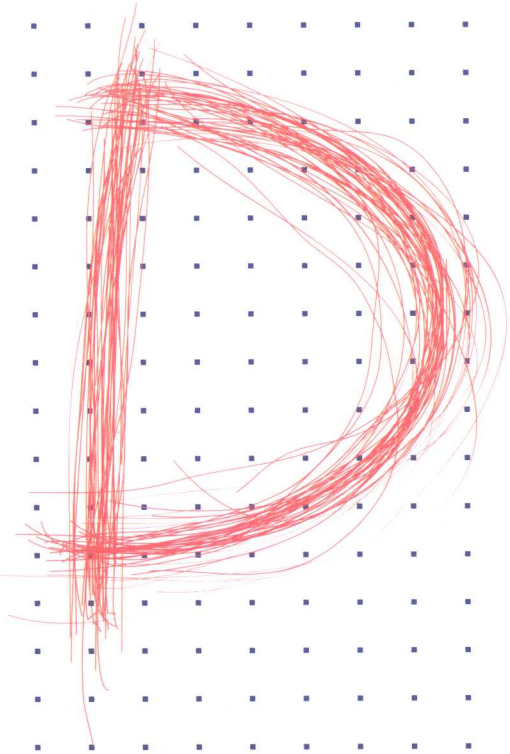
江 苏 省 普 通 高 校 精 品 教 材 建 设 项 目

高等院校传媒艺术实践学程

数字媒体艺术实践学程

Digital Media Art Practice Course of Study

马晓翔 樊飞燕 编著



凤凰出版传媒集团 江苏科学技术出版社

江苏省普通高校精品教材建设项目

高等院校传媒艺术实践学程

数字媒体艺术实践学程

Digital Media Art Practice Course of Study

凤凰出版传媒集团 江苏科学技术出版社

马晓翔 樊飞燕 编著

图书在版编目(CIP)数据

数字媒体艺术实践学程 / 马晓翔等编著. —南京：
江苏科学技术出版社，2010.7
高等院校传媒艺术实践学程
ISBN 978-7-5345-7226-5

Ⅰ.①数… Ⅱ.①马… Ⅲ.①多媒体－应用－艺术－
高等学校－教材 Ⅳ.①J-39

中国版本图书馆 CIP 数据核字(2010)第 037650 号

数字媒体艺术实践学程

编　　著　马晓翔　樊飞燕
责任编辑　徐晨岷
责任校对　郝慧华
责任监制　刘　钧

出版发行　江苏科学技术出版社(南京市湖南路 1 号 A 楼，邮编：210009)
网　　址　http://www.pspress.cn
集团地址　凤凰出版传媒集团(南京市湖南路 1 号 A 楼，邮编：210009)
集团网址　凤凰出版传媒网 http://www.ppm.cn
经　　销　江苏省新华发行集团有限公司
照　　排　江苏凤凰制版有限公司
印　　刷　江苏凤凰通达印刷有限公司

开　　本　889×1 194　1/16
印　　张　6
字　　数　120 000
版　　次　2010 年 7 月第 1 版
印　　次　2010 年 7 月第 1 次印刷

标准书号　ISBN 978－7－5345－7226－5
定　　价　45.00 元

江 苏 省 普 通 高 校 精 品 教 材 建 设 项 目

《高等院校传媒艺术实践学程》
编委会

主 编

张承志
南京艺术学院传媒学院院长、教授

王 方
南京艺术学院传媒学院副院长、副教授

编 委

黄晓白
南京艺术学院副书记、副研究员

刘伟冬
南京艺术学院副院长、教授

许 永
南京艺术学院传媒学院教授

庄 曜
南京艺术学院传媒学院副院长、教授

金昌庆
南京艺术学院传媒学院教授

吕 江
南京艺术学院传媒学院动画系主任、副教授

马晓翔
南京艺术学院传媒学院数字媒体艺术系主任、博士

《高等院校传媒艺术实践学程》
序　言

书籍是一种传播思想文化的重要媒介，当这套《高等院校传媒艺术实践学程》作为立项的“江苏省高校精品教材”，终于付梓印刷时，当它带着油墨的清香呈现于课堂和工作室，当它放在图书馆的书架上、装在学子们的书包里、展开于教师的讲台上时，其中的每一本书，每一个字句都凝聚着作者们的心血和付出。

传媒作为近年热门的专业而备受瞩目，当下的传媒业融合了最新的数字技术、视觉和听觉艺术的表现形式，将影像、动画、音乐等不同学科结合起来，形成了新的媒体、新的传播业态和新的发展趋势。《高等院校传媒艺术实践学程》中的每一本教程，都是从传媒学科中有较强艺术实践性质的课程遴选出来的，经过作者们反复的推敲、总结和梳理，运用了大量在实践教学中积累的经验和案例、课题编写而成，为学生在课程的学习中提供理论上的梳理和实践上的指导。

这套书的一个特点就是涉及的课程是以传媒艺术所涉猎的学科横向展开，而非以单一学科纵向集结的方式。《高等院校传媒艺术实践学程》涉及广播电视编导、动画、摄影、广告等诸多传媒学科领域的实践课程，这些学科之间形成互补和支撑，构建了大概念的传媒视域，整套丛书内容丰富，每一本书都是一扇向传媒艺术打开的窗口。

这套书的另一个特点就是紧紧围绕着“艺术”和“实践”两个关键词。

“艺术”是探讨传媒中不同媒介形式的表现手段，而在传媒艺术中，艺术的最终实现往往要依靠的是现代数字技术。技术和艺术的交融，是传媒艺术的一个特点。所以，本套书在对“艺术”的探寻中，同样有着非常具体的技术操作的解读和分析。

同样，“实践”是艺术得以实现的过程，思想和艺术的审美、表达最终都要通过“实践”得以完成，“实践”也是一切学习必须经历的过程。本套教程是从南京艺术学院传媒学院的艺术实践课程中总结、挖掘和梳理而成的，大量的案例和课题都是教师和作者平时教学成果的积累。

