 参数类型：*name* 是一个字符串。按照名称检索图层。名称是根据图层名称进行匹配的，如果没有图层名称，则根据源名称。如果存在副本名称，After Effects 会使用“时间轴”面板中的第一个（最高）名称。示例：

```
thisComp.layer("Solid 1")
```

layer(otherLayer, relIndex) 返回类型：图层、光照或摄像机。 参数类型：*otherLayer* 是图层对象，*relIndex* 是数值。检索属于 *otherLayer* 上面或下面的 *relIndex* 图层的图层。例如，如果“时间轴”面板中再下面的图层处于活动状态，则 `thisComp.layer(thisLayer, 1).active` 将返回 true。

标记 返回类型：MarkerProperty。

注意:

您无法按照标记编号访问合成标记。如果您在 After Effects 的早期版本中创建了在使用合成标记编号的项目，则必须更改这些调用以改用 `marker.key(name)`。由于合成标记的默认名称是数值，因此，转换引用以使用名称通常只是用引号括起编号的问题。

marker.key(index) 返回类型：MarkerKey。 参数类型：index 是数值。 返回具有指定索引的标记的 MarkerKey 对象。索引引用标记在合成时间中的顺序，而不是标记名称。例如，以下表达式返回第一个合成标记的时间：
`thisComp.marker.key(1).time`

marker.key(name) 返回类型：MarkerKey。 参数类型：name 是一个字符串。返回具有指定名称的标记的 MarkerKey 对象。name 值是标记对话框的注释字段中键入的标记名称，例如，`marker.key("1")`。对于合成标记，默认名称是数值。如果合成中的多个标记具有相同名称，则此方法返回第一次出现的（在合成时间）标记。标记密钥的值是字符串，不是数值。例如，以下表达式返回具有名称“0”的合成标记的时间：
`thisComp.marker.key("0").time`

marker.nearestKey(t) 返回类型：MarkerKey。 参数类型：t 是数值。返回时间最接近 t 的标记。例如，以下表达式返回最接近 1 秒时间的合成标记的时间：

`thisComp.marker.nearestKey(1).time` 此表达式返回最接近当前时间的合成标记的时间：
`thisComp.marker.nearestKey(time).time`

marker.numKeys 返回类型：数值。 返回合成中合成标记的总数。

numLayers 返回类型：数值。 返回合成中的图层数。

activeCamera 返回类型：摄像机。 返回通过其在当前帧渲染合成的摄像机的摄像机对象。此摄像机未必是您通过其查看“合成”面板的摄像机。

width 返回类型：数值。 返回合成宽度（以像素为单位）。将以下表达式应用于图层的位置属性，以将合成帧中的图层居中：
`[thisComp.width/2, thisComp.height/2]`

height 返回类型：数值。 返回合成高度（以像素为单位）。

duration 返回类型：数值。返回合成持续时间（以秒为单位）。

ntscDropFrame 返回类型：布尔值。如果时间码是丢帧格式，则返回 true。（After Effects CS5.5 和更高版本。）

displayStartTime 返回类型：数值。返回合成启动时间（以秒为单位）。

frameDuration 返回类型：数值。返回帧持续时间（以秒为单位）。

shutterAngle 返回类型：数值。返回合成的快门角度值（以度为单位）。

shutterPhase 返回类型：数值。返回合成的快门相位（以度为单位）。

bgColor 返回类型：数组 [4]。返回合成的背景颜色。

pixelAspect 返回类型：数值。返回合成的像素长宽比。

name 返回类型：字符串。返回合成的名称。

合成标记受保护区域属性

合成标记的“受保护区域”选项也可读作 **protectedRegion** 合成标记属性。

素材属性和方法（表达式引用）

要将“项目”面板中的素材项目用作表达式中的对象，请使用全局 **footage** 方法，正如 **footage("file_name")** 中一样。您还可以使用图层（其源为素材项目）上的 **source**

属性访问素材对象。

width 返回类型：数值。返回素材项目的宽度（以像素为单位）。

height 返回类型：数值。返回素材项目的高度（以像素为单位）。

duration 返回类型：数值。返回素材项目的持续时间（以秒为单位）。

frameDuration 返回类型：数值。返回素材项目中帧的持续时间（以秒为单位）。

ntscDropFrame 返回类型：布尔值。如果时间码是丢帧格式，则返回 true。（After Effects CS5.5 和更高版本。）

pixelAspect 返回类型：数值。返回素材项目的像素长宽比。

name 返回类型：字符串。返回“项目”面板中所示的素材项目的名称。

图层子对象属性和方法（表达式引用）

注意：

在 After Effects CC 和 CS6 中，“表达式语言”菜单、“图层子对象”、“图层常规”、“图层属性”、“图层 3D”以及“图层空间变换”已列入到“图层”子菜单中。

source 返回类型：合成或素材。返回图层的源合成或源素材对象。默认时间调整为源中的时间。示例：`source.layer(1).position`

sourceTime(t = time) 返回类型：数值。返回与时间 *t* 相应的图层源。（ After Effects CS5.5 和更高版本。）

sourceRectAtTime (t=时间, includeExtents= false) 返回类型：具有四个属性的 JavaScript 对象：[上边界、左边界、宽度、高度]。范围仅适用于形状图层，可根据需要增加图层范围的大小；也可增加段落文本图层（ After Effects 15.1 及更高版本）的大小，此时它会返回段落框的范围。示例：`myTextLayer.sourceRectAtTime().width`。

effect(name) 返回类型：效果。参数类型：*name* 是一个字符串。After Effects 在“效果控件”面板中按照名称查找效果。名称可以是默认名称或者用户定义的名称。如果多个效果具有相同名称，则会使用最接近“效果控件”面板顶层的效果。示例：`effect("Fast Blur")("Blurriness")`

effect(index) 返回类型：效果。参数类型：*index* 是数值。After Effects 在“效果控件”面板中按照索引查找效果，从 1 开始并从顶层计算。

mask(name) 返回类型：蒙版。参数类型：*name* 是一个字符串。名称可以是默认名称或者用户定义的名称。如果多个蒙版具有相同名称，则会使用第一个（最高）蒙版。示例：`mask("Mask 1")`

mask(index) 返回类型：蒙版。参数类型：*index* 是数值。After Effects 在“时间轴”面板中按照索引查找蒙版，从 1 开始并从顶层计算。

