第 5 章　Flash CS3 高级动画

5.1　本章目的及任务

本章目的

- 了解高级动画的制作技巧
- 熟练掌握场景和图层的运用
- 熟练掌握引导动画和遮罩动画的运用
- 熟练掌握按钮的运用
- 熟练掌握 ActionScript 在动画中的运用

本章任务

- 任务一　场景
- 任务二　图层
- 任务三　引导动画
- 任务四　遮罩动画
- 任务五　交互式动画——按钮
- 任务六　ActionScript 动画

5.2　任务一　场景

5.2.1　相关知识

场景就是舞台，在动画影片中，所有的角色、背景都需要一个舞台来展现，这个舞台就是场景。动画中，所有的动画元素都被放置在场景中。

场景面板：大部分场景操作都是通过场景面板来实现的。打开场景面板可以通过执行“窗口”→“其他面板”→“场景”菜单命令来实现，如图 5-1 所示。

添加场景：一部话剧通常会有多个“幕”，一部动画通常也会有多个场景，添加场景可以通过执行“插入”→“场景”菜单命令来实现，如图 5-2 所示。

删除场景：如果需要删除一个场景，可以通过在场景面板中点击删除按钮来实现，如图 5-3 所示。

编辑场景：需要编辑某一个场景，只需在场景面板中选择相应的场景就能实现，如图 5-4 所示。

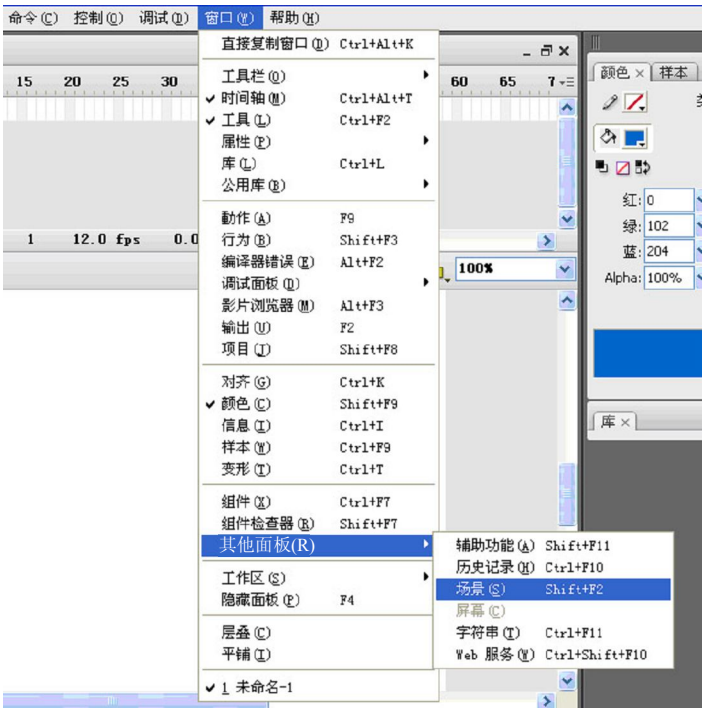
命令(C) 控制(O) 调试(D) 窗口(W) 帮助(H)
直接复制窗口(D) Ctrl+Alt+K
工具栏(O)
时间轴(M) Ctrl+Alt+T
工具(L) Ctrl+F2
属性(P)
库(L) Ctrl+L
公用库(B)
动作(A) F9
行为(B) Shift+F3
编译器错误(E) Alt+F2
调试面板(D)
影片浏览器(M) Alt+F3
输出(U) F2
项目(J) Shift+F8
对齐(G) Ctrl+K
颜色(C) Shift+F9
信息(I) Ctrl+I
样本(W) Ctrl+F9
变形(T) Ctrl+T
组件(X) Ctrl+F7
组件检查器(R) Shift+F7
其他面板(R)
工作区(S)
隐藏面板(P) F4
层叠(C)
平铺(I)
1 未命名-1
辅助功能(A) Shift+F11
历史记录(H) Ctrl+F10
场景(S) Shift+F2
屏幕(C)
字符串(T) Ctrl+F11
Web 服务(W) Ctrl+Shift+F10
颜色
样本
红: 0
绿: 102
蓝: 204
Alpha: 100%
库
100%
12.0 fps

图 5-1

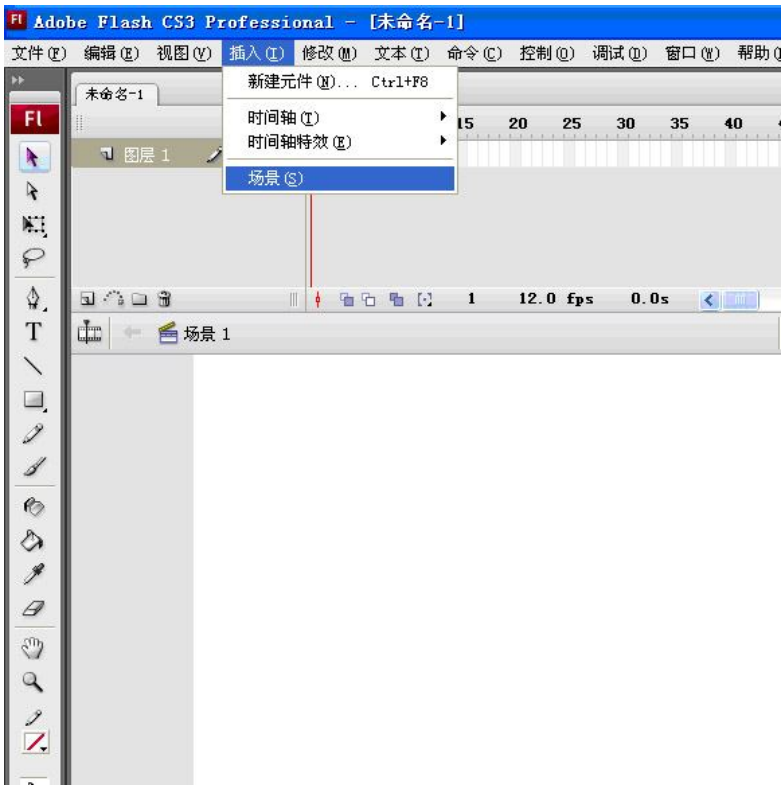
Adobe Flash CS3 Professional - [未命名-1]
文件(F) 编辑(E) 视图(V) 插入(I) 修改(M) 文本(T) 命令(C) 控制(O) 调试(D) 窗口(W) 帮助
新建元件(N)... Ctrl+F8
时间轴(T)
时间轴特效(E)
场景(S)
未命名-1
图层 1
12.0 fps
0.0s
场景 1

图 5-2

图 5-3

图 5-4

5.2.2　任务实现

制作有多个场景的简单动画

【设计思路】

❖　在“场景 1”中制作倒计时片头，在“场景 2”中制作一个简单动画。

【操作步骤】

（1）新建一个文件，将文档属性中的背景色改为黄色，“图层 1”命名为“背景”。

（2）使用线条工具和椭圆工具绘制如图 5-5 所示的图形。

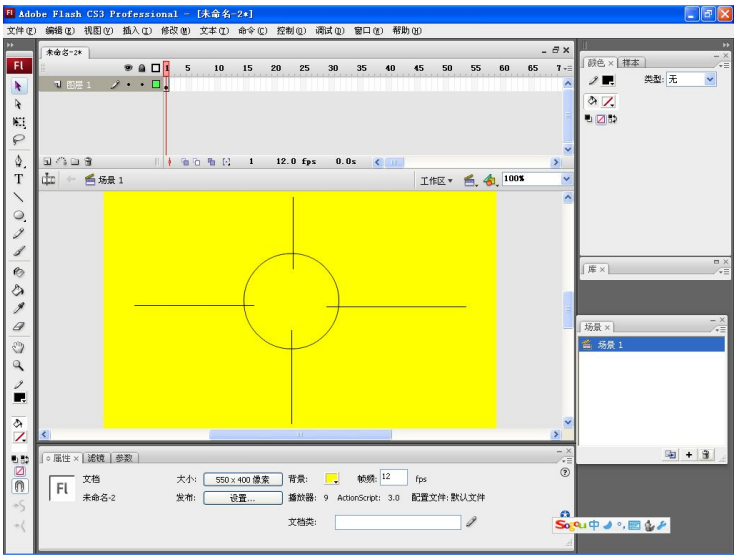

图 5-5

（3）插入新图层，命名为“数字”。

（4）使用文本工具在“数字”图层的第 1 帧上写上一个数字 10，如图 5-6 所示。

（5）在“数字”图层的第 5 帧处按 F6 键插入一个关键帧，使用文本工具将数字 10 改为 9，如图 5-7 所示。

（6）用同样的方法分别在“数字”图层的第 10、15、20、25、30、35、40、45 帧上插入关键帧、并将场景中的数字依次改为 8、7、6、5、4、3、2、1。如图 5-8 所示。