《高等院校传媒艺术实践学程》是老师们辛勤耕耘的成果，也希望成为学生学习传媒艺术的好教材，希望它对学生好学好用，够学够用。

南京艺术学院传媒学院院长　张承志

2009 年 11 月 16 日

前　言

《数字媒体艺术实践》是一本综合了网络动画、交互动画、网页特效、控件、游戏设计与制作于一体的实践教程。本教程从数字媒体艺术实践所涉及的应用范畴中精选了若干实例作为典型案例进行讲解，将设计、制作以及编程基础融汇贯通，潜移默化地解析制作步骤，让数字媒体艺术专业的学生、从事数字媒体艺术行业的人员、数字媒体艺术爱好者能够轻松地进行数字媒体艺术设计与制作。

本教程共六个课题，主要是 LOADING 动画的设计与制作、导航的设计与制作、多媒体控制界面的设计与制作、网页特效的设计与制作、交互动画设计与制作、项目实验。每个课题具有较强的针对性，以知识点为切入点，通过经典解读和课题步骤将一个课题讲解得透彻、到位，对于致力于数字媒体艺术实践创作的人员来说，这无疑是一本具有实践性指导意义的教程。

目　录

课题概述

Loading，英文是指“装载”的意思。所谓Loading动画主要是指在主体动画播放之前为装载而专门制作的一个过程动画。用户在互联网上观看Flash电影时，有时由于主体文件太大，或者网速的限制，动画需要加载一段时间后才能播放。而加载所需的时间对于观看者来说是未知的，多数用户都不会有足够的耐心长期等待无法预期的空白网页。Loading动画就是要告知用户目前Flash电影的装载情况，无论Loading动画多么简洁，它都可以起到缓解用户等待时的急躁心理的作用。如果能将Loading动画制作得精美动感，即使等待时间较长，也能吸引用户继续等待。

在使用本地机时，不会存在Loading的问题，但我们可以使用Flash软件模拟网页下载环境进行调试。

Loading动画分成两种，一种是没有下载进度提示的，另一种是精确显示下载进度的。我们可以根据需要选择使用，也可以结合两者以得到更好的效果。①

课题要求

掌握Loading动画制作的原理以及相关的属性与函数，如_framesloaded、_totalframes属性和ifFrameLoaded()、getBytesLoaded()、getBytesTotal()、getTimer()函数。

课题目标

通过对相关的AS语言和源代码的学习，学会Loading动画的设计与编程。

知识点

_framesloaded、_totalframes属性和ifFrameLoaded()、getBytesLoaded()、getBytesTotal()、getTimer()函数，动态文本的设置。

一、_framesloaded

_framesloaded是电影剪辑的属性，用来获取电影剪辑中已经下载的帧数，当然大多应用于电影剪辑的属性都可以应用于整部动画。

```
if(_root.mc._framesloaded>100){
  _root.gotoAndPlay(1)
}
```

在编程中将不允许对_framesloaded属性进行赋值，如果你想当下载的帧数等于100时根目录开始回放的话，请按下例制作。

```
if(_root.mc._framesloaded==100){
  _root.gotoAndPlay(1);
}
```

二、_totalframes

_totalframes属性是用来获取电影剪辑实体的总帧数，也可以用来获取动画的总帧数。在下例中会看到它的用法。

```
i=_root.mc._totalframes;
if(_root._currttentframe==i){
_root.stop();
}
```

程序中将电影剪辑实体的总帧数赋值赋予了变量i，而当主场景的动画播放指针播放到与电影

① http://www.7880.com/Info/Article-234626c0.html 2009.1.1

剪辑中的总帧数相同的数目时，动画停止播放。此属性同样为非赋值属性。

三、ifFrameLoaded()

ifFrameLoaded()函数也是用来获取已经下载的帧数的，与_framesloaded 不同的是它用于一个简单地行为来描述已下载的帧数。而且此函数似乎是专为 Loading 设计，它位于 Basic Actions 指令集，指令名称为 If Frames Is Loaded。以下实例将构成一个最为简单地 Loading。

```
ifFrameLoaded(_totalframes){
  gotoAndPlay(3);
}else{
gotoAndPlay(1);
}
```

将此程序加于影片的第 2 帧，可用于所有动画的预载技术。意思为当装入的帧数为总帧数时开始播放第 3 帧，如果不然，播放第一帧。在 Flash5 以后开始使用更多地函数和属性，所以此函数不推荐使用。

四、getBytesLoaded()

getBytesLoaded()可获取电影剪辑实体的已下载字节数，如果是外部动画将返回动画的总字节数。getBytesLoaded 用于更加精确的 Loading 设计，因为它并不像 _framesloaded 属性是获取影片的总帧数，而是以字节作为单位获取。如果说动画的最后一帧将是一个大型的图像或是声音文件的话，那么 _framesloaded 所获得的百分比将不准确，getBytesLoaded 有效地弥补了此方面的不足。例：

```
i=_root.getBytesTotal();
if(_root.getBytesLoaded()>=1000000){
  n=_root.getBytesLoaded();
  if(n<=i/4){
    _root.stop();
    trace(" 下载了 1M，还不到四分之一，动画太大，下载时间会很长，是否继续？ ")
  }
}
```

此句的意思为当动画下载到 1MB 时，比较是否已经下载了动画的四分之一，如果还不到四分之一，则停止动画的播放，在调试窗口显示“下载了 1M，...”等字符串如果已达到四分之一，则动画继续加载。此例的另一特点是，停止的地方如果有插入电影剪辑的话，电影剪辑将不会停止播放。也可以通过动态文本显示已经下载的字节，假设在动画的主场景中有一个变量名为 text 的动态文本变量，那么，例：

```
_root.text=_root.getBytesLoaded();
if(_root.getBytesLoaded()>=_root.getBytesTotal()){
  gotoAndPlay(3);
}else{
  gotoAndPlay(1);
}
```

动态文本框会动态显示已经下载的字节数为观众服务，观众也会了解在动画的下载过程中动态的进度。

五、getBytesTotal()

getBytesTotal()函数是用来获取动画或是电影剪辑的总字节数，当然我们可以通过对文件的大小来观察动画的总字节数，但对于网络上使用浏览器的观众来说，动态显示文件大小是很有必要的。还有，如果想观察动画中电影剪辑的体积就只有靠 getBytesTotal()函数了。

```
If(_root.getBytesTotal()>=1000000){
_root.stop();
}
```