图层常规属性和方法（表达式引用）

width 返回类型：数值。返回图层的宽度（以像素为单位）。它与 `source.width` 一样。

height 返回类型：数值。返回图层的高度（以像素为单位）。它与 `source.height` 一样。

index 返回类型：数值。返回合成中图层的索引号。

parent 返回类型：图层、光照或摄像机。返回图层的父图层对象（如果有）。示例：
`position[0] + parent.width`

hasParent 返回类型：布尔值。如果图层有父级，则返回 `true`；如果没有，则返回 `false`。使用 **hasParent** 属性可确定图层是否具有父图层。即使图层目前没有父图层，您也可以使用此属性。例如，以下表达式指示您对其应用父级的图层基于父级的位置摆动。如果图层没有父级，则会基于它自己的位置摆动。如果稍后向图层分配父级，则图层的行为会相应地更改：

```
idx = index;    if (hasParent) {        idx =  
parent.index;    }    thisComp.layer(idx).position.wiggle(5,20)
```

inPoint 返回类型：数值。返回图层的入点（以秒为单位）。

注意：


通常，`outPoint` 的值大于 `inPoint` 的值。但是，如果反转图层时间，则 `inPoint` 的值大于 `outPoint` 的值。同样，“开始时间”的值可以大于“入点”的值。


outPoint 返回类型：数值。返回图层的出点（以秒为单位）。


startTime 返回类型：数值。返回图层的起始时间（以秒为单位）。

hasVideo 返回类型：布尔值。如果图层有视频，则返回 `true`；如果没有，则返回 `false`。

hasAudio 返回类型：布尔值。如果图层有音频，则返回 true；如果没有，则返回 false。

active 返回类型：布尔值。如果“视频”开关  在图层中打开且当前时间处于图层的入点到图层的出点范围内，则返回 true；否则，返回 false。

enabled 返回类型：布尔值。如果“视频”开关  在图层中打开，则返回 true；否则，返回 false。

audioActive 返回类型：布尔值。如果“音频”开关  在图层中打开且当前时间处于图层的入点到图层的出点范围内，则返回 true；否则，返回 false。

sampleImage(point, radius = [.5, .5], postEffect=true, t=time) 返回类型：数组 [4]。
参数类型：point 是数组 [2]，radius 是数组 [2]，postEffect 是布尔值，t 是数值。对图层的颜色和 alpha 通道值进行采样，并返回指定点距离内像素的平均 alpha 加权值：[red, green, blue, alpha]。如果 postEffect 为 true，则采样值是渲染图层上的蒙版和效果后的图层的值；如果 postEffect 为 false，则采样值是渲染蒙版和效果前的图层的值。输入值 point 位于图层空间中；点 [0,0] 是图层中左上角像素的中心。输入值 radius 指定样本中心到采样矩形的边缘的水平和垂直距离。默认值会对一个像素采样。

注意：

postEffect 参数指的是直接应用于图层的效果，而非间接应用的效果，例如调整图层。

注意：

在表达式中使用 sampleImage 不再禁用多重处理。

此示例对一个 4 像素宽 3 像素高的矩形进行采样，以距图层左上角下侧和右侧 100 像素的点为中心。

```
thisComp.layer(1).sampleImage([100, 100], [2, 1.5])
```

Dan Ebberts 提供了如何使用 `sampleImage` 方法的示例，该示例位于 [MotionScript 网站](#) 上。

Todd Kopriva 提供了使用 `sampleImage` 方法和点控制效果在颜色校正期间监控指定点颜色的说明，这些说明位于 [After Effects Region of Interest 博客](#) 上。

图层属性特性和方法（表达式引用）

当您向图层添加蒙版、效果、绘画或文本时，After Effects 会将新属性添加到“时间轴”面板中。这些属性太多，无法在此列出，因此请使用关联器了解在表达式中引用它们的语法。

anchorPoint 返回类型：属性 [2 或 3]。在图层的坐标系（图层空间）中返回图层的锚点值。

position 返回类型：属性 [2 或 3]。在世界空间中返回图层的位置值（如果图层没有父级）。如果图层有父级，则会在父图层的坐标系中（在父图层的图层空间中）返回图层的位置值。

scale 返回类型：属性 [2 或 3]。返回图层的缩放值，表示为百分比。

rotation 返回类型：属性。返回图层的旋转值（以度为单位）。对于 3D 图层，它返回 z 旋转值（以度为单位）。

opacity 返回类型：属性。返回图层的不透明度值，表示为百分比。

audioLevels 返回类型：属性 [2]。返回图层的音频水平属性的值（以分贝为单位）。此值是 2D 值；第一个值表示左音频声道，第二个值表示右音频声道。此值不是源材料的音频轨道的振幅，而是音频水平属性的值，可能会受关键帧影响。

timeRemap 返回类型：属性。如果启用时间重映射，则返回时间重映射属性的值。

marker.key(index) 返回类型：MarkerKey。参数类型：*index* 是数值。返回具有指定索引的图层标记的 MarkerKey 对象。

marker.key(name) 返回类型：MarkerKey。参数类型：*name* 是一个字符串。返回具有指定名称的图层标记的 MarkerKey 对象。*name* 值是标记对话框的注释字段中键入的标记名称，例如，`marker.key("ch1")`。如果图层上的多个标记具有相同名称，则此方法会返回时间（图层时间）最早的标记。标记密钥的值是字符串，不是数值。属性的以下表达式在名称标识的两个标记之间将属性值从 0 渐变到 100：

```
m1 =  
marker.key("Start").time;  
m2 = marker.key("End").time;  
linear(time, m1, m2, 0, 100);
```

marker.nearestKey(t) 返回类型：MarkerKey。参数类型：*t* 是数值。返回时间最接近 *t* 的图层标记。例如，以下表达式返回图层上最接近 1 秒时间的标记的时间：

```
marker.nearestKey(1).time
```

以下表达式返回图层上最接近当前时间的标记的时间：

```
marker.nearestKey(time).time
```

marker.numKeys 返回类型：数值。返回图层上的标记的总数。

name 返回类型：字符串。返回图层的名称。

图层 3D 属性和方法（表达式引用）

orientation 返回类型：属性 [3]。返回 3D 图层的 3D 方向值（以度为单位）。

rotationX 返回类型：属性。返回 3D 图层的 x 旋转值（以度为单位）。

rotationY 返回类型：属性。返回 3D 图层的 y 旋转值（以度为单位）。

rotationZ 返回类型：属性。返回 3D 图层的 z 旋转值（以度为单位）。

lightTransmission 返回类型：属性。返回 3D 图层的透光率属性的值。

castsShadows 返回类型：属性。如果图层投影，则返回值 1.0。

acceptsShadows 返回类型：属性。如果图层接受阴影，则返回值 1.0。

acceptsLights 返回类型：属性。如果图层接受光，则返回值 1.0。

ambient 返回类型：属性。返回百分比形式的环境组件值。

diffuse 返回类型：属性。返回百分比形式的漫射组件值。

specular 返回类型：属性。返回百分比形式的镜面组件值。

shininess 返回类型：属性。返回百分比形式的反光度组件值。

metal 返回类型：属性。返回百分比形式的金属质感组件值。

图层空间变换方法（表达式引用）

使用图层空间变换方法将值从一个空间变换到其他空间，例如从图层空间到世界空间。

“from”方法可将值从命名空间（合成或世界）变换到图层空间。“to”方法可将值从图层空间变换到命名空间（合成或世界）。每个变换方法各采用一个可选参数来确定计算变换的时间；但是，您几乎始终可以使用当前（默认）时间。