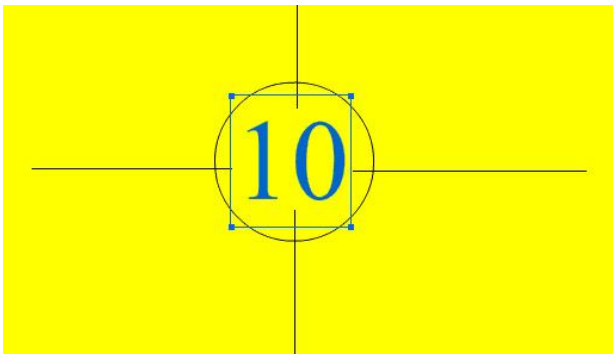

图 5-6

图 5-7

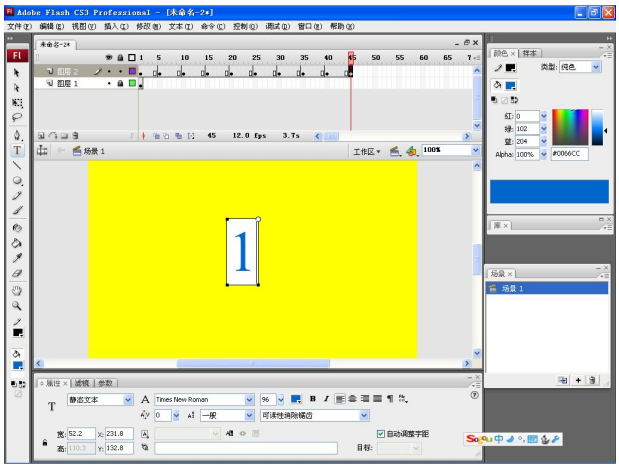

图 5-8

（7）在“背景”图层的时间轴上找到第 45 帧，按 F5 键插入一个普通帧，以显示背景，如图 5-9 所示。

图 5-9

（8）执行“插入”→“场景”菜单命令，新建“场景 2”，如图 5-10 所示。

图 5-10

（9）在“场景 2”中制作一个简单文字动画，使用文本工具输入“FlashCS3 高级动画”，如图 5-11 所示。

图 5-11

（10）在文字上右击鼠标，执行“时间轴特效”→“效果”→“分离”菜单命令，弹出分离窗口后点击“确定”按钮，如图 5-12 所示。

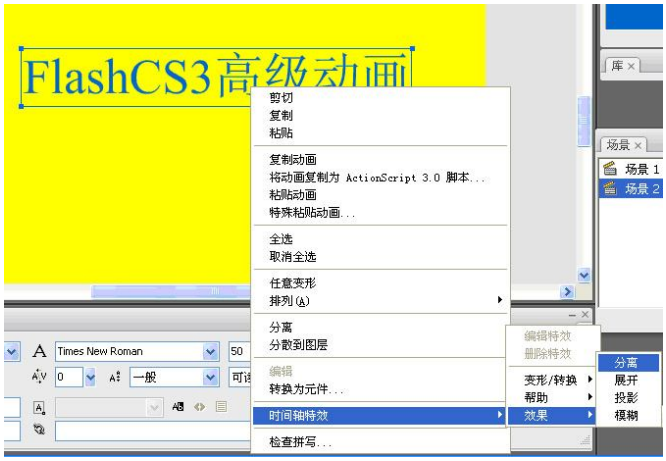

图 5-12

（11）保存文件并预览效果，如图 5-13 所示。

图 5-13

5.3 任务二 图层

5.3.1 相关知识

图层在很多软件中，特别是图形、图像、动画、影片制作类软件中都有使用。图层就是将画面中的不同元素分隔开，以便于对不同部分进行独立操作。

编辑图层：时间轴的左侧就是“图层”面板，可以通过面板中的 4 个按钮“插入图层”、“添加运动引导层”、“插入图层文件夹”、“删除图层”来对图层进行编辑操作，如图 5-14 所示。

图 5-14

图层的状态：图层有四种不同的状态，分别为“编辑状态”、“显示或隐藏状态”、“锁定或解锁状态”、“显示轮廓或颜色状态”，可以通过点击图层名称右侧的按钮来选择不同状态，如图 5-15 所示。

图 5-15

图层属性：在图层名称上右击鼠标，选择“属性”，可以打开“图层属性”对话框对图层的属性进行修改，如图 5-16 所示。

图 5-16

图层命名：当文档中图层过多时，给图层起一个容易记忆的好名字就显得很必要。给图层命名可用鼠标双击图层面板中的图层，然后输入新的名称即可。

5.3.2　任务实现

制作多图层波纹效果

【设计思路】

❖　通过多个图层上的圆形放大来实现波纹的效果。

【操作步骤】

（1）新建一个文件。

（2）执行“插入”→“新建元件”菜单命令，新建一个名为“圆”的图形元件，如图 5-17 所示。

图 5-17

（3）在图形元件中绘制一个椭圆，不要填充色并调整至中央位置，如图 5-18 所示。

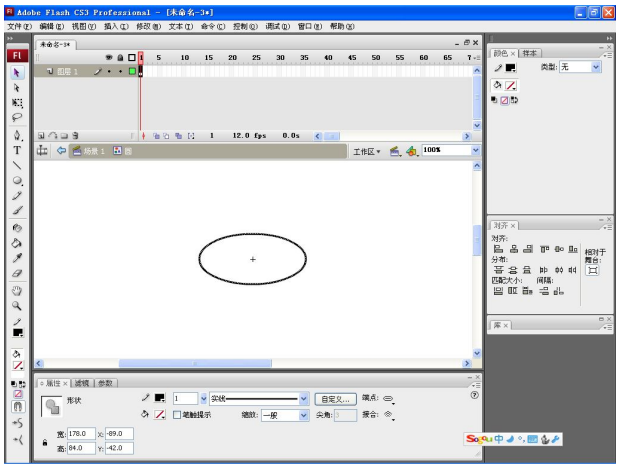

图 5-18

（4）返回“场景 1”，将图形元件“圆”从库中拖至“图层 1”的中间位置。为了能够准确定位，可以执行“视图”→“网格”→“显示网格”菜单命令，如图 5-19 所示。