这个程序的意思是当动画的总字节超过 1M 时停止动画播放。

六、getTimer()

getTimer()函数用来获取电影剪辑或是动画的已经播放时间数，此函数并不仅仅应用于 Loading 的制作，在今后的学习过程中还会接触到它。getTimer()函数获取的时间是以毫秒作为计算单位的，一般在程序制作过程中还会对它除以 1000 来取得

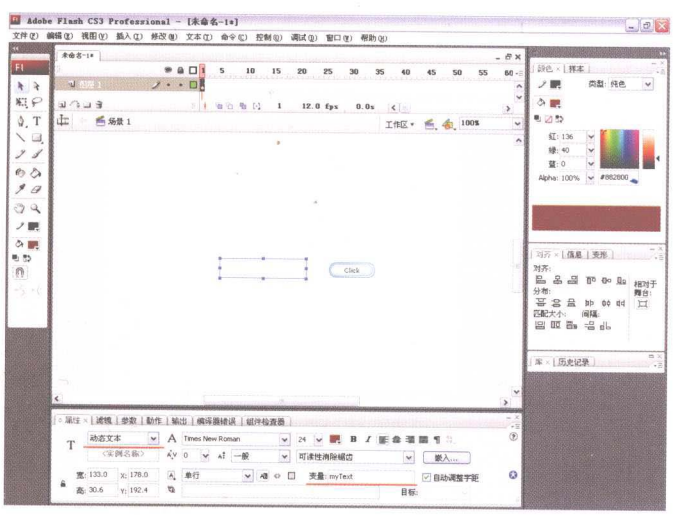


图 1-0-1　在属性面板中设置动态文本框的变量名为 myText

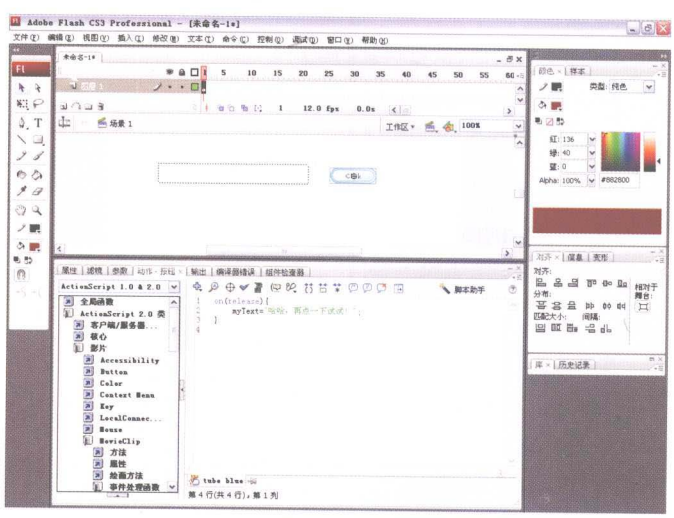


图 1-0-2　为按钮输入 AS 代码

秒，这样更加符合对于时间播放程序的显示。假设动画中有一个 text 的动态文本框变量。例：

```
text=getTimer()/1000;
```

通过帧循环或是其他的诸如 OnClipEvent(enterFrame)等行为的控制动态的显示动画播放的时间过程。又例如：

```
text=getTimer()/1000;
if(text>=10){
  gotoAndPlay(3);
}else{
  gotoAndPlay(1);
}
```

假设此程序位于动画的主场景的第 2 帧。那么，当开始播放 10 秒钟之后才会正式开始播放，主动画（假设主动画自第 3 帧开始）不然只会在第 1 帧与第 2 帧之间循环。

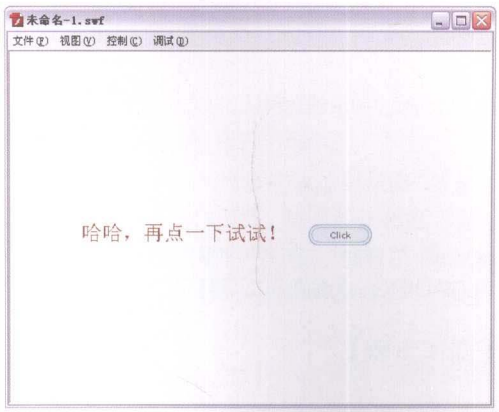


图 1-0-3 运行结果

七、动态文本

Flash 中的动态文本是一种比较特殊的文本，可以在 flash 影片播放过程中通过 ActionScript 脚本动态改变文本框中内容。比如在制作游戏动画时，有时需要一个文本框来记录并显示用户的游戏得分，这个得分是根据游戏情况动态而改变的。这时候，我们就需要用到动态文本。

创建动态文本的方法与创建一般文本的方法相似，使用文本工具在编辑区拖出一个文本对象，然后只需在属性面板中选择“动态文本”即可。而为了与 ActionScript 联系起来，我们要在属性面板中设置动态文本的变量名，这个变量名必须与 ActionScript 脚本中对应的变量名一致，这样才能进行动态的赋值。

