



# 优化 Flash Lite 2.0 的内容

Josh Ulm  
首席设计师  
体验主管，移动和设备部门

对本文有贡献的人员：  
Rosalind Morrison, Adobe Consulting; Walter Luh, Flash Lite Engineering; Jian Zheng, Flash Lite Engineering; Jeremy Clark, XD

# 目录

简介 .....	3
已知问题 .....	4
帧速率 .....	4
需要记住的事项 .....	5
工作区大小和发布配置文件 .....	5
时间间隔和监听器 .....	5
跳帧 .....	5
全局和本地变量 .....	5
匿名函数 .....	5
缩放和旋转位图 .....	5
渐变色带 .....	5
优化内容 .....	6
艺术作品格式 .....	6
压缩的 JPG .....	6
透明的 PNG .....	6
渐变 .....	6
矢量复杂性 .....	6
矢量角和曲线 .....	6
形状轮廓 .....	6
嵌入的文本 .....	6
多线动态文本 .....	7
隐藏电影剪辑 .....	7
变形处理区域 .....	7
图层类型 .....	7
清除源文件 .....	7
首帧初始化 .....	7
数学和浮点数 .....	7
数学例程数据 .....	8
循环迭代 .....	8
XML 数据 .....	8
关于作者 .....	9

# 简介

本文汇集了使用 Flash Lite 2.0 为移动设备创建轻巧、快速的 Flash 内容的许多技巧和见解。

大多数 Flash 开发人员都很了解优化 Flash 内容的普遍真理，比如不要用动画处理巨大的复杂艺术作品，不要一次对数量庞大的内容进行变形处理，以及不要过度使用透明度。在台式机上，Flash 8（最终！）解决了这些性能问题。但是，Flash Lite 开发人员面临着一个更加复杂的情形。在那些我们刚好没有精力去测试的设备上，我们需要得到疯狂的好莱坞效果，至少这种情况下还没有解决这些性能问题。而且，有些设备执行得比其他设备好，有时这种情况会非常引人注目。而且因为移动创作经常需要我们发布到许多不同的设备上，这种情况下，可能我们必须为最低的共有设备条件进行创作。那就是说，此处稍有不同，在别处可能会完全不一样。

许多建议都会指导您避免使用某些公共创作技术。显然，如果没有矢量、动态文本和动画，Flash 就不成其为 Flash，因此不要因为这些功能是禁区就失望地走开，它们不是禁区。但您将必须试验哪些功能可以运行，哪些功能不能运行，然后才能知道究竟如何。在您阅读本文档时，请记住优化移动内容经常需要衡量所付出的代价。技术“A”可能看起来较好，而技术“B”会产生好得多的性能。要达到技术“A”或技术“B”，您将需要了解并预料到优化移动内容的过程需要在目标设备上大量的反复测试。在实际的硬件上测试您的文件无可替代；不存在任何其他方法来确认实际的性能、真实的颜色、文本可读性、物理交互、界面响应性以及最终的实际移动体验。

# 已知问题

## 帧速率

主机的定时器分辨率从根本上影响在该设备上 Flash Lite 可获得的最大帧速率。因此，在创作层上指定的 fps（预期的 fps）将不会转换为实际的 fps，即使对于空的 SWF（没有任何呈现或 AS）也是这样！会存在需要对预期的 fps 所做的调整，以获得您应从您的内容预期的最大 fps。

因此，请将帧速率设置为目标设备的原生 fps。如果您不确信您的目标设备的理想帧速率，您应测试不同的帧速率以弄清是否调整它们会显著影响您的电影可获得的帧速率。

一个示例将帮助阐明这一点：在基于 Symbian 系列 60v2 的电话上，定时器分辨率为 1/64 秒。这意味着 DoPlay 可以在下列时间间隔处得到调用：1/64、2/64、3/64、4/64、5/64 等。因此，在这样的设备上可用的帧速率 (fps) 有：64、32、21.3、16、12.8 等。

因此，当播放以 20 fps 创作的内容时，理论上应该每 1/20 秒调用它一次。但是，Symbian 仅有 1/64 秒的分辨率，这样 1/20 落在 Symbian 可以提供的 3/64 和 4/64 时间间隔之间。播放器是保守的，它将仅在超过 1/20 时间间隔时才前进帧，即在过去 4/64 秒之后。这相当于 16 fps！存在 20% 的调整，换言之，在感受到的性能方面有 20% 的损失！