（5）在第 10 帧上按 F6 键插入关键帧，使用变形工具将图形元件放大。在第 1 帧和第 10 帧之间右击鼠标创建补间动画，如图 5-20 所示。

（6）新建图层 2，在“图层 2”的第 2 帧插入关键帧，从库中将图形元件“圆”拖至中间位置，如图 5-21 所示。

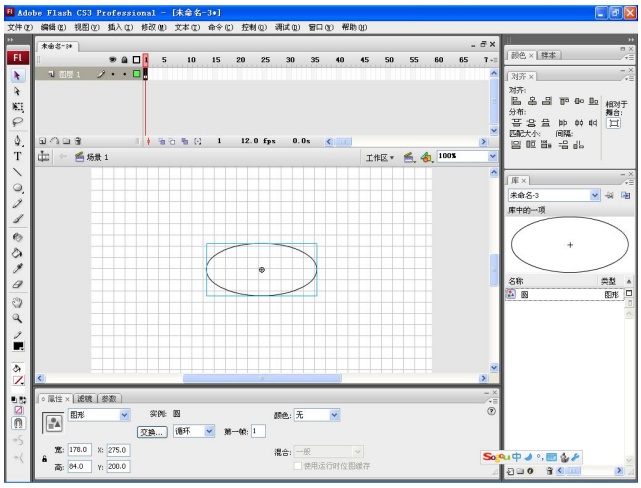

图 5-19

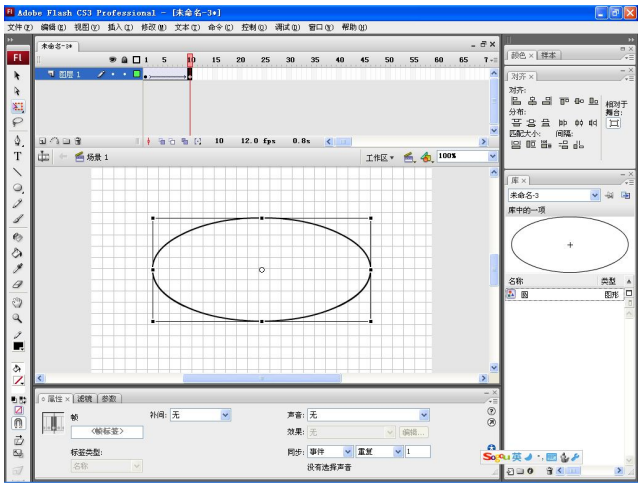

图 5-20

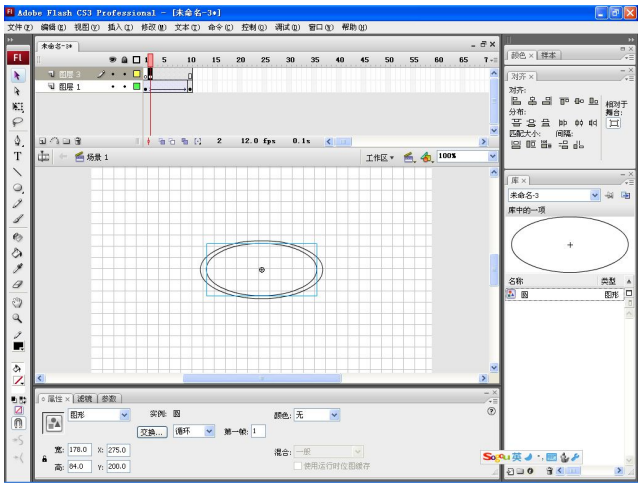

图 5-21

（7）在“图层 2”的第 11 帧插入关键帧，将图形元件放大。在第 2 帧和第 11 帧之间创建补间动画，如图 5-22 所示。

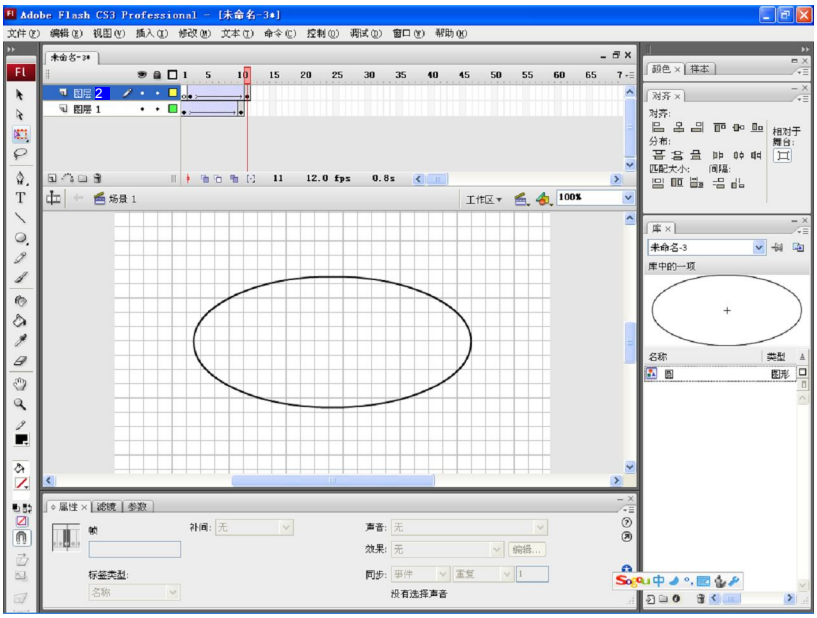

图 5-22

（8）用同样方法新建图层 3、图层 4、图层 5，制作波纹效果，如图 5-23 所示。

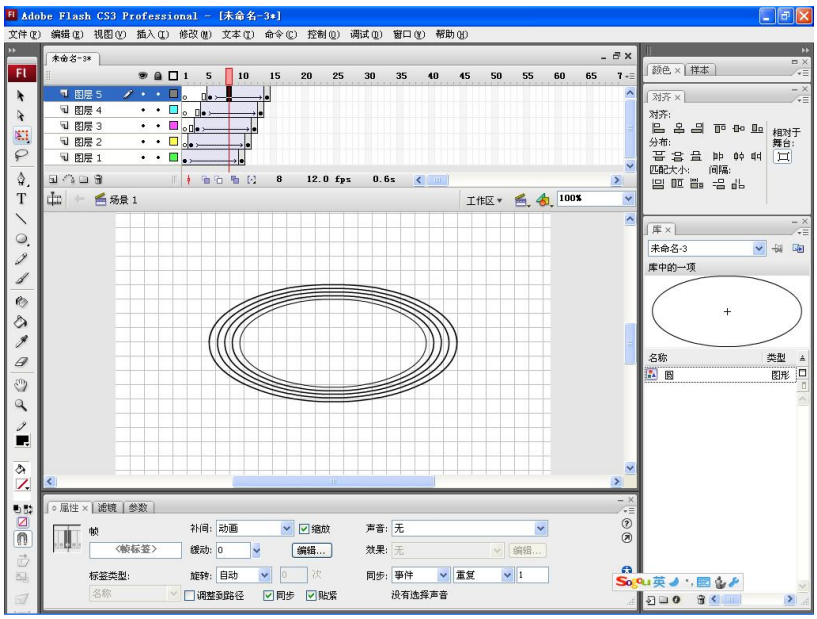

图 5-23

（9）保存文件并预览效果，如图 5-24 所示。

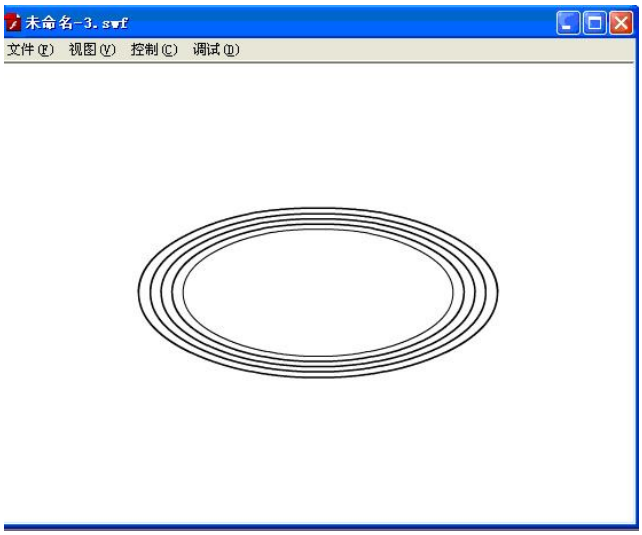

图 5-24

5.4 任务三 引导动画

5.4.1 相关知识

动画中的运动经常是不规则的曲线运动，引导动画就是要让动画元素沿着一条特定的路线运动。

将一个或多个层链接到一个运动引导层，使一个或多个对象沿同一条路径运动的动画形式被称为“引导路径动画”。这种动画可以使一个或多个元件完成曲线或不规则运动。

一个最基本“引导路径动画”由两个图层组成，上面一层是“引导层”，它的图层图标为，下面一层是“被引导层”，图标同普通图层一样。

创建引导路径动画的方法是在普通图层上点击时间轴面板的“添加引导层”按钮，该层的上面就会添加一个引导层，同时该普通层缩进成为“被引导层”。

5.4.2 任务实现

操作一 制作蝴蝶飞舞的动画

【设计思路】

❖ 首先制作一个蝴蝶元件，之后使用引导动画使蝴蝶元件沿曲线移动。

【操作步骤】

（1）新建一个文件，将“图层 1”命名为“蝴蝶”。

（2）执行“文件”→“导入”→“导入到舞台”菜单命令，打开“实例素材\第 5 章\.蝴蝶.jpg”导入一张图片，如图 5-25 所示。

（3）按 Ctrl+B 组合键将图片分离，使用套索工具中的魔棒将蝴蝶周围白色部分去掉，并将蝴蝶图形转换成图形元件，如图 5-26 所示。

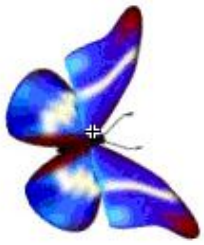

图 5-25

图 5-26

（4）在时间轴上找到第 30 帧，按 F6 键插入关键帧。在第 1 帧和第 30 帧之间创建补间动画，如图 5-27 所示。

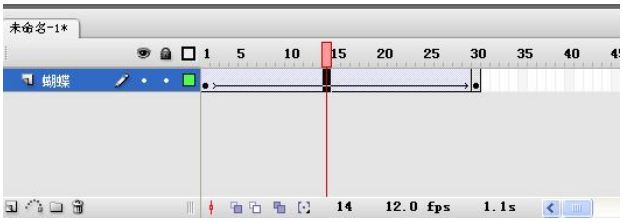

图 5-27

（5）在“图层”面板上点击“添加运动引导层”按钮创建引导层，使用铅笔工具在引导层上画一条曲线，如图 5-28 所示。

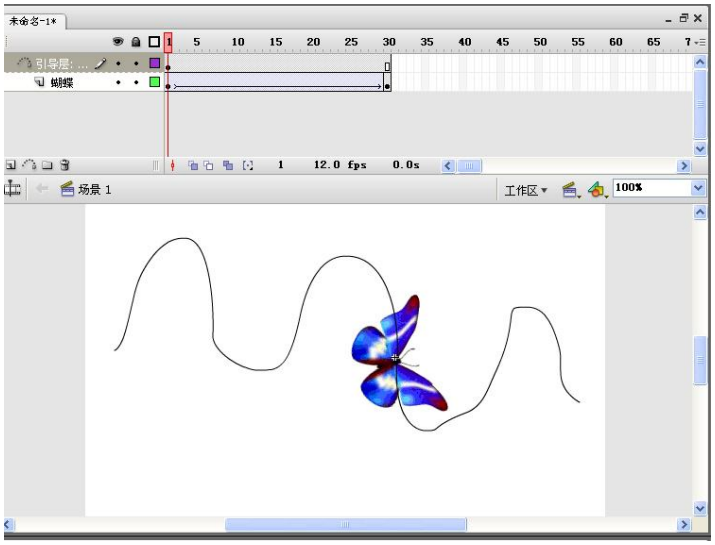

图 5-28

（6）点击“图层 1”的第 1 帧，将蝴蝶放在曲线的左侧端点上。点击“图层 1”的第 30 帧，将蝴蝶放在曲线的右侧端点上，如图 5-29 所示。

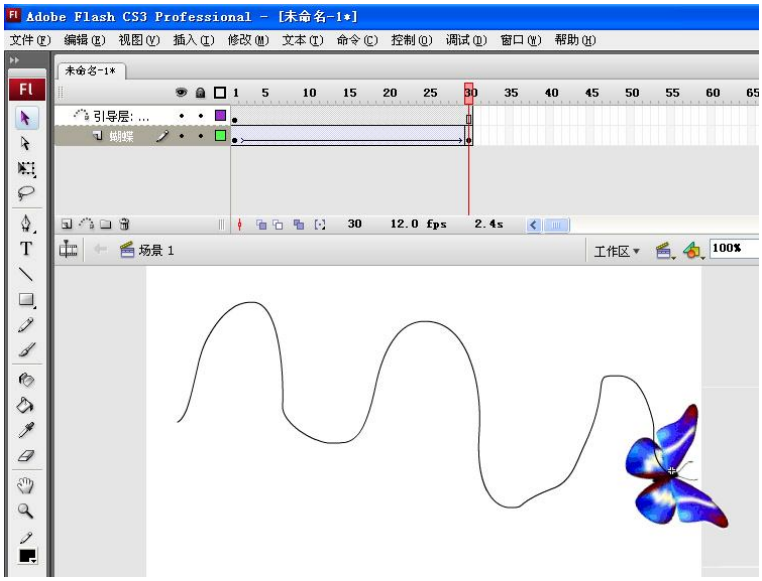

图 5-29

（7）保存文件并预览效果。

操作二　制作太阳系旋转的动画

【设计思路】

- ❖ 该动画包括两个部分，月球围绕地球旋转，地球围绕太阳旋转，同时太阳、地球、月球都在自转。
- ❖ 制作时需要注意的是，当引导线是一个闭合的圆时，不能完成引导动画。必须将闭合的线擦出一个缺口。

【操作步骤】

（1）新建一个文件。

（2）执行“插入”→“新建元件”菜单命令，新建一个图形元件，名为“星球”。在元件中绘制一个有渐变色的圆球，如图 5-30 所示。

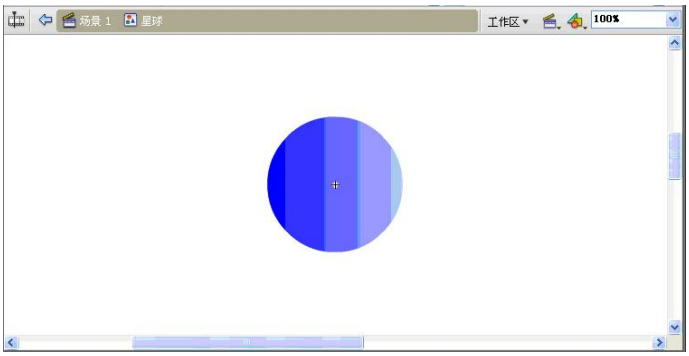

图 5-30

（3）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，命名为“自转”，如图 5-31 所示。

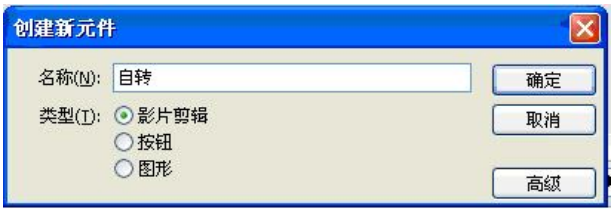

图 5-31

（4）编辑影片剪辑元件“自转”，从库中找出一个图形元件“星球”放在舞台中央，在时间轴上找到第 15 帧，插入关键帧。在第 1～15 帧之间创建补间动画，在属性中找到旋转，选择“逆时针”旋转，如图 5-32 所示。

图 5-32

（5）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，名为“地月系”。将“图层 1”命名为“地球图层”，从库中找出一个元件“自转”，放在舞台中间并调整大小，如图 5-33 所示。