举个简单的例子，在舞台中布置一个按钮，和一个动态文本，设置动态文本的变量名为“myText”，如图 1-0-4 所示。

选择按钮，按【F9】打开动作面板，输入如下动作代码：

```
on(release){
    myText=" 哈哈，再点一下试试！ ";
}
```

如图 1-0-2 所示。

运行测试，当点击按钮后，页面上就会显示文字“哈哈，再点一下试试！”语句，这样就实现了动态文本框与 ActionScript 脚本的响应。如图 1-0-5 所示。

以上是与 Loading 动画相关的属性与函数用法的介绍，下面将通过具体的实例来进行讲解。

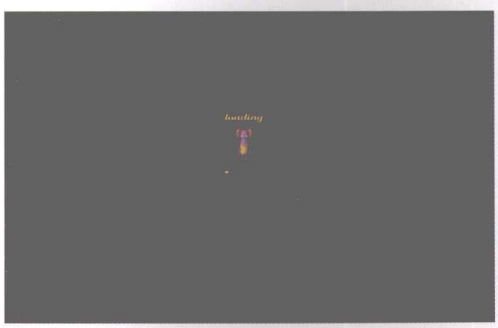


图 1-0-4 英国心脏基金会 Loading 动画截图
http://food4thought.bhf.org.uk/

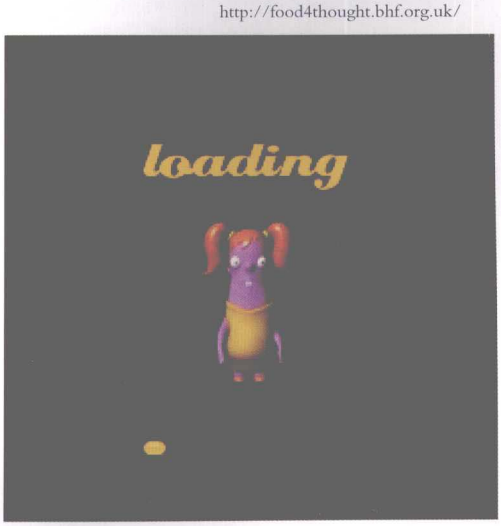


图 1-0-5 英国心脏基金会 Loading 动画截图（局部）

经典解读

这是英国心脏基金会所属网站的 Loading 动画。该 Loading 动画将卡通的动画角色、趣味的音乐和 Loading 进程结合起来，是一个精确显示进度的 Loading 动画。深灰色的场景将 Loading 动画的生动形式与下载时间的精确定位衬托得十分鲜明，整个 Loading 动画的形式赋有趣味，并设计有效。卡通的三维小人妙趣横生，而小人上方的文字“loading”和下方的状态条则有效地表现了网站下载的时间状况。另外，该 Loading 动画运用了 Flash 内的 AS 语言，为状态条的灵活运行创造了条件。

课题步骤

子课题一　Loading 动画的设计

例 1：设计一个 loading 动画，实现让文字“LOADING…”绕圆周循环运动

【操作步骤】

打开Flash cs3 创建一个新文件，设置文件属性为 400 × 300 像素，背景色为 #006699。如图 1-1-1 所示。

点击工具箱上“文本工具”按钮，在主场景上输入文字 “LOADING…”，设置字体为 Arial TUR，字号为 20，文字颜色为 #6699FF，粗体。

选中文字，按【Ctrl+B】将文字打散。如图 1-1-2 所示。

将文字“L”转换成图形元件。选中文字“L”，右击选择“转换为元件”，命名为“L”，类型为“图形”，如图 1-1-3 所示。同样的方法将其他文字转换成图形元件，并分别命名为“L”，“O”，“A”，“D”，“I”，“N”，“G”，“.”。将主场景中的文字删除。

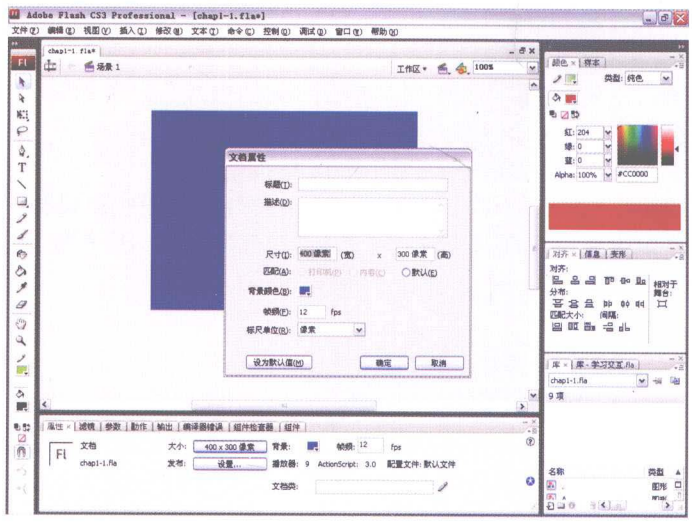


图 1－1－1　创建新文档并设置文档属性

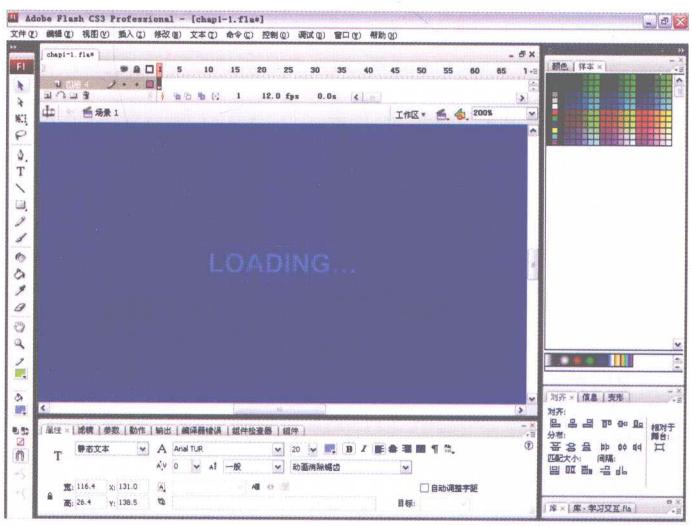


图 1－1－2　输入文字“LOADING”