当变换方向矢量（例如两个位置值之间的差值）时使用“Vec”变换方法。当变换点（例如位置）时，使用简单的（非“Vec”）变换方法。合成和世界空间对于 2D 图层是一样的。然而，对于 3D 图层，合成空间与活动摄像机有关，而世界空间独立于摄像机。

toComp(point, t=time) 返回类型：数组 [2 或 3]。参数类型：*point* 是数组 [2 或 3]，*t* 是数值。将点从图层空间变换到合成空间。

fromComp(point, t=time) 返回类型：数组 [2 或 3]。参数类型：*point* 是数组 [2 或 3]，*t* 是数值。将点从合成空间变换到图层空间。3D 图层中生成的点可能有非零值，即使它位于图层空间中。示例：`fromComp(thisComp.layer(2).position)`

toWorld(point, t=time) 返回类型：数组 [2 或 3]。参数类型：*point* 是数组 [2 或 3]，*t* 是数值。将点从图层空间变换到与视角无关的世界空间。示例：

`toWorld.effect("Bulge")("Bulge Center")` Dan Ebberts 在其 [MotionScript 网站](#)上提供了一个使用 `toWorld` 方法仅沿一个轴自动定向图层的表达式。例如，这可用于在保持垂直时让字符从一侧移到另一侧以跟随摄像机。Rich Young 在其 [AE Portal 网站](#)上提供了一系列使用 `toWorld` 方法将摄像机和光与包含 CC 球面效果的图层相连的表达式。

fromWorld(point, t=time) 返回类型：数组 [2 或 3]。参数类型：*point* 是数组 [2 或 3]，*t* 是数值。将点从世界空间变换到图层空间。示例：

`fromWorld(thisComp.layer(2).position)` 有关如何使用此方法的示例，请参阅[表达式示例：在两个图层之间创建凸出](#)。

toCompVec(vec, t=time) 返回类型：数组 [2 或 3]。参数类型：*vec* 是数组 [2 或 3]，*t* 是数值。将矢量从图层空间变换到合成空间。示例：`toCompVec([1,0])`

fromCompVec(vec, t=time) 返回类型：数组 [2 或 3]。参数类型：*vec* 是数组 [2 或 3]，*t* 是数值。将矢量从合成空间变换到图层空间。示例（2D 图层）：
`dir=sub(position, thisComp.layer(2).position);`
`fromCompVec(dir)`

toWorldVec(vec, t=time) 返回类型：数组 [2 或 3]。参数类型：*vec* 是数组 [2 或 3]，*t* 是数值。将矢量从图层空间变换到世界空间。示例：
`p1 = effect("Eye Bulge 1")("Bulge Center");`
`p2 = effect("Eye Bulge 2")("Bulge Center");`
`toWorld(sub(p1, p2))`

fromWorldVec(vec, t=time) 返回类型：数组 [2 或 3]。参数类型：*vec* 是数组 [2 或 3]，*t* 是数值。将矢量从世界空间变换到图层空间。示例：
`fromWorld(thisComp.layer(2).position)`

fromCompToSurface(point, t=time) 返回类型：数组 [2]。参数类型：*point* 是数组 [2 或 3]，*t* 是数值。在从活动摄像机中进行查看时出现的位置将位于合成空间中的点投影到图层表面上的点（零 *z* 值）。此方法有助于设置效果控制点。仅用于 3D 图层。

摄像机属性和方法（表达式引用）

除了 **source**、**effect**、**mask**、**width**、**height**、**anchorPoint**、**scale**、**opacity**、**audioLevels**、**timeRemap** 以及所有材质属性外，“摄像机”对象具有与“图层”对象相同的属性和方法。

pointOfInterest 返回类型：属性 [3]。返回摄像机在世界空间中的目标点值。

zoom 返回类型：属性。返回摄像机的缩放值（以像素为单位）。下面是图层的缩放属性的一个表达式，可在更改图层的 **z** 位置（深度）或摄像机的缩放值时保持帧中图层的相对大小：


```
cam = thisComp.activeCamera;    distance =  
length(sub(position, cam.position));    scale * distance /  
cam.zoom;
```

depthOfField 返回类型：属性。如果摄像机的景深属性打开，则返回 1；如果景深属性关闭，则返回 0。

focusDistance 返回类型：属性。返回摄像机的焦距值（以像素为单位）。

aperture 返回类型：属性。返回摄像机的光圈值（以像素为单位）。

blurLevel 返回类型：属性。返回百分比形式的摄像机模糊层次值。

active 返回类型：布尔值。如果摄像机是当前时间合成的活动摄像机，则返回 **true**：摄像机图层的“视频”开关  打开，当前时间处于范围摄像机图层的入点到摄像机图层的出点范围内，且它是“时间轴”面板中列出的第一个（最高）此类摄像机图层。否则，返回 **false**。

光照属性和方法（表达式引用）

除了 `source`、`effect`、`mask`、`width`、`height`、`anchorPoint`、`scale`、`opacity`、`audioLevels`、`timeRemap` 以及所有材质属性外，“光照”对象具有与“图层”对象相同的属性和方法。

pointOfInterest 返回类型：属性 [3]。返回光照在世界空间中的目标点值。

intensity 返回类型：属性。返回百分比形式的光照强度值。

color 返回类型：属性 [4]。返回光的颜色值。

coneAngle 返回类型：属性。返回光的锥形角度（以度为单位）。

coneFeather 返回类型：属性。返回百分比形式的光的锥形羽化值。

shadowDarkness 返回类型：属性。返回百分比形式的光的阴影深度值。

shadowDiffusion 返回类型：属性。返回光的阴影扩散值（以像素为单位）。

David Van Brink 在其 [omino pixel 博客](#) 上提供了演示如何将表达式与光结合使用的指导文章和示例项目。

效果属性和方法（表达式引用）

active

返回类型：布尔值。如果效果打开（“效果”开关  处于选定状态），则返回 true。

param(name) 返回类型：属性。参数类型：*name* 是一个字符串。返回效果中的属性。效果控制点始终位于图层空间中。示例：`effect("Bulge").param("Bulge Height")`

param(index) 返回类型：属性。参数类型：*index* 是数值。返回效果中的属性。效果控制点始终位于图层空间中。例如，`effect("Bulge").param(4)` 返回凸出高度属性。