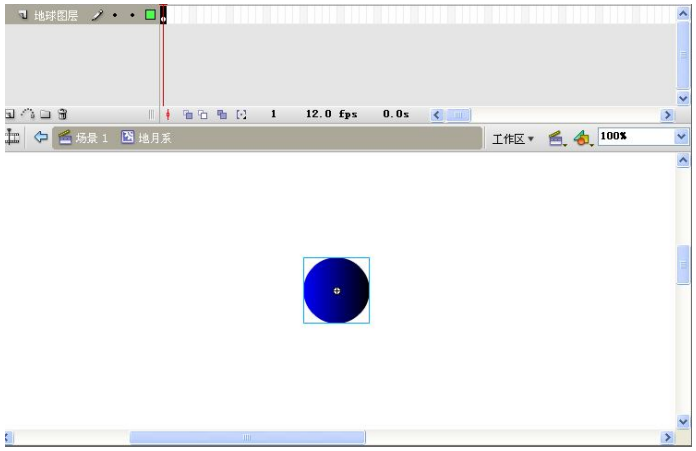

图 5-33

（6）在“地球图层”的时间轴上找到第 50 帧，按 F5 键插入普通帧。

（7）在影片剪辑“地月系”中新建图层并命名为“月球图层”，从库中再次找出元件“自转”放在“月球图层”的第 1 帧上，调整大小，如图 5-34 所示。

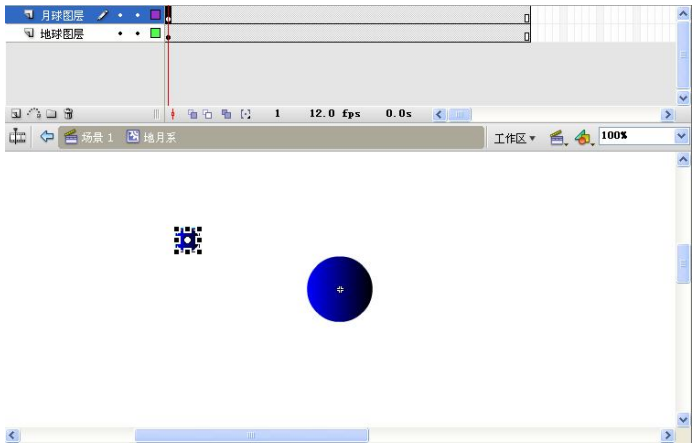

图 5-34

（8）找到“月球图层”的第 50 帧，插入一个关键帧。在第 1～50 帧之间创建补间动画，如图 5-35 所示。

图 5-35

（9）点击“图层”面板中的“添加运动引导层”按钮，为“月球图层”创建引导层。使用椭圆工具画一圆形线条，并使用橡皮擦工具将圆形擦出一个小缺口，如图 5-36 所示。

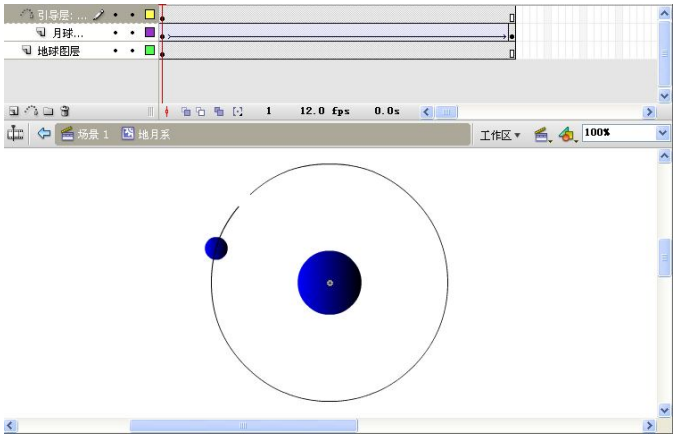

图 5-36

（10）选择“月球图层”的第 1 帧，将小圆形放在线条的一个端点上。再选择“月球图层”的第 50 帧，将小圆形放在线条的另一个端点上，如图 5-37 所示。

（11）返回“场景 1”新建图层，命令为“太阳图层”，从库中找出一个元件“自转”，放在舞台中央，并在“太阳图层”的第 100 帧处插入普通帧，如图 5-38 所示。

（12）在“场景 1”中新建图层，命名为“地月图层”，从库中将元件“地月系”放在“地

月图层”的第 1 帧上并调整其大小。

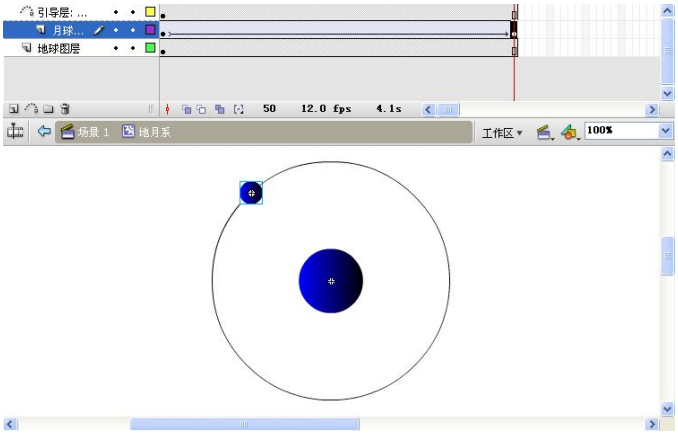

图 5-37

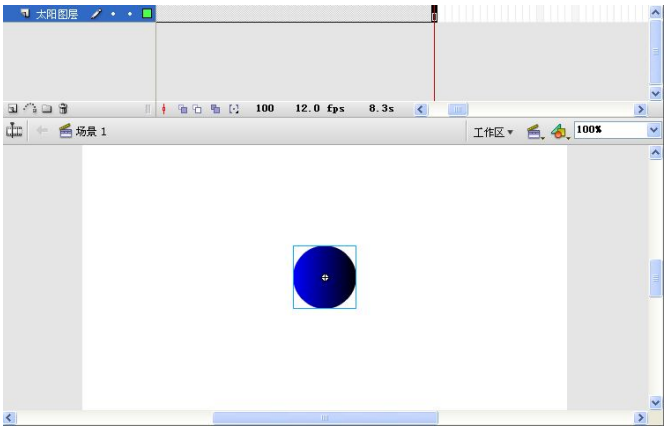

图 5-38

（13）在“地月图层”的第 100 帧插入一个关键帧。在第 1～100 帧之间创建补间动画，如图 5-39 所示。

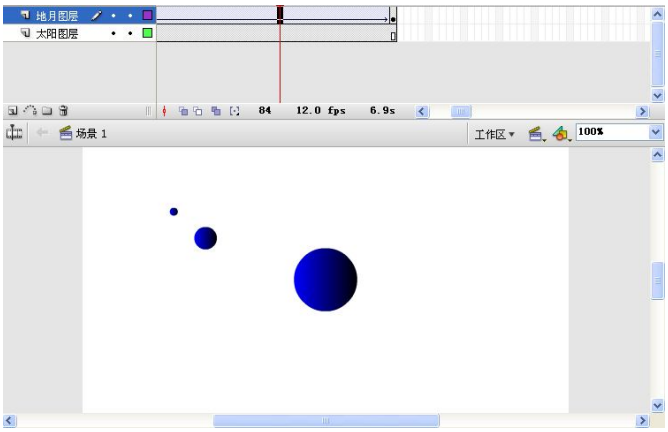

图 5-39

（14）点击“图层”面板中的“添加运动引导层”按钮，为“地月图层”创建引导层。使用椭圆工具绘制一个圆形，并使用橡皮擦工具将圆形擦出一个小缺口。如图 5-40 所示。

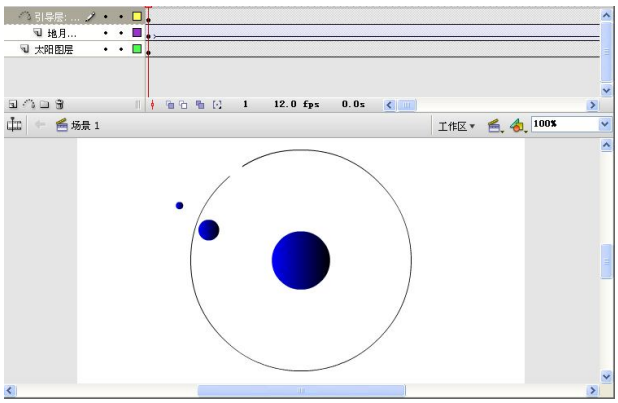

图 5-40

（15）选择“地月图层”的第 1 帧，将“地月系”放在线条的一个端点上。再选择“地月图层”的第 100 帧，将”地月系”放在线条的另一个端点上，如图 5-41 所示。

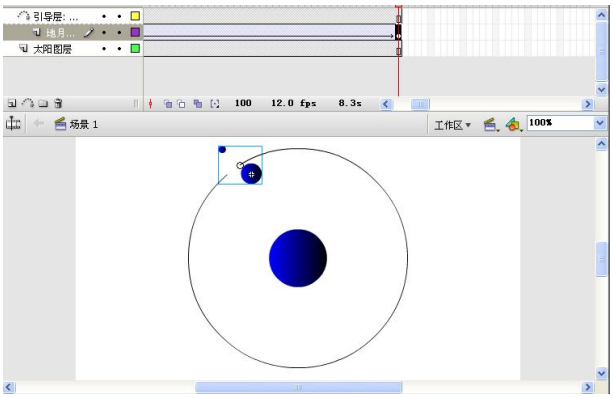

图 5-41

（16）保存文件并预览效果。

操作三　制作光绕文字运动的动画

【设计思路】

❖ 想要让星光围绕文字运动，就要有文字形状的引导线，这条引导线可以使用墨水瓶工具来实现。

【操作步骤】

（1）新建一个文件，背景色为黑色。

（2）执行“插入”→“新建元件”菜单命令，新建一个图形元件，名为“星星”，画出如图 5-42 所示的图形。

（3）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，名为“星光”。将“星星”元件放在舞台中间，在第 5 帧上插入关键帧，并缩小“星星”。在第 10 帧上插入关键帧，并放大“星星”，如图 5-43 所示。