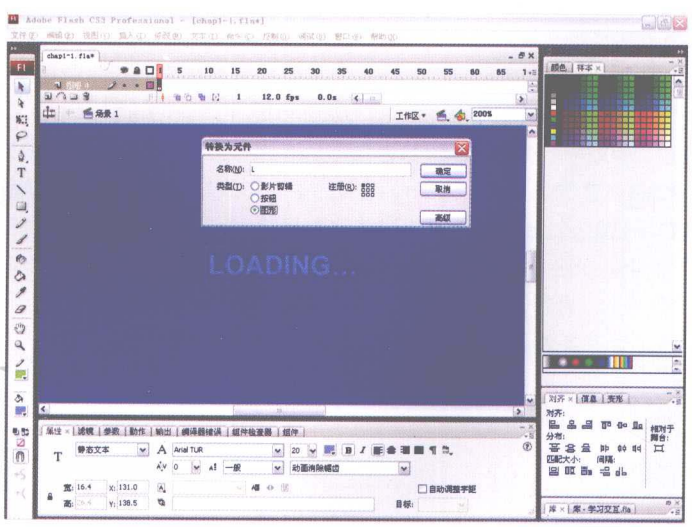


图 1－1－3　将文字“L”转换为图形元件

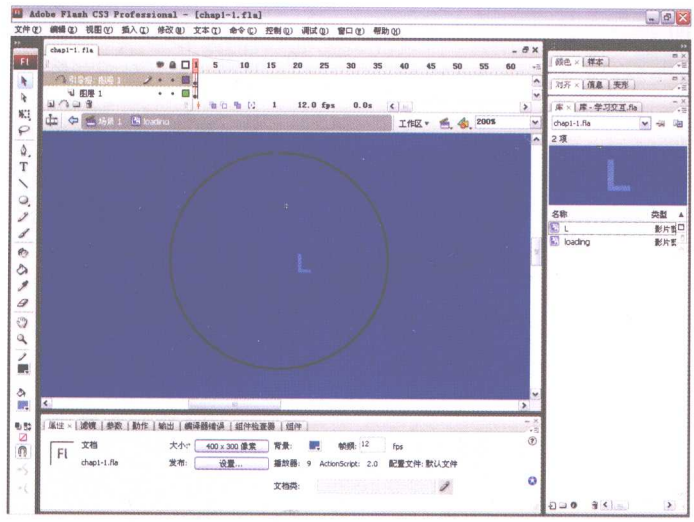


图 1－1－4　绘制引导线

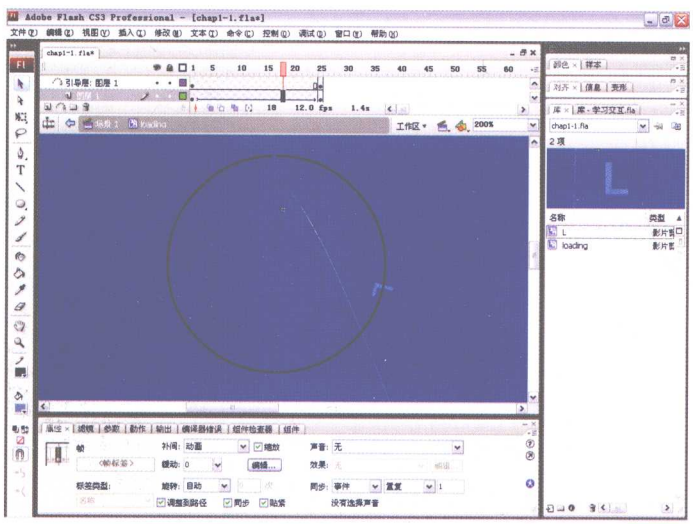


图 1－1－5　制作路径动画

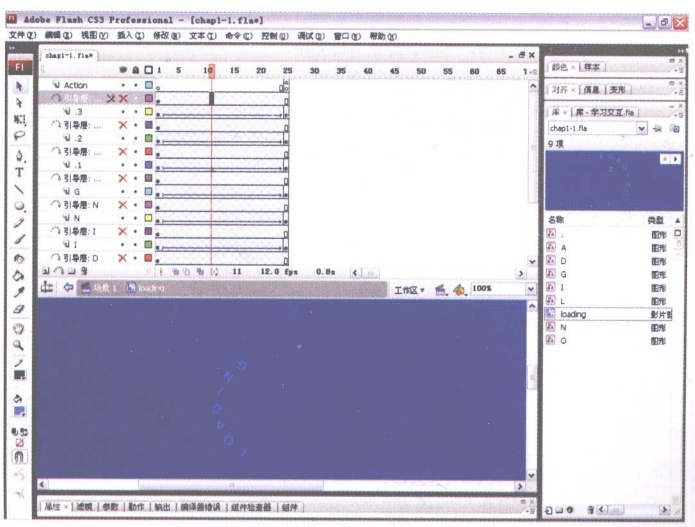


图 1－1－6　为其他文字制作路径动画

按 Ctrl+F8，创建影片剪辑，命名为“loading”。

将图层 1 重命名为“L”，将图形元件“L”拖入影片剪辑“loading”的图层 1 的第一帧中。

选中图层 1，点击“添加运动引导层”按钮，在图层 1 上添加一引导层。选择绘制椭圆工具，按住【Shift】键，在引导层上绘制一直径为 150 像素的圆形，并将填充部分删除，只保留圆周。选中第 25 帧，按【F5】插入帧，将时间轴延长至 25 帧。在圆的上顶点（一定要是上顶点）用橡皮擦（笔头最小），擦出一个缺口。如图 1-1-4 所示。

创建元件“L”的运动引导层动画。将图层 1 的第一帧的文字“L”吸附到引导线缺口的左侧。在第 25 帧创建关键帧，将第 25 帧的文字“L”吸附到引导线缺口的右侧。并创建“补间动画”，选择“调整到路径”。如图 1-1-5 所示。

这样就实现了使文字“L”绕圆周运动。下面继续为其他文字制作圆周运动。

在“引导层:L”之上添加图层 3，将图层 3 重命名为“O”，将图形元件“O”拖入图层 3 的第一帧中。在图层“O”上添加一引导层，名字为“引导层:O”。

选择“引导层:L”的第一帧，右击选择“复制帧”，选择“引导层:O”的第一帧，按【Ctrl+Shift+V】（粘贴到当前位置）。按【Ctrl+T】，打开“变形”窗口，设置圆环“旋转”20 度。将“引导层:L”隐藏。

将图层“O”的第一帧的文字“O”吸附到引导线缺口的左侧。在第 25 帧创建关键帧，将第 25 帧的文字“O”吸附到引导线缺口的右侧，并创建“补间动画”，选择“调整到路径”。