蒙版属性和方法（表达式引用）

注意：

您可以将蒙版路径属性与其他路径属性（形状图层和笔刷笔触的路径）相连，但是无法访问这些属性以便通过表达式直接进行数值处理。

maskOpacity 返回类型：属性。返回百分比形式的蒙版的不透明度值。

maskFeather 返回类型：属性。返回蒙版的羽化值（以像素为单位）。

maskExpansion 返回类型：属性。返回蒙版的扩展值（以像素为单位）。

invert 返回类型：布尔值。如果蒙版已反转，则返回 true；如果未反转，则返回 false。

属性特性和方法（表达式引用）

value 返回类型：数值、数组或字符串。返回属性在当前时间的值。

valueAtTime(t) 返回类型：数值或数组。参数类型：*t* 是数值。返回属性在指定时间（以秒为单位）的值。例如，要从一组（四个）值中随机选择每个帧的属性值，请在 0、1、2 和 3 秒将您的四个值设为关键帧，然后将以下表达式应用于该属性：

```
valueAtTime(random(4))
```

注意：

Dan Ebberts 在其 [MotionScript 网站](#)上提供了使用 valueAtTime 和 velocityAtTime 方法的更多示例和技术。

velocity 返回类型：数值或数组。返回当前时间的临时速度值。对于空间属性（例如位置），它返回正切矢量值。结果与属性的维度相同。

velocityAtTime(t) 返回类型：数值或数组。参数类型：*t* 是数值。返回指定时间的临时速度值。

speed 返回类型：数值。返回 1D，等于属性在默认时间更改的速度的正速度值。此元素只能用于空间属性。

speedAtTime(t) 返回类型：数值。参数类型：*t* 是数值。返回指定时间的空间速度值。

wiggle(freq, amp, octaves=1, amp_mult=.5, t=time)

返回类型：数值或数组。参数类型：*freq*、*amp*、*octaves*、*amp_mult* 和 *t* 是数值。随机摇动（摆动）属性值。*freq* 值是每秒摆动的频率。*amp* 值是向其应用了此值的属性单位中的振幅。*octaves* 是要加在一起的噪声的八度数。此值控制摆动的详细程度。使此值高于默认值 1 可在摆动中包括更高的频率，使此值低于默认值 1 可在摆动中包括振幅谐波。*amp_mult* 是 *amp* 乘以每个八度的值。此值控制谐波减弱的速度。默认为 0.5；使其更接近 1 可以与基本频率相同的振幅添加谐波，使其更接近 0 可以添加更少的细节。*t* 是基础启动时间。此值默认为当前时间。如果您希望输出成为在不同时间采样的属性值的摆动，请使用此参数。示例：`position.wiggle(5, 20, 3, .5)` 每秒产生约 5 次摆动，平均大小约 20 像素。除了主要摆动之外，其他两个级别的详细摆动发生的频率为每秒 10 次和 20 次摆动，各自的大小为 10 和 5 像素。对于二维属性（例如缩放），以下示例按相同值摆动两个维度：`v = wiggle(5, 10); [v[0], v[0]]` 此示例具有两维属性，只能在 Y 轴上摇摆：`freq = 3; amp = 50; w = wiggle(freq, amp); [value[0], w[1]]`；Dan Ebberts 在其 [MotionScript 网站](https://motionscript.com/) 上提供了演示如何使用 `wiggle` 方法的时间参数创建循环动画的示例表达式和详细说明。

temporalWiggle(freq, amp, octaves=1, amp_mult=.5, t=time) 返回类型：数值或数组。参数类型：*freq*、*amp*、*octaves*、*amp_mult* 和 *t* 是数值。在摆动的时间对属性进行采样。*freq* 值是每秒摆动的频率，*amp* 是向其应用了此值的属性单位中的振幅，*octaves* 是要加在一起的噪声八度数，*amp_mult* 是 *amp* 乘以每个八度的值，*t* 是基础启动时间。为使此函数有意义，必须对其采样的属性制作动画，因为此函数只更改采样时间而不是值。示例：`scale.temporalWiggle(5, .2)`

smooth(width=.2, samples=5, t=time) 返回类型：数值或数组。参数类型：*width*、*samples* 和 *t* 是数值。随着时间的推移平滑属性值，将值的大而短的偏差转换为更小、分布更均匀的偏差。此平滑通过在指定时间向属性值应用框滤镜来完成。*width* 值是平均滤镜的时间（以秒为单位）范围。*samples* 值是随着时间均匀分布的分离样本的数目；为更大的平滑度使用更大的值（但性能下降）。通常，您会希望 *samples* 是奇数，以便当前时间的值包括在平均数中。示例：`position.smooth(.1, 5)`

loopIn(type="cycle", numKeyframes=0) 返回类型：数值或数组。循环从向图层的出点前进的图层的第一个关键帧中计算的时间段。该循环从图层的入点播放。numKeyframes 值确定循环的段：循环的段是图层从第一个关键帧到 numKeyframes+1 个关键帧的部分。例如，**loopIn("cycle", 3)** 循环以第一个和第四个关键帧为界的段。默认值为 0 意味着所有关键帧都会循环。您可以使用关键帧循环方法来重复一系列关键帧。您可以对多数属性使用这些方法。例外包括无法用“时间轴”面板中的简单数值表示的属性，例如源文本属性、路径形状属性以及色阶效果的直方图属性。过大的关键帧或持续时间值修剪为最大允许值。过小的值会导致不变的循环。

| 循环类型 | 结果 |
|-----------------|--|
| cycle | (默认) 重复指定段。 |
| pingpong | 重复指定段，向前和向后交替。 |
| offset | 重复指定段，但会按段开始和结束时属性值的差异乘以段已循环的次数偏移每个周期。 |
| continue | 不重复指定段，但继续基于第一个或最后一个关键帧的速度对属性进行动画制作。例如，如果图层的缩放属性的最后一个关键帧是 100%，则图层将继续从 100% 缩放到出点，而不是直接循环回出点。此类型不接受 keyframes 或 duration 参数。 |

loopOut(type="cycle", numKeyframes=0) 返回类型：数值或数组。循环从向图层的入点后退的图层的最后一个关键帧中计算的时间段。循环一直播放到图层的出点。要循环的片段由指定数量的关键帧决定。numKeyframes 值设置要循环的关键帧段的数量；指定范