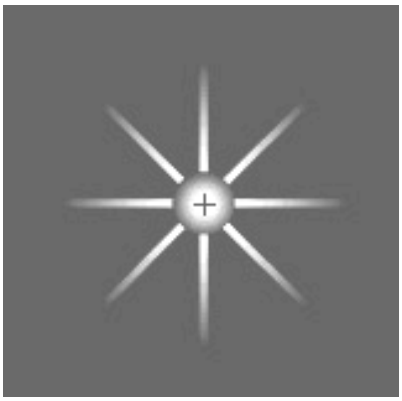

图 5-42

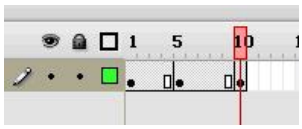

图 5-43

（4）返回“场景 1”，用文本工具在“图层 1”上写一个字母，如图 5-44 所示。

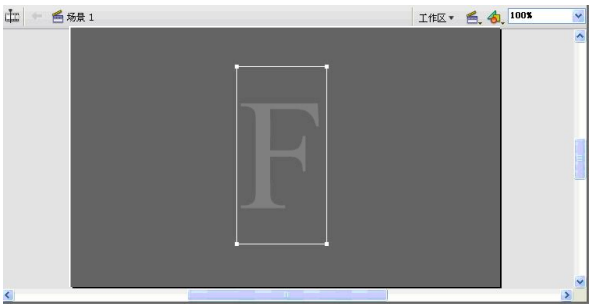

图 5-44

（5）将字母分离，在时间轴上找到第 30 帧，插入普通帧，如图 5-45 所示。

图 5-45

（6）新建图层 2，命名为“星光”，将“星光”元件放在第 1 帧上，在第 30 帧上插入关键帧，并在第 1 帧和第 30 帧之间创建补间动画，如图 5-46 所示。

（7）点击“添加运动引导层”按钮，为“星光”层添加引导层。复制“图层 1”中的字母，将字母粘贴到引导层中相同位置上。

（8）使用墨水瓶工具点击引导层中的字母，这时字母周围出现红色轮廓线。删除字母颜色，并使用橡皮擦工具将红色轮廓线擦出一个小缺口，如图 5-47 所示。

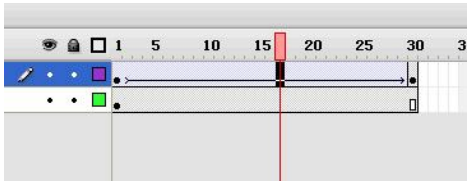

图 5-46

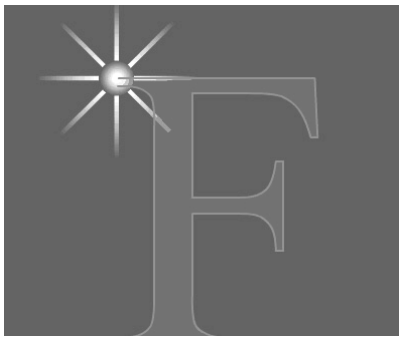

图 5-47

（9）在“星光”层的第 1 帧，将“星光”放在红线的一个端点上。在“星光”层的第 30 帧，将“星光”放在红线的另一个端点上，如图 5-48 所示。

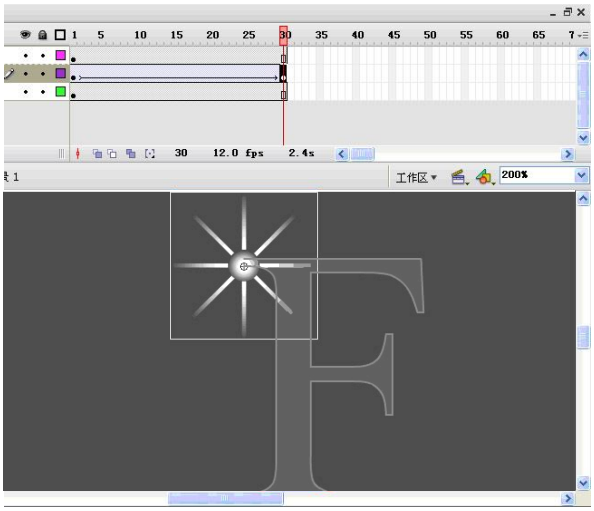

图 5-48

（10）保存文件并预览效果。

5.5 任务四 遮罩动画

5.5.1 相关知识

遮罩层是一种特殊的图层，之所以说它特殊，是因为遮罩层上有图形的部分是透明的，而没有图形的部分是不透明的。这就可以让我们使用遮罩层来显示一些需要看到的画面部分，而隐藏一些不需要看到的画面部分。

在 Flash CS3 中，遮罩动画也确实是通过遮罩层来达到有选择地显示位于其下方被遮罩层中的内容的目的，在一个遮罩动画中，遮罩层只有一个，被遮罩层可以有任意个。

遮罩主要有两种用途，一个是用在整个场景或一个特定区域，使场景外的对象或特定区域外的对象不可见，另一个是用来遮罩住某一元件的一部分，从而实现一些特殊的效果。

在 Flash 中没有一个专门的按钮来创建遮罩层，遮罩层其实是由普通图层转化的。只要在某个图层上右击，在弹出菜单中勾选“遮罩”，该图层就会生成遮罩层，层图标就会从普通层图标变为遮罩层图标，系统会自动把遮罩层下面的一层关联为被遮罩层，在缩进的同时图标变为，如果想关联更多层被遮罩，只要把这些层拖到被遮罩层下面就行了。

遮罩层中的图形对象在播放时是看不到的，遮罩层中的内容可以是按钮、影片剪辑、图形、位图、文字等，但不能使用线条，如果一定要用线条，可以将线条转化为“填充”。

被遮罩层中的对象只能透过遮罩层中的对象被看到。在被遮罩层，可以使用按钮、影片剪辑、图形、位图、文字、线条。

可以在遮罩层、被遮罩层中分别或同时使用形状补间动画、动作补间动画、引导线动画等动画手段，从而使遮罩动画变成一个可以施展无限想象力的创作空间。

5.5.2　任务实现

操作一　制作探照灯效果

【设计思路】

❖　使用遮罩层上左右移动的圆形来呈现背景上的文字。

【操作步骤】

（1）新建一个文件，将“图层 1”命名为“文字”。

（2）执行“插入”→“新建元件”菜单命令，新建一个图形元件，名为“圆”，绘制一个圆形，如图 5-49 所示。

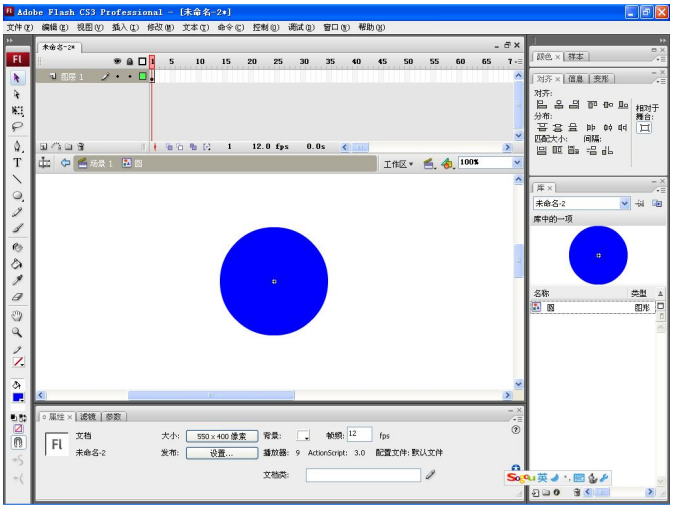

图 5-49

（3）返回“场景 1”，使用文本工具在“文字”层上输入文字“Flash CS3 高级动画”，在第 30 帧上插入普通帧。

（4）新建图层 2，命名为“遮罩”，将“圆”元件放在第 1 帧左侧的位置，并在第 15、30 帧分别插入关键帧，如图 5-50 所示。

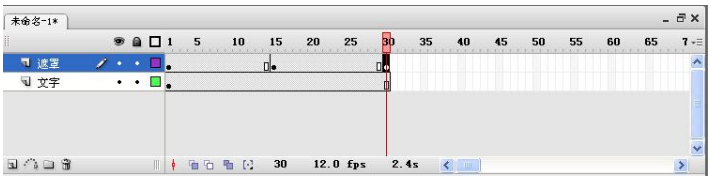

图 5-50

（5）选择第 15 帧，将“圆”元件拖至文字右侧。在第 1 帧和第 15 帧、第 15 帧和第 30 帧之间分别创建补间动画。

（6）在“遮罩”层上右击鼠标选择“属性”，在弹出的“图层属性”对话框中设置类型为“遮罩层”，如图 5-51 所示。

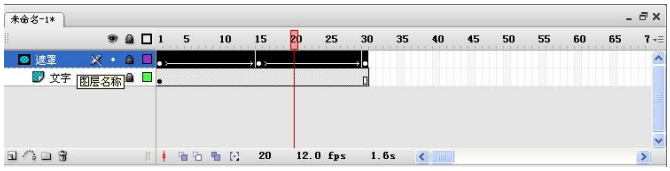

图 5-51

（7）保存文件并预览效果。

操作二　制作遮罩文字动画

【设计思路】

❖ 将两层相同的文字图层放在一起，上层比下层亮度高一些，增加遮罩层后，就会有明暗不同的部分变化效果，仿佛一束光照过文字。

【操作步骤】

（1）新建一个文件，将“图层 1”命名为“文字一”。

（2）执行“插入”→“新建元件”菜单命令，新建一个图形元件，命名为“文字”。使用文本工具输入文字“Flash CS3 高级动画”，如图 5-52 所示。

图 5-52

（3）执行“插入”→“新建元件”菜单命令，新建一个图形元件，命名为“闪光”，并绘制一个矩形，如图 5-53 所示。

（4）返回“场景 1”。将“文字”元件放在“文字一”层的第 1 帧上，并在第 30 帧插入普通帧。

（5）新建图层 2，命名为“文字二”，将“文字”元件放在“文字二”层的第 1 帧上，并改变元件的亮度，如图 5-54 所示。

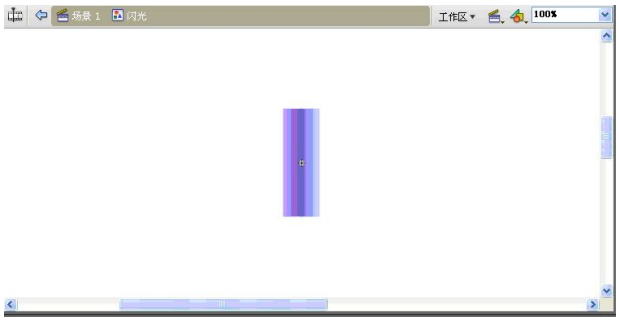

图 5-53

图 5-54

（6）新建图层 3，命名为“遮罩层”，将“闪光”元件放在“遮罩层”第 1 帧上的左侧位置，在第 15、30 帧上分别插入关键帧。

（7）选择第 15 帧，将“闪光”元件放在文字右侧位置。在第 1～15 帧、第 15～30 帧之间分别创建补间动画。

（8）在“遮罩层”上右击鼠标选择“属性”，在弹出的“图层属性”对话框中设置类型为“遮罩层”。

（9）保存文件并预览效果。

操作三　制作水波倒影动画

【设计思路】

❖　使用遮罩层技术制作波动效果。

【操作步骤】

（1）新建一个文件，将“图层 1”命名为“背景”。

（2）执行“文件”→“导入”→“导入到舞台”菜单命令，打开素材文件“实例素材\第 5 章\风景.jpg”，修改图片大小如图 5-55 所示，右击鼠标选择“转换为元件”，将图片转换为图形元件，在第 10 帧处插入普通帧。