重复上述方法，分别为“A”，“D”，“I”，“N”，“G”，“.”，“.”，“.” 制作环绕动画。只是在 STEP10 中的旋转度数每次增加 20 度。时间轴如图 1-1-6 所示。

新建图层 Action，在第 25 帧插入关键帧，打开动作面板，输入 gotoAndPlay(1);。

返回主场景，将影片剪辑“Loading”放进主场景的图层 1 的第一帧中，居中显示。

保存为 chap1-1.fla，并运行。

子课题二　非精确显示进度的 Loading 动画制作

（一）GIF Loading 动画制作

GIF 动画实际上就是一系列连续出现的静态图像，每一幅静态图像称为一帧，当这些帧连续、快速地显示时就会形成动画效果。利用 Photoshop 和 ImageReady 软件可以轻松地制作 Gif Loading 动画。

首先利用 Photoshop 软件制作出一系列规则变化的静态图片，将其统一保存在一个文件夹内，然后在 ImageReady 中，将此文件夹导入。

例 2：运用 Photoshop 和 ImageReady 软件制作光条绕圆周循环闪烁显示的 GIF Loading 动画。

【操作步骤】

打开 Photoshop，新建一空白文档，设置其属性如图 1-2-1 所示。

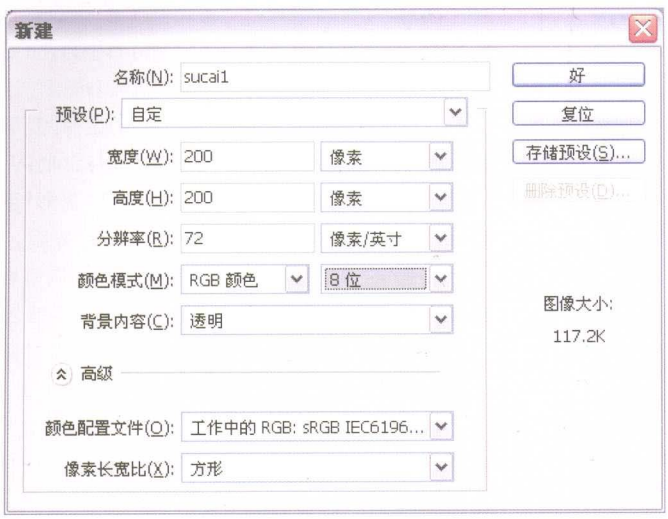


图 1－2－1　新建 Photoshop 文档

【视图】>【标尺】，显示标尺。在标尺上点击鼠标右键，选择像素，此时标尺以像素为单位显示。将鼠标分别移动到上侧和左侧的标尺上，按住鼠标左键，将参考线拖动到 100 像素处。此时，两参考线的交点即是画布的中心。再分别从上侧和左侧的标尺上拖出 2 根参考线（共 4 根）到 75 像素和 125 像素处，形成一个长为 50 像素的正方形，如图 1-2-2 所示。

【视图】>【对齐到】>【参考线】。在工具箱中选择【椭圆工具】，按住【Shift】键，紧贴方形参考线在其内部拖出一个正圆，如图 1-2-3 所示。

新建图层 2，将前景色设置为 #4F4F4F。选择【圆角矩形工具】，在圆形正上方拖出一长条。如图 1-2-4 所示。

在图层 2 上右击，选择【复制图层】为“图层 3”。按【Ctrl+T】设置“自由变换”，在上侧属性栏中设置旋转 30 度，按【Enter】键确定。选择【移动工具】，将长条移动到如图 1-2-5 所示的位置。