# 需要记住的事项

## 工作区大小和发布配置文件

创建新电影时，请确保正确设置您的文档。尽管 Flash 电影可以流畅地缩放，但是如果电影不是以其原生工作区大小运行的且必须在播放器中进行缩放，则会存在一个性能最佳点。确保在设置文档的工作区大小时要与您的目标设备的分辨率匹配。同时确保在“发布设置”下将您的 Flash Player 设置为正确版本的 Flash Lite，并在“设备中心”中选择适当的设备配置文件。

## 时间间隔和监听器

卸载电影剪辑时，如果任何 ActionScript 函数仍引用 SWF 数据，则将不会重新收集 SWF 数据内存。使用时间间隔和监听器就是这样两种情形，这种情形下除非首先清除 ActionScript，否则不会释放数据。请确保在使用 `unloadMovie` 或 `removeMovieClip` 删除内容之前，通过使用 `clearInterval` 清除任何活动的时间间隔并使用 `removeListener` 删除任何活动的监听器。

## 跳帧

当使用 `gotoAndPlay` 时，请记住在播放请求的帧之前，需要初始化在当前帧和请求的帧之间的每一个帧。如果在这两个帧之间的每一个帧包含不同的内容，则使用不同的电影剪辑与使用时间线相比可能会更加有效。

## 全局和本地变量

在函数内，本地变量是以这样的方式登记的：这种方式下播放器获取和设置本地变量比获取和设置全局变量要快得多。可能的话，请使用 `var`。

## 匿名函数

请避免使用匿名语法定义函数：`myObj.eventName = function( ) { ... };`

显式定义的函数更有效：`function myFunc( ) { ... }; my Obj.eventName = myFunc;`

## 缩放和旋转位图

Flash Lite 播放器不支持位图平滑。这意味着如果您缩放或旋转位图，则外观将变得又粗又短。最好以位图的原生旋转和分辨率使用位图。如果您有一个需要缩放或旋转的图形，请考虑使用矢量图形。

## 渐变色带

当前在市场上的大多数设备仍仅支持 16 位色（上千种颜色），而不支持 24 或 32 位色（上百万种颜色）。这意味着渐变通常将显示为实色带状条纹，而不是平滑的渐变过渡。解决此问题的一种方法是通过称为 5\_6\_5 的电报使用一种免费的第三方 Photoshop 滤镜来对您的位图进行后处理，这样可以将位图的色深降至 16 位并抖动它。可以在此处找到该滤镜：<http://www.telegraphics.com.au/sw/>。

# 优化内容

## 艺术作品格式

可能的话，为艺术作品使用位图而非矢量。多个矢量的渲染会降低性能，而渲染位图要快得多。

## 压缩的 JPG

解压缩 JPG 将降低性能。因此，如果内存允许，请尝试使用 PNG。

## 透明的 PNG

请确保最大限度地减少 PNG 文件中透明度的数量 – 甚至对于位图的透明部分，该播放器必须计算重绘。例如，如果您有一个代表前景元素的透明 PNG，请不要以全屏大小导出透明的 PNG。请以前景元素的实际大小导出它。

## 渐变

最大限度地减少矢量渐变的使用。它们对于播放器计算和渲染来说太昂贵。如果您必须使用它们，请注意线性渐变渲染得比圆形渐变快。

## 矢量复杂性

如果绝对必须使用矢量形状，请尽可能优化它们。形状的维度越多，Flash 渲染该形状所必须做的事情就越多。优化对于小矢量形状（如图标）会尤其有帮助。您的图标可能很小以至于许多细节都丢失了，而如果要保留形状的复杂性，则优化对于播放器进行渲染是额外的工作。大多数情况下，使用位图来取代矢量是一个更好的解决办法。