围从最后一个关键帧向后进行计算。例如，`loopOut("cycle", 1)` 循环以最后一个关键帧和倒数第二个关键帧为界的段。默认值为 0 意味着所有关键帧都会循环。有关更多信息，请参阅 `loopIn` 的项。David Van Brink 在 [omino pixel 博客](#) 上提供了一篇指导性文章和一个示例项目，介绍如何使用回声效果、粒子运动场效果和 `loopOut` 方法，对一群程式化的游动细菌进行动画制作。

loopInDuration(type="cycle", duration=0) 返回类型：数值或数组。循环从向图层的出点前进的图层的第一个关键帧中计算的时间段。该循环从图层的入点播放。要循环的片段由指定的持续时间决定。duration 值设置要循环的段中的合成秒数；指定的范围从第一个关键帧进行计算。例如，`loopInDuration("cycle", 1)` 循环整个动画的第一秒。默认的 0 意味着要循环的段开始于图层出点。有关更多信息，请参阅 `loopIn` 的项。

loopOutDuration(type="cycle", duration=0) 返回类型：数值或数组。循环从向图层的入点后退的图层的最后一个关键帧中计算的时间段。循环一直播放到图层的出点。要循环的片段由指定的持续时间决定。duration 值设置要循环的段中的合成秒数；指定的范围从最后一个关键帧向前进行计算。例如，`loopOutDuration("cycle", 1)` 循环整个动画的最后一秒。默认的 0 意味着要循环的段开始于图层入点。有关更多信息，请参阅 `loopIn` 的项。

key(index) 返回类型：Key 或 MarkerKey。参数类型：*index* 是数值。按数字返回 Key 或 MarkerKey 对象。例如，`key(1)` 返回第一个关键帧。

key(markerName) 返回类型：MarkerKey。参数类型：*markerName* 是字符串。返回具有此名称的 MarkerKey 对象。仅对标记属性使用。

nearestKey(t) 返回类型：Key 或 MarkerKey。返回最接近指定时间的 Key 或 MarkerKey 对象。

numKeys 返回类型：数值。 返回属性的关键帧数目。返回标记属性的标记数目。

注意:

如果您使用“分离维度”命令将位置属性的维度分离到各个组件，则关键帧数会更改，因此，此方法返回的值会更改。

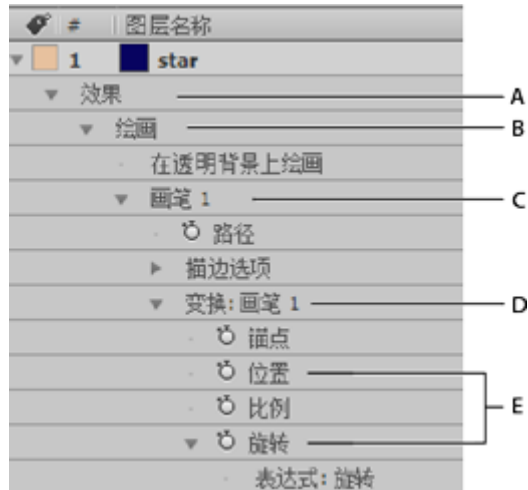
propertyGroup(countUp = 1) 返回类型：组。返回与对其编写表达式的属性相关的属性组。例如，如果您将 **propertyGroup(1)** 表达式添加到笔刷笔触的旋转属性，则该表达式会瞄准包含旋转属性的变换属性组。如果您改为添加 **propertyGroup(2)**，则该表达式会瞄准笔刷属性组。此方法使您可在属性层次结构中建立独立于名称的关系。当复制包含表达式的属性时，此方法特别有用。**propertyGroup** 的 **numProperties** 方法返回属性组中属性的数目。此示例返回包含对其编写表达式的属性的组中的属性数

目：
`thisProperty.propertyGroup(1).numProperties`

propertyIndex 返回类型：数值。返回与其属性组中的其他属性相关的属性的索引，包括蒙版、效果、文本动画、选择器、形状、跟踪器以及跟踪点中的属性组。

name 返回类型：字符串。返回属性或属性组的名称。

示例：使用 propertyGroup 方法和 propertyIndex 属性进行动画制作

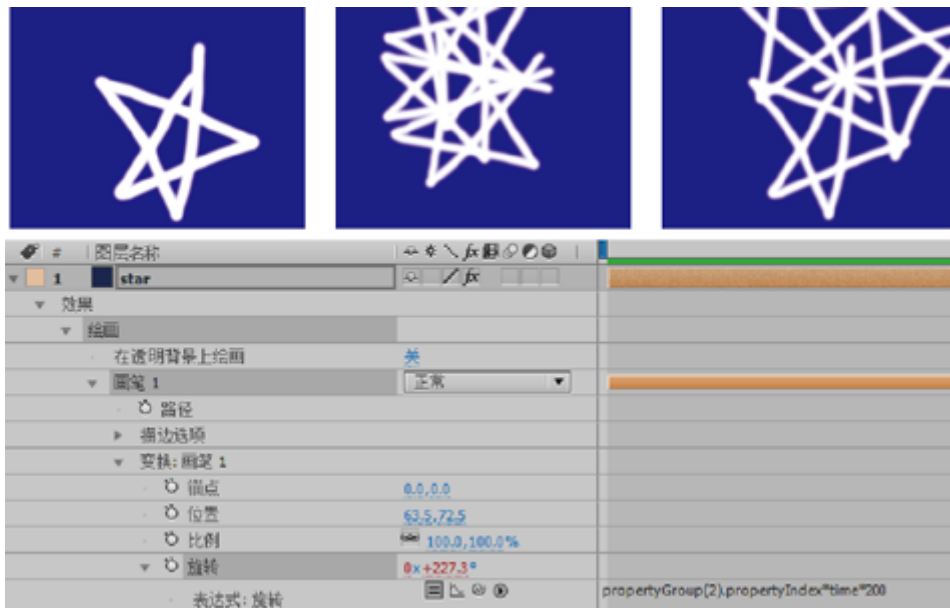


与笔刷笔触的位置属性相关的 `propertyGroup` 的值

A. `propertyGroup(4)` **B.** `propertyGroup(3)` **C.** `propertyGroup(2)` **D.** `propertyGroup(1)` **E.** Position `propertyIndex` 值为 2 ; Rotation `propertyIndex` 值为 4。

在此示例中，每个笔刷笔触的 `propertyGroup` 方法都会瞄准笔刷属性组，因为该组是旋转属性下面的两个属性组。每个笔刷笔触中的 `propertyIndex` 属性稍后会为每个笔刷笔触返回一个唯一值。生成的值稍后会乘以时间和 200 并应用于每个旋转值，以不同的方式旋转每个笔刷笔触，创建旋转的绘画笔触：
`propertyGroup(2).propertyIndex * time * 200`

`propertyGroup(2).propertyIndex * time * 200`



使用表达式对笔刷笔触进行动画制作

项目属性（表达式引用）

表达式方法：

- `thisProject` 对象 `thisProject` - 代表包含表达式的项目。

类型：

项目对象；只读

- 项目 `fullPath` 属性 `thisProject.fullPath` - 平台特定的绝对文件路径，包括项目名称。如果项目未保存，将返回空字符串。

类型:

字符串；只读。

- 项目 `bitsPerChannel` 属性 `thisProject.bitsPerChannel` - 项目的颜色深度，以每通道位数 (bpc) 为单位，如“项目设置”>“颜色管理”中所设置。值为 8、16 或 32 之一。与脚本项目属性 `app.project.bitsPerChannel` 等效。

类型

数字；只读。

- 项目 `linearBlending` 属性 `thisProject.linearBlending` - “项目设置”>“颜色管理”中的“使用 1.0 灰度系数混合颜色”选项的状态。与脚本项目属性 `app.project.linearBlending` 等效。