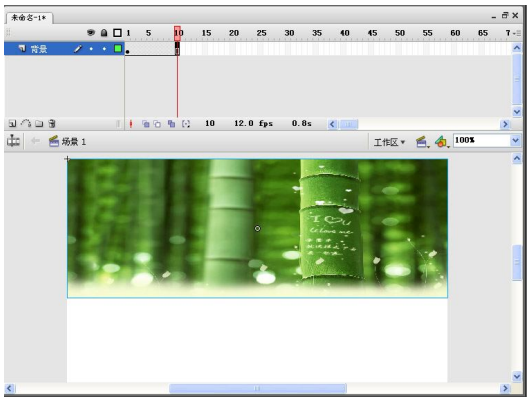

图 5-55

（3）在图片上右击鼠标，复制图片。新建图层 2，命名为“倒影一”，将复制的图片粘贴至“倒影一”图层，执行“修改”→“变形”→“垂直翻转”菜单命令，并移动位置如图 5-56 所示。

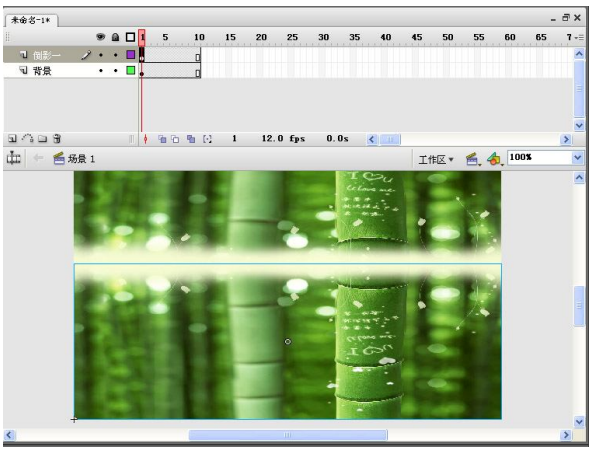

图 5-56

（4）选择倒影图片，在属性栏中修改“颜色”→“色调”为墨绿色，如图 5-57 所示。

图 5-57

（5）选择倒影图片，右击鼠标，复制图片。新建图层 3，命名为“倒影二”，将复制的图片粘贴到新图层中，并移动位置如图 5-58 所示。

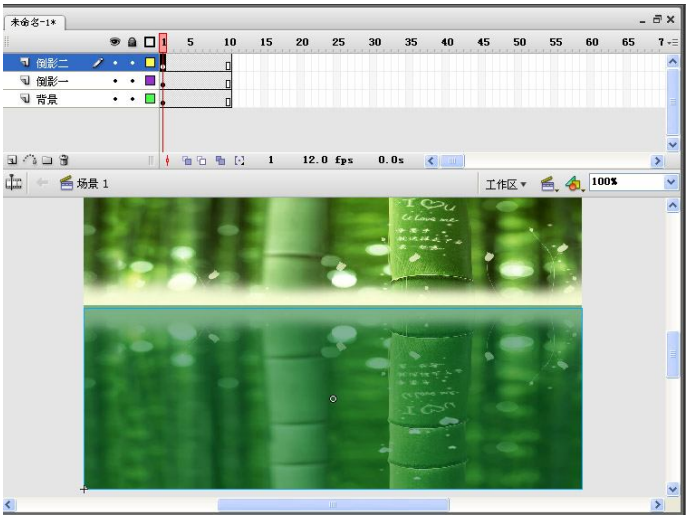

图 5-58

（6）新建图层 4，命名为“遮罩”，使用矩形工具绘制如图 5-59 所示的图形。

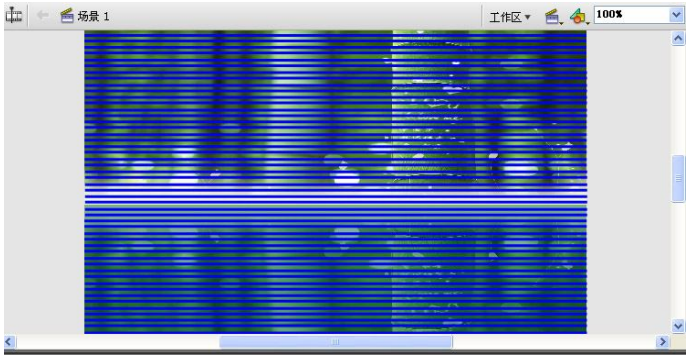

图 5-59

（7）右击鼠标，将矩形条转换成图形元件，在第 10 帧插入一个关键帧，将矩形条向下移动一点距离，并创建补间动画，如图 5-60 所示。

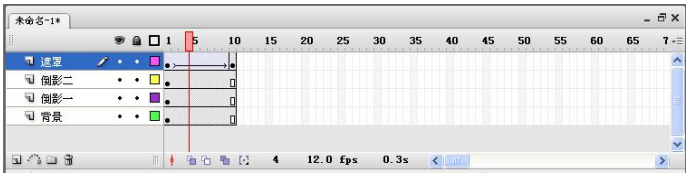

图 5-60

（8）在“遮罩”层上右击鼠标选择“属性”，在弹出的“图层属性”对话框中设置类型为“遮罩层”。

（9）保存文件并预览效果，如图 5-61 所示。

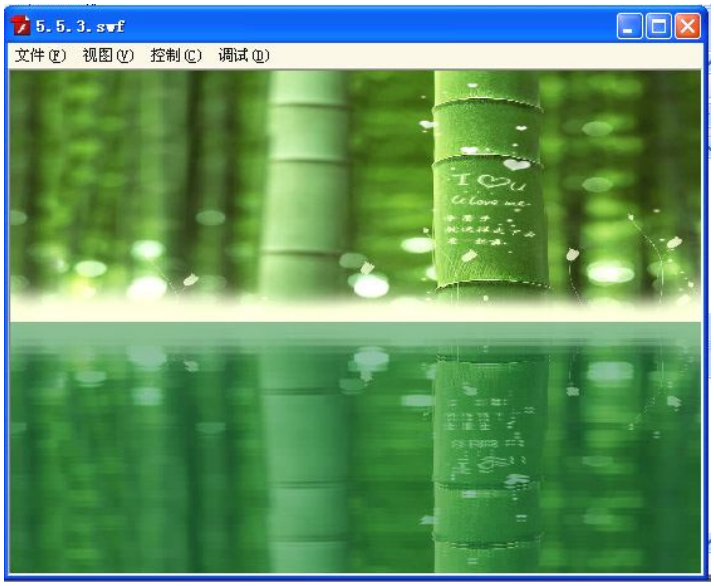

图 5-61

5.6 任务五 交互式动画——按钮

5.6.1 相关知识

Flash 元件有 3 种类型，即图形元件、按钮元件和影片剪辑元件。按钮元件可以给我们提供一个与动画交流的机会。按钮在制作中分为 4 个不同状态，弹起、指针经过、按下、点击。通过在这 4 个不同状态下制作不同的图形，制作出动态的按钮效果。

5.6.2 任务实现

操作一 制作风车动画

【设计思路】

❖ 制作两种风车效果，即静态的风车和转动的风车，并在不同按钮状态下放置不同的风车。

【操作步骤】

（1）新建一个文件。

（2）执行“插入”→“新建元件”菜单命令，新建一个图形元件，命名为“风车”。在元件中绘制如图 5-62 所示的图形。

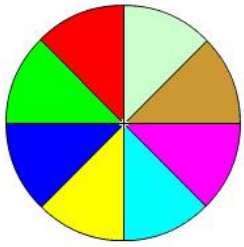

图 5-62

（3）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，命名为“风车转动”。在元件中将图形元件“风车”拖至第 1 帧，在第 10 帧上插入关键帧，并在第 1 帧和第 10 帧之间创建补间动画，设置旋转方式为“逆时针”，如图 5-63 所示。

图 5-63

（4）执行“插入”→“新建元件”菜单命令，新建一个按钮元件，命名为“风车按钮”。将图形元件“风车”放在“弹起”状态下，如图 5-64 所示。

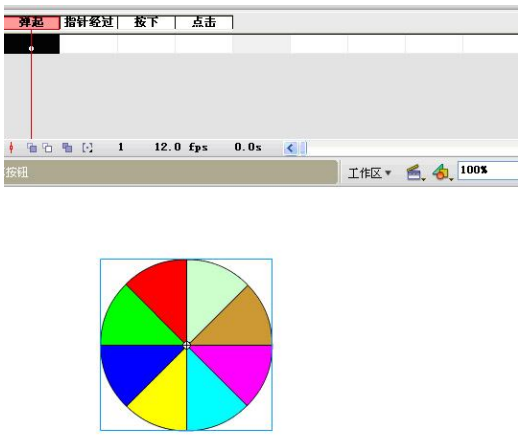

图 5-64

（5）在“指针经过”状态下插入关键帧，将“风车转动”元件放在舞台中间，如图 5-65 所示。

图 5-65

（6）返回“场景 1”。将按钮“风车按钮”拖至舞台中，如图 5-66 所示。

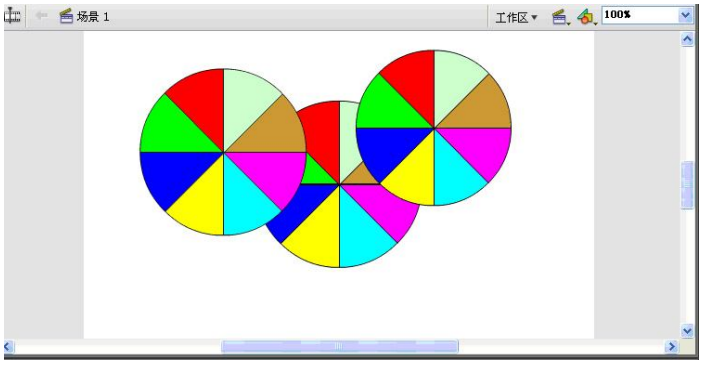

图 5-66

（7）保存文件并预览效果。

操作二　制作按钮控制影片播放

【设计思路】

❖　使用按钮元件控制影片的播放、停止，并使用语句 on(){};。

【操作步骤】

（1）新建一个文件。

（2）制作一个简单的变形动画。在第 1 帧上使用椭圆工具绘制一个圆形，在第 20 帧上插入一个关键帧，删除圆形，使用矩形工具绘制一个正方形，在第 1 帧和第 20 帧之间创建补间形状动画，如图 5-67 所示。