## 矢量角和曲线

从数学上讲，角渲染要比曲线渲染简单。可能的话，请坚持使用直边，特别是对于非常小的矢量形状。

## 形状轮廓

填充仅有一个的外部形状要渲染，而轮廓有一个内部和一个外部要渲染。这意味着画一条线与填充相比要做双倍的工作。可能的话，请避免画线。

## 嵌入的文本

文本本质上仅是一个非常复杂的矢量形状。这使得字体类型成为 Flash 要渲染的更加复杂的形状之一。当然，文本通常是必要的，因此完全要避免它极难。如果您不使用文本，请避免对它进行动画处理或将它置于某个动画上，并考虑将文本作为位图使用。

## 多线动态文本

在播放器中换行是一个非常耗时的过程。静态文本字段并不是问题，因为换行是在编译时预先计算的。但对于多行动态和输入文本，字符串的换行不会进行缓冲处理。换行是在运行时在播放器中完成的，且每次需要重绘文本字段时都要重新计算换行。对于动态文本，使用动态文本字段是不可避免的，但可能的话，请考虑使用静态文本字段。

## 隐藏电影剪辑

请避免使用 `_alpha = 0` 和 `_visible = false` 来隐藏屏幕上的电影剪辑。如果您只是关闭电影剪辑的可见性或将其 `alpha` 更改为零，则它仍包含在播放器的扫描线渲染计算中，这些计算会影响性能。同样，也不要尝试通过将电影剪辑遮掩到另一个艺术作品的后面来隐藏电影剪辑。它将仍包含在播放器的计算中。请将电影剪辑完全从工作区中移出或使用 `removeMovieClip` 删除它们。

## 变形处理区域

当 Flash 绘制一个动画区域时，它是通过围绕该区域定义一个矩形包围盒来实现的。您可以通过使该矩形尽可能小来优化此绘图。这意味着，如果您可以，应避免重叠变形，因为 Flash 会将合并的区域视为单一的矩形，从而产生一个更大的总区域。请在播放器内使用 Flash 8 的“显示重绘区域”功能来优化您的动画。

## 图层类型

可能的话，尝试将位图图层和矢量图层排列得彼此靠近一些。随着播放器渲染电影，它需要根据内容的类型实施不同的渲染器。在渲染器之间的切换会花费时间，不是很多，但如果发生的次数很多，它会累计。通过使其上有矢量内容的图层排列得相互靠近以及使其上有位图内容的图层排列得相互靠近，播放器可以通过更少的切换次数更快地渲染它们。

## 清除源文件

这应该不成问题，但确定您将要保持您的电影尽可能小并在编译之前删除任何无关的内容和代码。确保删除未使用过的电影剪辑，去除不必要的帧和代码循环，并避免太多的帧或无关的帧。不管您信与不信，那些空帧真的会累计！

## 首帧初始化

尽管通过将您的所有内容置于电影的开始处来预加载它们在台式机上是行得通的，但在移动设备上这样做会导致电影开始时很慢。在整个电影中将内容自然地隔开，这样会在使用电影剪辑时才初始化它们。

## 数学和浮点数

最大限度地减少数学函数和浮点数的使用。计算这些函数会降低内容的性能。

## 数学例程数据

如果您必须使用数学例程，请考虑预先计算这些值并将它们存储在一个变量数组中。从数据表中抽取值要比让播放器在运行时计算它们快得多。

## 循环迭代

运行循环会由于每个迭代都要检查条件所招致的开销而变得很昂贵。如果迭代的成本和循环开销可比较时，请展开循环来分别执行多个操作。这会导致代码大小变大，但性能更快。

## XML 数据

可能的话，请避免加载和解析 XML 文件；这样会消耗处理器并影响性能。可能的话，请将数据存储在简单的名称/值对中，并通过使用 loadVars 从文本文件中加载它们，或从预编译的 SWF 文件加载它们。

## 关于作者

Josh Ulm 是 Adobe Systems, Inc. 的移动和设备部门的首席设计师和设计主管。自从 2004 年加入 Macromedia (后来 Macromedia 并入 Adobe) 之后, 他一直引领着移动和设备部门来定义移动体验平台, 并直接与开发人员和客户一起创建引人入胜的 Flash Lite 体验。他的工作为合并的公司及其客户推动了许多产品和技术的成功采用; 他经常被邀请去展示和开发公司的体验构想; 而且他是 Flash 和移动开发人员社区的一名积极的受尊敬的老手。