类型:

布尔值；只读。

Key 属性和方法（表达式引用）

当您访问 Key 对象时，您可从中获得 `time`、`index` 和 `value` 属性。例如，以下表达式为您提供三个 Position 关键帧的值：`position.key(3).value`。

在对具有关键帧的不透明度属性编写时，以下表达式将忽略关键帧值并且仅使用关键帧的时间放置来确定闪光应发生的位置：
`d = Math.abs(time - nearestKey(time).time); easeOut(d, 0, .1, 100, 0)`

```
d = Math.abs(time - nearestKey(time).time);  
easeOut(d, 0, .1, 100, 0)
```

value 返回类型：数值或数组。返回关键帧的值。

time 返回类型：数值。返回关键帧的时间。

index 返回类型：数值。返回关键帧的索引。

MarkerKey 属性（表达式引用）

您可以使用相同方法访问合成标记和图层标记的值。通过 `thisLayer.marker` 对象访问图层标记；通过 `thisComp.marker` 对象访问合成标记。

考虑到表达式的用途，标记是一种特殊类型的 Key 对象，因此，您可以使用 `nearestKey(time)` 等方法访问标记，且标记还具有 `time` 和 `index` 属性。`index` 属性不是标记的编号（名称）；它是关键帧索引号，表示时间标尺中标记的顺序。

表达式可以使用可在“合成标记”或“图层标记”对话框中设置的标记的所有值。文本图层的源文本属性的以下表达式显示最接近当前时间的图层标记的时间、持续时间、注释（名称）、章节、URL、帧目标以及提示点名称，以及标记是否为事件提示点的标记：

```
m = thisLayer.marker.nearestKey(time);
s = "time:" + timeToCurrentFormat(m.time) + "\r" +
    "duration: " + m.duration + "\r" +
    "key index: " + m.index + "\r" +
    "comment:" + m.comment + "\r" +
    "chapter:" + m.chapter + "\r" +
    "URL:" + m.url + "\r" +
    "frame target: " + m.frameTarget + "\r" +
    "cue point name: " + m.cuePointName + "\r" +
    "Event cue point? " + m.eventCuePoint + "\r";
```

```
for (param in m.parameters){  
    s += "parameter: " + param + " value: " + m.parameters[param] + "\r";  
}  
s
```

因为素材项目中的 XMP 元数据可以基于该项转换为图层的图层标记，所以表达式可与 XMP 元数据进行交互。有关信息，请参阅 [After Effects 中的 XMP 元数据](#)。

Dan Ebberts 在 [After Effects 开发人员中心](#) 提供了一个教程，其中包括将 XMP 元数据与表达式结合使用的示例。

duration 返回类型：数值。 标记的持续时间（以秒为单位）。

comment 返回类型：字符串。 标记对话框中的注释内容字段。

chapter 返回类型：字符串。 标记对话框中的章节内容字段。

url 返回类型：字符串。 标记对话框中的 URL 内容字段。

frameTarget 返回类型：字符串。 标记对话框中的帧目标内容字段。

eventCuePoint 返回类型：布尔值。 标记对话框中的提示点类型设置。对于事件为 True；对于导航为 False。

cuePointName 返回类型：字符串。 标记对话框中的提示点名称内容字段。

parameters 返回类型：字符串值的关联数组。 标记对话框中的参数名称和参数值内容字段。例如，如果您有名为“背景颜色”的参数，则您可以使用以下表达式在最近的标记中访问

其值：`thisComp.marker.nearestKey(time).parameters["background color"]`

MarkerValue.protectedRegion (表达式引用)

表达式方法：

`thisComp.marker.key(index).protectedRegion`

描述：

此值为 True 时，合成标记可看作受保护区域。

类型：

布尔值；只读。

访问有关形状、蒙版和画笔描边的路径点的表达式 (表达式引用)

您可以针对以下对象使用表达式读写路径点或顶点的 x 坐标和 y 坐标：

- 图层蒙版
- 贝塞尔曲线形状
- 针对“绘画”和“Roto 笔刷和优化边缘”效果的画笔描边。

表达式方法：

- **Path points() 方法：**`{pathProperty}.points(t = time)` 获取路径上所有点的 x、y 坐标。图层蒙版路径点的坐标相对于其左上角的图层的原点。贝塞尔曲线形状路

径点的坐标相对于路径的形状组的锚点，例如，“**变换：形状 1**”>“**锚点**”。笔刷笔触路径点的坐标与笔触的开始相关；第一个点为 [0,0]。（可选）指定采样到路径的时间。复制路径时，您可以将此方法传递到点参数的 `createPath()` 方法。

参数：

| | |
|---|---------------------------------------|
| t | （可选）数字。采样路径的合成时间（以秒为单位）。默认值为时间（当前时间）。 |
|---|---------------------------------------|

返回：

数偶数组的数组，四舍五入到小数点后四位。

- **Path inTangents() 方法：**`{pathProperty}.inTangents(t = time)` 获取路径上所有点的入点手柄的 x、y 坐标。切点坐标值是相对于父点的坐标的位移。即，值 [0,0] 在入点处不产生弯度。当复制路径时，可以针对 `inTangents` 参数将此方法传递给 `createPath()` 方法。（可选）指定采样到路径的时间。

参数：

| | |
|---|---------------------------------------|
| t | （可选）数字。采样路径的合成时间（以秒为单位）。默认值为时间（当前时间）。 |
|---|---------------------------------------|

返回：

数偶数组的数组，四舍五入到小数点后四位。

- **Path outTangents() 方法：**`{pathProperty}.outTangents(t = time)` 获取路径上所有点的出点手柄的 x、y 坐标。切点坐标值是相对于父点坐标的位移 - 值 [0,0] 在引出切点处不产生弯度。当复制路径时，可以针对 `outTangents` 参数将此方法传递给 `createPath()` 方法。（可选）指定采样到路径的时间。

参数：

| | |
|---|--|
| t | (可选) 数字。采样路径的合成时间 (以秒为单位)。默认值为时间 (当前时间)。 |
|---|--|

返回：

数偶数组的数组，四舍五入到小数点后四位。

- **Path isClosed() 方法**：`{pathProperty}.isClosed()` 决定路径是开放的还是闭合的。如果路径是闭合的，则返回 `true`；如果路径是开放的，则返回 `false`。当复制路径时，可以针对 `is_closed` 参数将此方法传递给 `createPath()` 方法。

参数：

无

返回：

布尔值

- **Path pointOnPath() 方法**：`{pathProperty}.pointOnPath(percentage = 0.5, t = time)` 获取路径上任意点的 x、y 坐标。该点表示为路径弧长的百分比。第一个点为 0%，最后一个点为 100%。如果路径是闭合的，那么 0% 和 100% 将返回相同的坐标。弧长的百分比用于确保沿路径的速度一致。除了 0% 和 100% 之外，百分比并非必须与路径上的贝塞尔曲线点对应 - 对于三个点的路径，第二点并非必须在 50%。这也意味着对于具有相同点的开放路径和闭合路径，由于闭合路径含有附加长度，因此沿开放路径的百分比将不会返回与闭合路径相同的坐标。(可选) 指定采样到路径的时间。