按上述步骤，绕圆形继续制作 10 个长条。结果如图 1-2-6 所示。

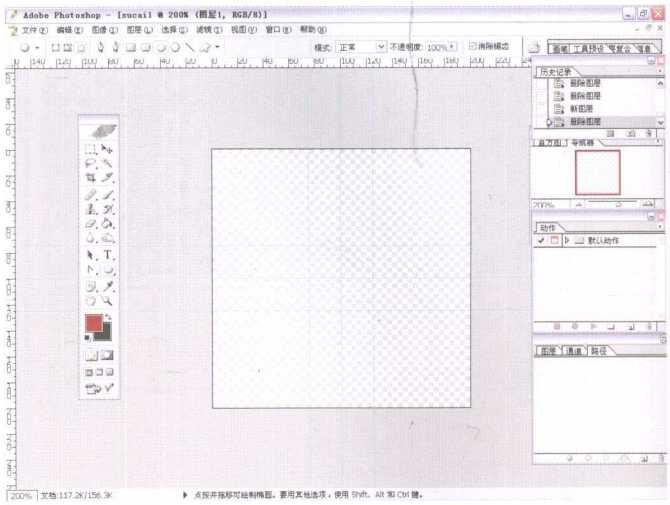


图 1－2－2　设置参考线

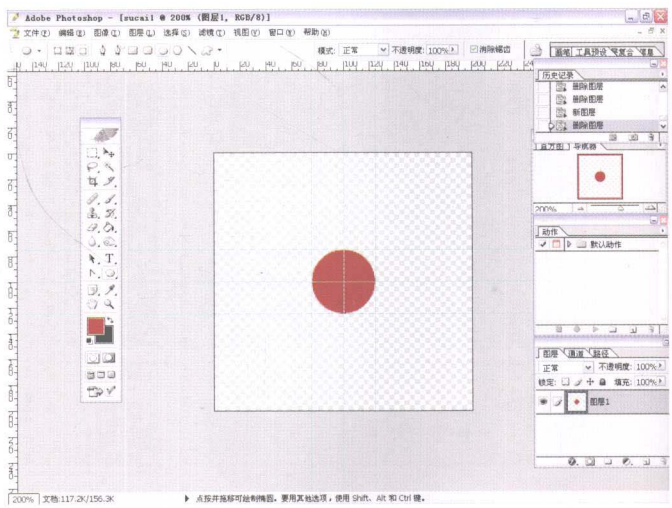


图 1－2－3　绘制正圆

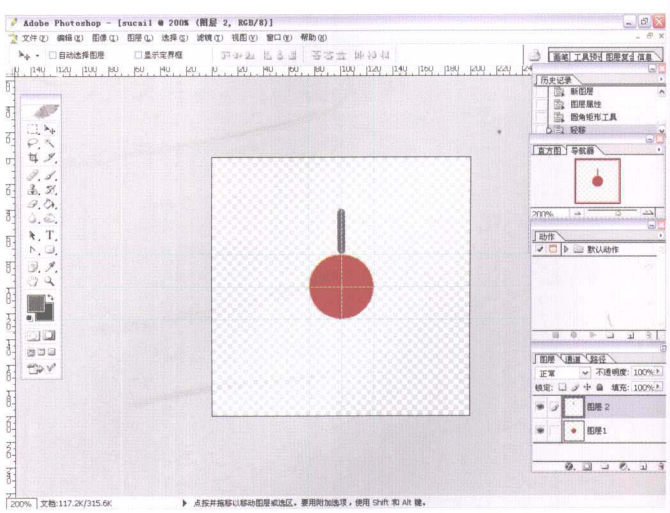


图 1－2－4　绘制光条

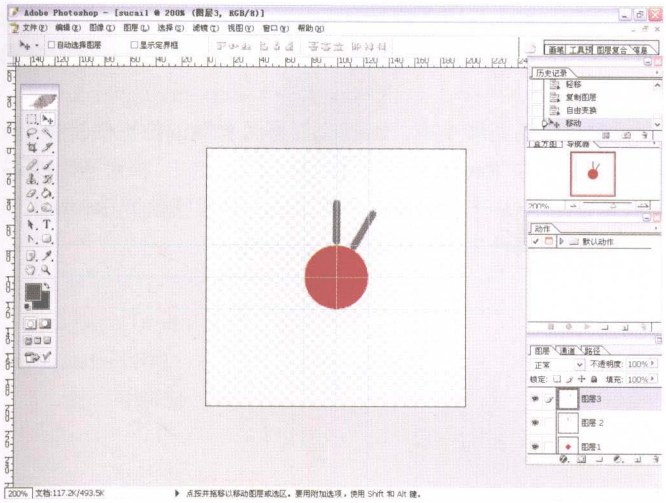


图 1-2-5　复制光条

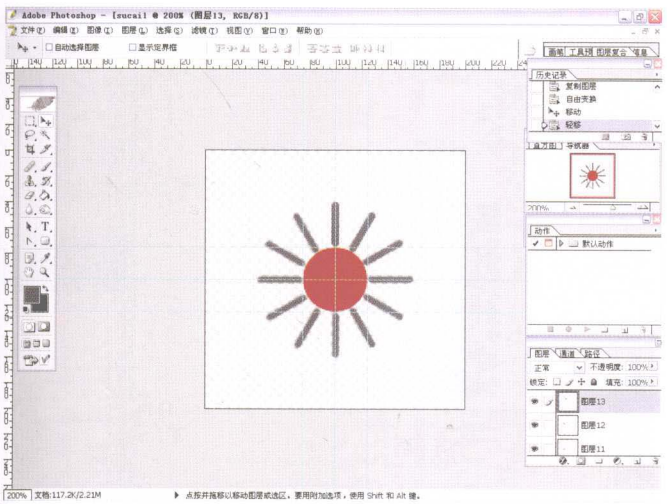


图 1-2-6　复制光条

设置前景色为白色，选中图层 2，用【油漆桶工具】将图层 2 的长条设置为白色。设置前景色为浅灰色(#E1E1E1)，选中图层 3，用【油漆桶工具】将图层 3 的长条设置为浅灰色。设置前景色为灰色(#A9A9A9)，选中图层 4，用【油漆桶工具】将图层 4 的长条设置为灰色。如图 1-2-7 所示。

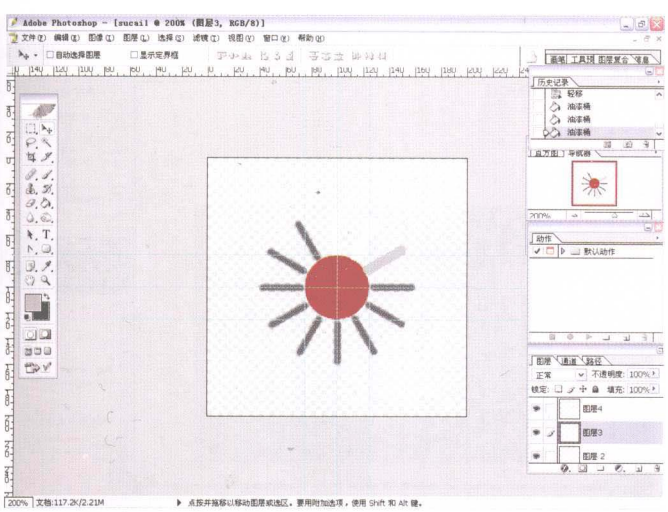


图 1-2-7　为光条设置渐变色

将图层 1 设置为隐藏。新建一文件夹为 sucai，将文件保存到 sucai 文件夹下，文件名为 sucai1.psd。

【文件】>【存储为】，将 sucai1.psd 文件另存为 sucai2.psd。在sucai2.psd 文件上修改，将白色、浅灰、灰色长条的位置依次向左旋转。按此方法分别制作图片，命名为 sucai3.psd，sucai4.psd，sucai5.psd，sucai6.psd，sucai7.psd，sucai8.psd，sucai9.psd，sucai10.psd，sucai11.psd，sucai12.psd。