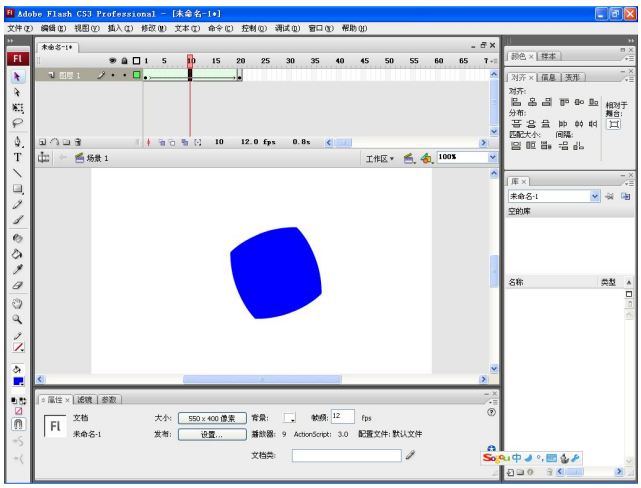

图 5-67

（3）新建图层 2，命名为“按钮”，执行“窗口”→“公用库”→“按钮”菜单命令，打开按钮公用库，找到 rounded green play 按钮，如图 5-68 所示。

图 5-68

（4）将 play 按钮放在“按钮”层中，选中按钮，执行“窗口”→“动作”菜单命令打开“动作”面板，输入如图 5-69 所示的代码。

（5）在库中找到 stop 按钮，将按钮放在“按钮”层中，打开“动作”面板，输入如下所示的代码：

```
On(release){
  Stop();
}
```

（6）保存文件并预览效果。

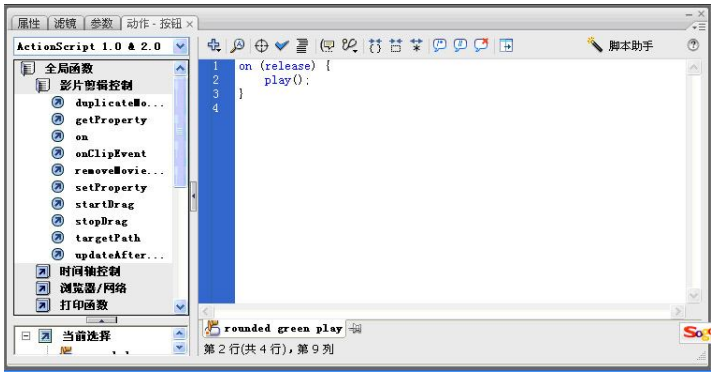

图 5-69

5.7　任务六　ActionScript 动画

5.7.1　相关知识

动作脚本英文为 ActionScript，简称 AS，是一种程序设计语言，脚本由一条条的命令组成，这些命令也叫代码，指示计算机完成相应的操作。动作脚本可以实现和用户的交互响应，实现较复杂、更逼真的动画效果。

脚本是由英语单词、数学符号和函数构成的。下面是一个 ActionScript 的例子：

```
on (press)
{ gotoAndPlay ("my frame"); }
```

可以通过其中的关键单词推测这段程序的作用。单词 press 表示按，即用鼠标在某种对象上面单击，这里的对象就是按钮。第二行中的长单词可以分开读成“go to and play”，可以将其理解为命令 Flash 到达影片中的一个特定位置并从这一特定位置开始播放影片。从以上可以看出，ActionScript 可以控制 Flash 影片的播放。

Flash 影片可以包含若干场景，每个场景都有时间轴，每条时间轴从第 1 帧开始。如果不添加 ActionScript，Flash 影片会自动从场景 1 的第 1 帧开始播放，直到场景 1 的最后一帧，然后接着播放场景 2，依此类推。ActionScript 的主要目的就是用来改变这种自动而死板的线性播放行为，一段脚本可以使影片在一个特定的帧上停止，循环播放前面的部分，甚至于让用户控制要播放哪一帧。ActionScript 能够使影片完全脱离被动的线性播放模式。这还不是 ActionScript 的所有功能，它还可以将 Flash 影片从简单的动画改变为具有交互能力的电脑程序。

1. 动作面板

可以通过执行“窗口”→“动作”菜单命令或按 F9 键打开“动作”面板。

2. 写代码的基本步骤

建立一个专门写代码的层，选定某个关键帧或元件实例，打开“动作”面板，在面板右侧的脚本窗格中写代码，一般每条语句写一行，以分号结束。

写代码时需要注意书写规范：ActionScript 中严格区分大小写字母，标点符号必须在英文

状态下输入，关键字一般呈现蓝色（关键字是具有特定含义的保留字，是用于执行一项特定操作的单词）。

3. 事件

事件是 SWF 文件播放时发生的动作。

（1）观察事件。

- 按钮。

Press：当按下按钮时触发。

Release：当释放按钮时触发。

ReleaseOutside：鼠标指针位于按钮内部按下按钮，然后将鼠标指针移到该按钮外部并释放鼠标时触发。

RollOut：当鼠标指针移至按钮区域之外时触发。

RollOver：当鼠标指针移过按钮区域时触发。

DragOut：当在按钮上单击鼠标，然后将鼠标指针拖动到按钮之外时触发。

DragOver：当用户在按钮外部按下鼠标，然后将鼠标指针拖动到按钮之上时触发。

- 影片剪辑。

EnterFrame：以 SWF 文件的帧频重复触发。

MouseDown：当按下鼠标时触发。

MouseMove：当鼠标移动时触发。

MouseUp：释放鼠标时触发。

Press：鼠标指针处于影片剪辑之上单击鼠标时触发。

Release：在影片剪辑上释放鼠标时触发。

ReleaseOutside：当影片剪辑区域中按下鼠标并且在影片剪辑区域之外释放它后触发。

RollOut：当鼠标指针移动到影片剪辑区域的外面时触发。

RollOver：当鼠标指针划过影片剪辑区域时触发。

- 帧事件。

在主时间轴或影片剪辑时间轴上，当播放头进入关键帧时会发生。

（2）ActionScript 中的事件处理。

- 事件处理函数。

事件处理函数由三部分组成：事件所应用的对象、对象的事件处理函数方法的名称和分配给事件处理函数的函数。

例如：新建一个文件文档，建立一个 10 帧的动画，放置一个按钮元件，命名为 stop_btn。在第 1 帧上写以下代码，将一个函数分配给按钮的 onPress 事件处理函数，该函数让动画停止。

```
Stop_btn.onPress=function(){
   Stop();
}
```

- 点语法。

所有的语言，无论是计算机语言还是人类的语言都要遵循一定的规则以便于理解。这些规则称为语法。Flash 使用点语法，意为脚本的各个部分通过句点“.”连接在一起。主要用途有两个：一是设置元件的属性和调用方法，二是指示元件的访问路径。

4. 影片剪辑属性

通过调整影片剪辑的各种属性可以改变影片剪辑的位置和显示状态。

_x 和_y 属性代表影片剪辑在场景中的水平坐标和垂直坐标。

_xscale 和_width 属性决定影片剪辑在水平方向上的显示宽度。

_yscale 和_height 属性决定影片剪辑在垂直方向上的显示高度。

_rotation 属性可以旋转影片剪辑。

_alpha 属性代表影片剪辑的透明度，

_visible 属性决定影片剪辑是否可见。

使用关键字_xmouse 和_ymouse 可以获取鼠标光标在屏幕中的坐标位置。

例如设置元件的基本格式属性是：元件名.属性名=属性值。

在场景中放 3 个元件，分别为风车影片剪辑元件、水果影片剪辑元件和按钮元件取实例名为 fc_mc,sg_mc,sx_btn。

添加以下代码：

```
onEnterFrame=function(){
  fc_mc.rotation+=20;
sg_mc._x+=3;
  sg_mc._xscale=sg_mc._yscale+=2;
sg_mc._alpha-=2;
}
_btn.onRelease=function(){
  Delete onEnterFrame;
  Sg_mc._x=275;
  Sg_mc._y=200;
  Sg_mc._xscale=sg_mc._yscale=100;
  Sg_mc._alpha=100;
}
```

以上代码通过帧频事件和按钮事件来设置改变影片剪辑类元件的属性。

5. 影片剪辑的常用方法

常用方法有 Stop,play,gotoandplay,gotoandstop,attachmovie,createemptymovieclip,duolicatemovieclip,startdrag,stopdrag,swapdepths,getnexthighestdepth 等。

这些方法的调用格式是：元件名.方法名(参数列表)

例如：通过元件的 onMouseDown 和 onMouseUp 事件，了解 startDrag 和 stopDrag 方法的调用。

建立一个影片剪辑元件，放入主场景中，起实例名为_mc，在第 1 帧写事件响应代码如下：

```
onMouseDown = function(){
  _mc. startDrag();
}
onMouseUp = function(){
  _mc. stopDrag();
}
```

以上代码的作用是当按下鼠标时，影片剪辑跟随鼠标，松开鼠标时停止跟随。

6. 影片播放控制

控制影片播放流程的命令有许多，如下所示：

stop：使影片停止在当前时间轴的当前帧中。

play：使影片从当前帧开始继续播放。

gotoAndStop：跳转到用帧标签或帧编号指定的某一特定帧并停止。

gotoAndPlay：跳转到用帧标签或帧编号指定的某一特定帧并继续播放。

nextFrame：使影片转到下一帧并停止。

prevFrame：使影片回到上一帧并停止。

stop 命令常常用在帧动作中，以使影片停止并等待用户控制。其他命令常常用在按钮的事件处理函数中。

5.7.2 任务实现

操作一 制作数字滚动效果

【设计思路】

❖ 使用随机语句、复制语句来制作多个数字串从上至下移动。

【操作步骤】

（1）新建一个文件，背景色为绿色。

（2）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，命名为“数字 1”。使用文本工具，设置文本类型为“动态文本”，颜色为深绿色。在“属性”面板的“变量”文本框中输入 num 作为该变量的名称，如图 5-70 所示。