参数：

| | |
|-----|---|
| 百分比 | (可选) 0 到 1 之间的数字。采样沿路径的弧长的百分比。值小于 0 和大于 1 的部分会被剪掉。默认值为 0.5。 |
| | |

| | |
|---|--|
| t | (可选) 数字。采样路径的合成时间 (以秒为单位)。默认值为时间 (当前时间)。 |
|---|--|

返回:

数偶数组。

- **Path tangentOnPath() 方法** {pathProperty}.tangentOnPath(percentage = 0.5, t = time) 获取路径上经过计算的任意点的出点手柄的 x、y 坐标。切点坐标值是相对于父点的坐标的位移 - 值 [0,0] 在出点处不产生弯度。入点手柄是此值的负值 (将 x、y 坐标乘以 -1)。切线的父点表示为路径的弧长的百分比。如果用户定义的点也在该弧长百分比的范围内, 那么 tangentOnPath() 返回的坐标从其父点开始计算, 并且将与 outTangents() 返回的那些坐标有所不同。父点的坐标与 tangentOnPath() 坐标之间的直线距离将始终为 1。您可以将返回的坐标相乘以产生更长的切线, 例如, (myPath.tangentOnPath()*100)。(可选) 指定采样到路径的时间。

参数:

| | |
|-----|---|
| 百分比 | (可选) 0 到 1 之间的数字。采样沿路径的弧长的百分比。值小于 0 和大于 1 的部分会被剪掉。默认值为 0.5。 |
| t | (可选) 数字。采样路径的合成时间 (以秒为单位)。默认值为时间 (当前时间)。 |

返回:

数偶数组。

- **Path normalOnPath() 方法** {pathProperty}.normalOnPath(percentage = 0.5, t = time) 获取路径上经过计算的任意点的法线的 x、y 坐标。法线的切点坐

标值是相对于父点的坐标的位移 - 值 [0,0] 与父点相同。法线的父点表示为路径的弧长的百分比。请阅读 `pointOnPath()` 方法的说明以了解有关弧长百分比的详细信息。
`normalOnPath()` 返回的坐标从其父点开始计算。父点的坐标与 `normalOnPath()` 坐标之间的直线距离将始终为 1。您可以将返回的坐标相乘以产生更长的法线，例如，
`(myPath.normalOnPath()*100)`。(可选) 指定采样到路径的时间。

参数：

| | |
|-----|---|
| 百分比 | (可选) 0 到 1 之间的数字。采样沿路径的弧长的百分比。值小于 0 和大于 1 的部分会被剪掉。默认值为 0.5。 |
| t | (可选) 数字。采样路径的合成时间 (以秒为单位)。默认值为时间 (当前时间)。 |

返回：

数偶数组。

- **Path createPath() 方法** `{pathProperty}.createPath(points = [[0,0], [100,0], [100,100], [0,100]], inTangents = [], outTangents = [], is_closed = true)` 根据一组点和切线创建路径对象。这些点由表示其 x、y 坐标的数偶数组的数组定义。数组的长度必须至少为 1，可以为任意更大的长度。点的入点手柄和出点手柄由表示其 x、y 位移坐标的数偶数组的数组定义。切线数组的长度必须与点参数完全相同。切点坐标值是相对于父点的坐标的位移 - 值 [0,0] 在入点处不产生弯度。可以将路径的 `points()`、`inTangents()`、`outTangents()` 和 `isClosed()` 方法传递给 `points`、`inTangents`、`outTangents` 和 `is_closed` 参数来复制路径。可以将同一路径的 `points` 和 `tangents` 传递给 `createPath()`，经过修改即可生成不同结果。例如，以下表达式将通过不传递 `inTangents` 或 `outTangents` 参数从“蒙版 1”删除曲线：

```
myMask = mask("Mask 1").path;
myMask.createPath(myMask.points());
```

下面的示例通过传递“蒙版 1”的 points 和 tangents，并通过将 `is_closed` 设置为 `false` 来将其转换为开放路径：

```
myMask = mask("Mask 1").path;
myMask.createPath(myMask.points(), myMask.inTangents(),
myMask.outTangents(), false);
```

| | |
|-------------|---|
| points | 长度为 1 或更大的数组包含表示路径点的 [x,y] 坐标的数偶数组。必需，除非不传递任何参数（例如， <code>createPath()</code> ）。默认值为 <code>[[0,0], [100,0], [100,100], [0,100]]</code> 。 |
| is_closed | （可选）布尔值。决定蒙版是否为闭合。如果为 <code>true</code> ，则会将最后一个点连接到第一个点。默认值为 <code>true</code> 。 |
| inTangents | 包含数偶数组的数组，表示路径点的出点手柄的 [x, y] 位移坐标。必需，除非不传递任何参数（例如， <code>createPath()</code> ）。数组长度必须与 points 长度相同，或者您可以传递一个空数组 (<code>[]</code>)，系统会假定该数组长度与 points 长度相同，并为所有切线设置 <code>[0,0]</code> 。默认值为一个空数组。 |
| outTangents | 包含数偶数组的数组，表示路径点的入点手柄的 [x, y] 位移坐标。必需，除非不传递任何参数（例如， <code>createPath()</code> ）。数组长度必须与 points 长度相同，或者您可以传递一个空数组 (<code>[]</code>)，系统会假定该数组长度与 points 长度相同，并为所有切线设置 <code>[0,0]</code> 。默认值为一个空数组。 |

返回：

路径对象。

示例：

- 示例 1

该示例在 time=0 时将点和切线坐标列表从“形状图层 1”图层的“形状 1”的“路径 1”写入字符串。将其应用于文本图层的源文本属性，以读出形状的坐标，以及入点和出点。

```
pointsList = "";
sampleTime = 0;
myShape = thisComp.layer("Shape Layer 1").content("Shape
1").content("Path 1").path;

for (i = 0; i < myShape.points(sampleTime).length; i++) {
    pointsList += "c: " + myShape.points(sampleTime)[i].toString()
+ " i: " + myShape.inTangents(sampleTime)[i].toString() + " o: "
+ myShape.outTangents(sampleTime)[i].toString() + "\n";
}

pointsList;
```

- 示例 2

该示例会读取“纯色深灰 1”上的“蒙版 1”的第一个顶点的坐标，并将其转换为合成坐标。将其应用于效果的 2D 点控制，例如“写入”或“CC 粒子仿真系统 II”，以便效果追踪或跟踪动画蒙版的第一个点。复制效果并更改路径点指引值 ([0]) 以追踪或跟踪蒙版的其他点。

```
myLayer = thisComp.layer("Dark Gray Solid 1");
myLayer.toComp(myLayer.mask("Mask 1").maskPath.points()[0]);
```

数据驱动动画（表达式引用）

表达式方法：

- **素材 sourceText 属性 {footageItem}.sourceText** 返回 JSON 文件的内容作为字符串。eval() 方法可用于将字符串转换为一个 sourceData 对象数组，与 sourceData 属性的结果相同，单个数据流可通过其被引用为数据的层次结构属性。例如：

```
var myData = eval(footage("sample.json").sourceText);  
  
myData.sampleValue;
```

类型:

字符串，.JSON 文件的内容；只读。

- **素材 sourceData 属性 {footageItem}.sourceData** 返回 .JSON 文件的数据作为 sourceData 对象的一个数组。.JSON 文件的结构将决定数组的大小和复杂度。单个数据流可以被引用为数据的层次结构属性。例如，对于名为 Color 的数据流，以下属性将从第一个数据对象返回 Color 的值：

```
footage("sample.json").sourceData[0].Color
```

典型用例是将 .JSON 文件的 sourceData 分配到变量，然后引用所需的数据流。例如：

```
var myData = footage("sample.json").sourceData;  
  
myData[0].Color;
```

类型:

一个 sourceData 对象的数组；只读。

- **素材 dataValue()方法 {footageItem}.dataValue(dataPath)** 在 .mgJSON 文件中返回指定静态或动态数据流的值。接受单数组值以将层次结构中的路径定义到所需的数据流。例如：

`footage("sample.mgjson").dataValue([0])` 返回第一个子项的数据。

`footage("sample.mgjson").dataValue([1][0])` 在第二个组中返回第一个子项的数据。

参数:

| | |
|----------|--------------------------|
| dataPath | 数组，必选。层次结构中到静态或动态数据流的路径。 |
|----------|--------------------------|

返回:

数据流的值。

- **素材 dataKeyCount() 方法** {footageItem}.dataKeyCount(dataPath) 返回 .mgJSON 文件中指定静态数据流中的样本数量。接受单数组值以将层次结构中的路径定义到所需的静态数据流。

例如：

- footage("sample.mgjson").dataKeyCount([0]) 返回第一个子项的样本计数
- footage("sample.mgjson").dataKeyCount([1][0]) 返回第二个组的样本计数

参数：

| | |
|----------|--------------------------|
| dataPath | 数组，必选。层次结构中到静态或动态数据流的路径。 |
|----------|--------------------------|

返回：

动态数据流中的样本数。

- **素材 dataKeyTimes() 方法** {footageItem}.dataKeyTimes(dataPath, t0 = startTime, t1=endTime) 返回 .mgJSON 文件中指定动态数据流的采样时间（以秒为单位）。也可指定返回样本的时间跨度。默认情况下，返回动态数据流中 startTime 和 endTime 之间所有样本的时间，如 .mgJSON 文件中数据流的 samplesTemporalExtent 属性所定义。接受单数组值以将层次结构中的路径定义到所需的动态数据流。

以下示例返回第一个子项在 1 秒和 3 秒之间的样本时间：

```
footage("sample.mgjson").dataKeyTimes([0], 1, 3)
```

参数：

| | |
|----------|---|
| dataPath | 数组，必选。层次结构中到动态数据流的路径。 |
| t0 | (可选) 数字。返回样本的时间跨度的开始时间 (以秒为单位)。默认设置为 startTime。 |
| t1 | (可选) 数字。返回样本的时间跨度的结束时间 (以秒为单位)。默认设置为 endTime。 |

返回：

表示采样时间的数组。

- **素材 dataKeyValues() 方法** {footageItem}.dataKeyValues(dataPath, t0 = startTime, t1=endTime) 返回 .mgJSON 文件中指定动态数据流中的样本值。也可指定返回样本的时间跨度。默认情况下，返回动态数据流中 startTime 和 endTime 之间所有样本的时间，如 .mgJSON 文件中数据流的 samplesTemporalExtent 属性所定义。接受单数组值以将层次结构中的路径定义到所需的动态数据流。

例如：

footage("sample.mgjson").dataKeyValues([0], 1, 3) 返回第一个子项在 1 秒和 3 秒之间的采样值。

参数：

| | |
|----------|--------------------------------------|
| dataPath | 数组，必选。层次结构中到动态数据流的路径。 |
| t0 | (可选) 数字。返回样本的时间跨度的开始时间 (以秒为单位)。默认设置为 |

| | |
|----|---|
| | startTime。 |
| t1 | (可选) 数字。返回样本的时间跨度的结束时间 (以秒为单位)。默认设置为 endTime。 |

返回：

表示样本值的数组。

十六进制转 RGB 颜色转换方法 (表达式引用)

hexToRgb() 这种颜色转换方法，可将十六进制颜色值（例如#FF00FF）转换为 RGBA 颜色值。需要将颜色参数关联到 JSON 或 CSV/TSV 数据源中以十六进制字符串形式表示的颜色值时，此功能非常有用。

表达式方法：

hexToRgb(hexString) 将表示颜色的三个二位十六进制数组合转化为 RGB，或将四个二位十六进制数组合转化为 RGBA 空间。对于三个二位十六进制数组合，其 Alpha 值默认为 1.0。

参数：

| | |
|-----------|---|
| hexString | 表示三个二位十六进制数组合（6 个数位，无 Alpha 通道）或四个二位十六进制数组合（8 个数位，有 Alpha 通道）的字符串，只包含数字或 A–F 的字母。可选的开头字符 0x、0X 或 # 会被忽略。超过 8 个数位的字符都将被忽略。 |
|-----------|---|

返回：

RGBA 颜色值数组。

示例：

以下所有值都将返回 [1.0, 0.0, 1.0, 1.0]：

- `hexToRgb("FF00FF")`
- `hexToRgb("#FF00FF")`
- `hexToRgb("0xFF00FF")`
- `hexToRgb("0XFF00FFFF")` 注意：四个二位十六进制数组合的 8 个数位中，最后两个数位表示将 Alpha 设为 1.0。

更多此类内容

[表达式基础知识](#)

[表达式示例](#)

[时间码和时间显示单位](#)

[关键帧插值](#)

[自动放慢速度](#)

[摄像机、光和目标点](#)



Twitter™ 与 Facebook 中的内容不在 Creative Commons 的条款约束之下。

[法律声明](#) | [在线隐私策略](#)



ADOBE AFTER EFFECTS

[< 查看所有应用程序](#)

[学习和支持](#)

[开始使用](#)

[用户指南](#)

[教程](#)

在社区提问

发表问题并获得专家解答。

[立即提问](#)

联系我们

来自真正专业人士的真心实意的帮助。

[立即开始](#)

此页面是否有帮助？

☐ 是

☐ 否