图 5-70

（3）新建图层 2，命名为“动作”，执行“窗口”→“动作”菜单命令，打开“动作”面板，如图 5-71 所示，输入以下代码：

```
num=random(10);
```

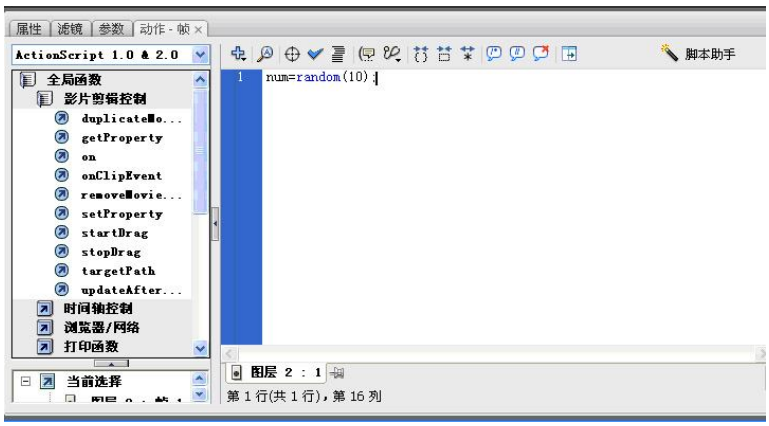

图 5-71

（4）在两个图层的第 2 帧位置分别插入普通帧。

（5）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，命名为“数字 2”。将库中的元件“数字 1”放在舞台中央位置，选中实例，在“属性”面板中的实例名称文本框中输入 myNum，如图 5-72 所示。

图 5-72

（6）选择第 1 帧，打开“动作”面板，输入以下代码：

```
for(var i=1;i<10;i++){
        myNum.duplicateMovieClip("myNum"+i,i)
        this["myNum"+i]._y=myNum._y+i*myNum._height*0.7
        this["myNum"+i]._alpha=10*(10-i)+10
}
```

（7）返回“场景 1”，从“库”面板中将“数字 2”影片剪辑元件拖放到舞台中的任意位置。选中“数字 2”影片剪辑实例，在“属性”面板中为其命名为 myNum，如图 5-73 所示。

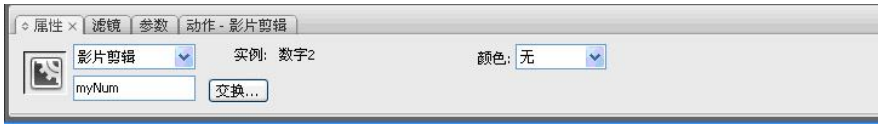

图 5-73

（8）选中实例，然后打开“动作”面板，在其中输入以下代码：

```
onClipEvent(load) {
        _alpha=random(95)+5
        a=((100-_alpha)*4)*0.08+4;
        _y=-myNum._height*10;
        _x=random(400);
        _xscale=_yscale=1.2*((_alpha*0.4)*2+30);
}
onClipEvent (enterFrame) {
        _y+=a;
        if (_y>300) {
        this.removeMovieClip();
        }
}
```

（9）新建图层，命名为“动作”，打开“动作”面板，如图 5-74 所示，在其中添加以下代码：

```
var n=0;
onEnterFrame=function () {
        myNum.duplicateMovieClip("ball"+n++, n);
```

```
        if (n>300) {
        n = 0;
        }
}
```

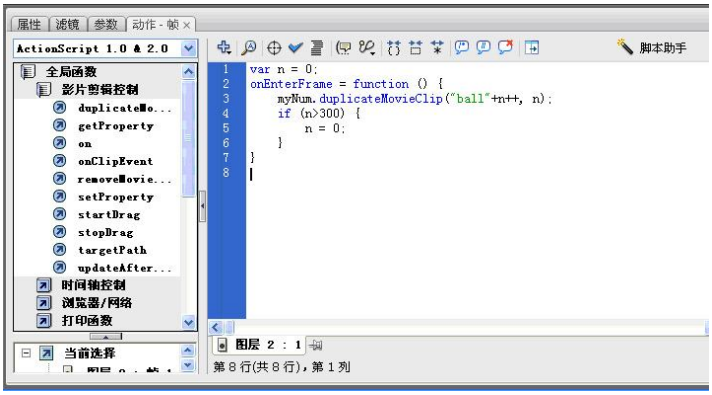

图 5-74

（10）保存文件并预览效果，如图 5-75 所示。

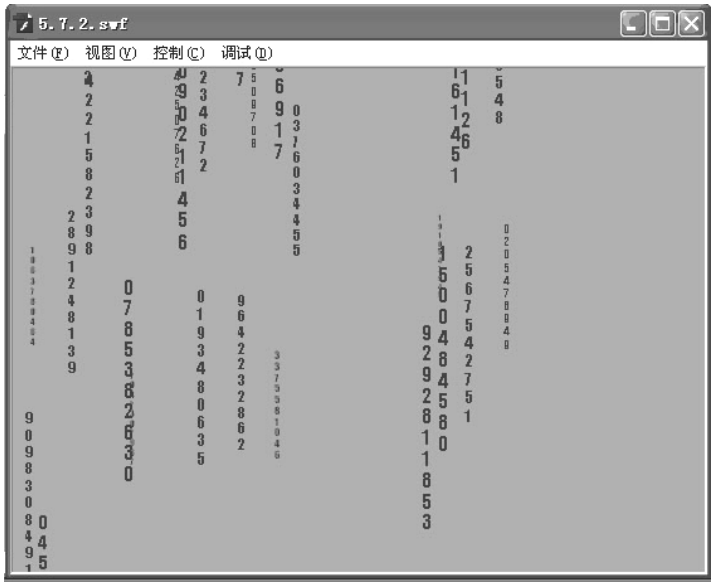

图 5-75

操作二　制作文字波动效果

【设计思路】

❖ 输入一行文字，为每一个文字加上一段运算实例位置的代码。每个文字的位置数值依次减少，以实现波动效果。

【操作步骤】

（1）新建一个文件，背景色为黑色。

（2）执行“插入”→“新建元件”菜单命令，新建一个影片剪辑元件，命名为“动”，

并使用文本工具输入一个白色文字“动”，如图 5-76 所示。

（3）选择文字，执行“复制”、“粘贴”命令，并执行“修改”→“变形”→“垂直翻转”菜单命令，制作文字倒影。选中倒影文字，在“颜色”面板中将 Alpha 值设置为 50%，如图 5-77 所示。

图 5-76

图 5-77

（4）选中上面的文字，打开“滤镜”面板，为文字添加发光效果，颜色为绿色，如图 5-78 所示。

（5）用同样方法制作“画”、“制”、“作”、“案”、“例”、“教”、“程”等另外 7 个影片剪辑元件，如图 5-79 所示。

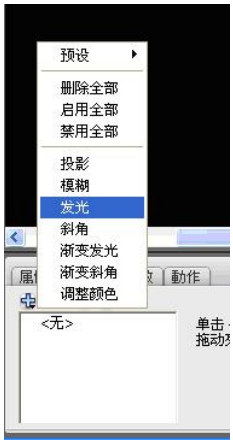

图 5-78

图 5-79

（6）返回“场景 1”，将 8 个元件依次放在场景中，如图 5-80 所示。

图 5-80

（7）选择第一个文字实例，打开“动作”面板，如图 5-81 所示，输入以下代码：

```
onClipEvent (load){
        t=360-0;
}
onClipEvent (enterFrame){
        this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
        this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
        if(t<=360)
            t+=5
            else
            t=t-360
}
```

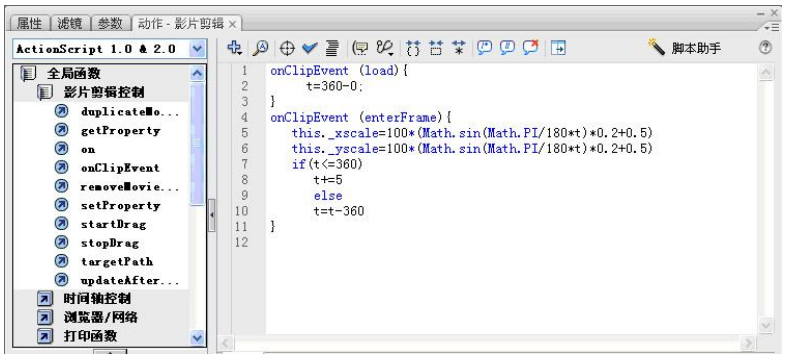

图 5-81

（8）选择第二个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
        t=360-20;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
      t+=5
      else
      t=t-360
}
```

（9）选择第三个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-40;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
      t+=5
```

```
     else
     t=t-360
}
```

（10）选择第四个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-60;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
     t+=5
     else
     t=t-360
}
```

（11）选择第五个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-80;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
     t+=5
     else
     t=t-360
}
```

（12）选择第六个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-100;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
     t+=5
     else
     t=t-360
}
```

（13）选择第七个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-120;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
```

```
   if(t<=360)
      t+=5
      else
      t=t-360
}
```

（14）选择第八个文字实例，打开“动作”面板，输入以下代码：

```
onClipEvent (load){
    t=360-140;
}
onClipEvent (enterFrame){
   this._xscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   this._yscale=100*(Math.sin(Math.PI/180*t)*0.2+0.5)
   if(t<=360)
      t+=5
         else
         t=t-360
}
```

（15）保存文件并预览效果，如图 5-82 所示。

图 5-82

5.8 本章小结

本章通过 6 个任务的实现，阐述了高级动画制作的基本原理，为我们打开了一扇通向高层次动画制作的大门。通过场景和图层，可以制作较长的有多个镜头的影片；通过引导层和遮罩层，可以制作特殊效果的动画；通过按钮，可以实现交互式的动画制作；通过 ActionScript 代码，初步了解了动画与编程相结合的强大威力。

5.9　习题

（1）利用遮罩层制作地球自转动画。
（2）利用引导层制作旋转飞行动画。
（3）制作按钮控制的演示文稿。
（4）制作三维旋转动画。
（5）制作演示课件。