打开 ImageReady 软件，【文件】>【导入】>【作为帧的文件夹】，选择 sucai 文件夹将其导入。如图 1-2-8 所示。

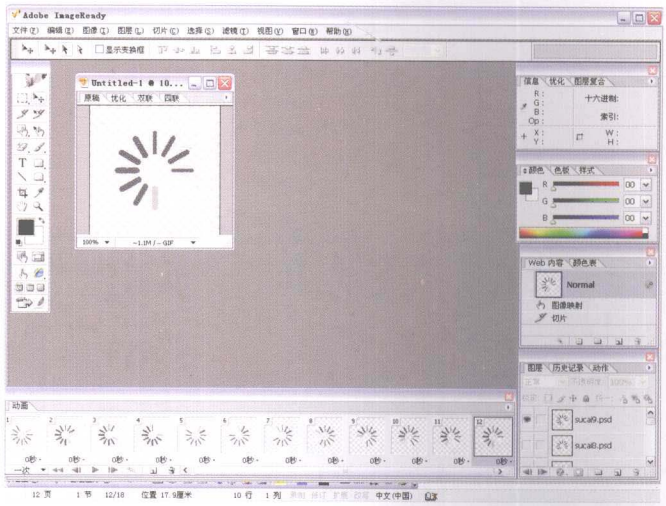


图 1-2-8　将文件导入 ImageReady

点击【动画】面板中的播放键，预览 GIF 动画效果。【文件】>【存储优化结果】，将文件保存为 Gif_loading.gif。

这样一个 GIF Loading 动画就完成了。

(二) Flash Loading 动画制作

例 3：在例 1 中，我们已经制作了一个 Loading 动画，现在将其应用到主动画之前，以达到 Loading 的效果。

【操作步骤】

打开文件 chap1-1.fla，将其另存为 chap1-3.fla。如图 1-3-1 所示：

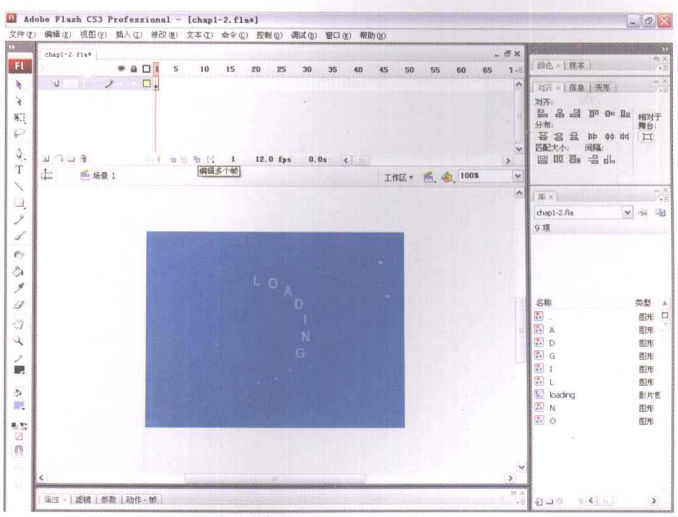


图 1-3-1　将 chap1-1.fla 另存入 chap1-3.fla

选中图层 1 的第 2 帧，按【F6】插入关键帧。点击时间轴左下方的【插入图层】按钮，在图层 1 上方添加图层 2，将图层 2 拖至图层 1 下方，并在图层 2 的第 3 帧上插入关键帧。

将已准备好的比较大的 Flash 文件打开，这里选择课题三的案例文件“视频控制.fla”文件。在时间轴上右击，选择【选择所有帧】菜单，再右击，选择【复制帧】。如图 1-3-2 所示。

返回 chap1-3.fla 文档的主场景，在图层 2 的第 3 帧上右击，选择【粘贴帧】，将所有帧都复制到以第 3 帧开始的时间轴上。如图 1-3-3 所示。

现在调整文档的大小以适应视频控制.fla 的文档大小。在场景上单击，打开【属性】面板，调整其大小为 550 × 400，如图 1-3-4 所示。

然后调整图层 1 中，“loading”影片剪辑的位置使其位于舞台中央。选中“loading”，打开【对齐】面板，选中“相对于舞台”按钮，单击“水平中齐”和“垂直中齐”按钮，这样就使得“loading”影片剪辑位于舞台中央。如图 1-3-5 所示。

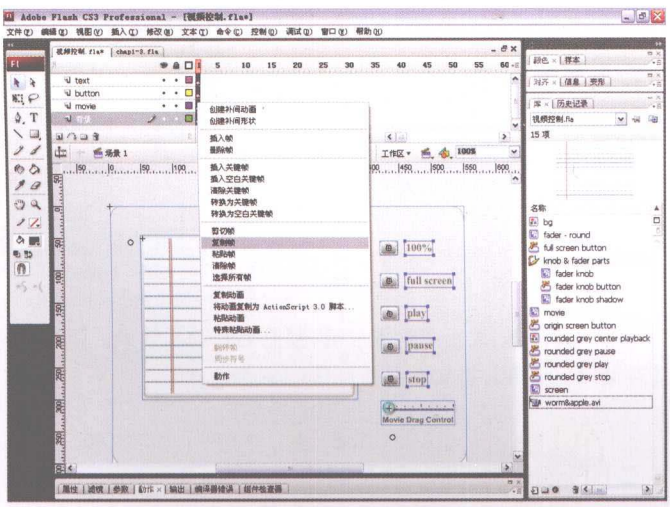


图 1-3-2　复制帧

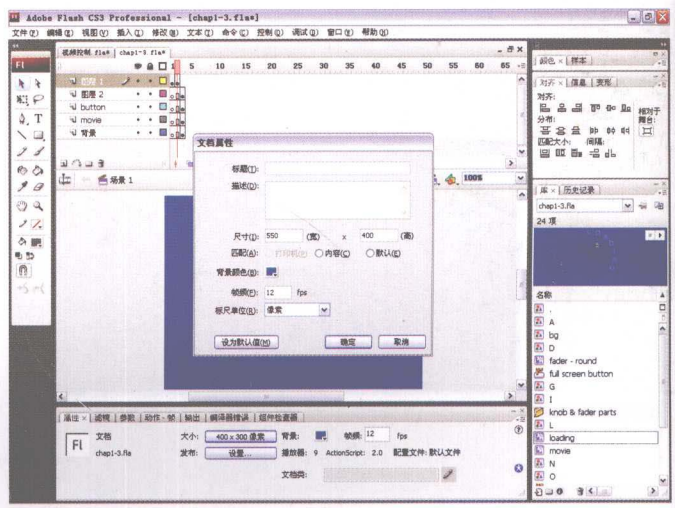


图 1-3-4　设置页面属性

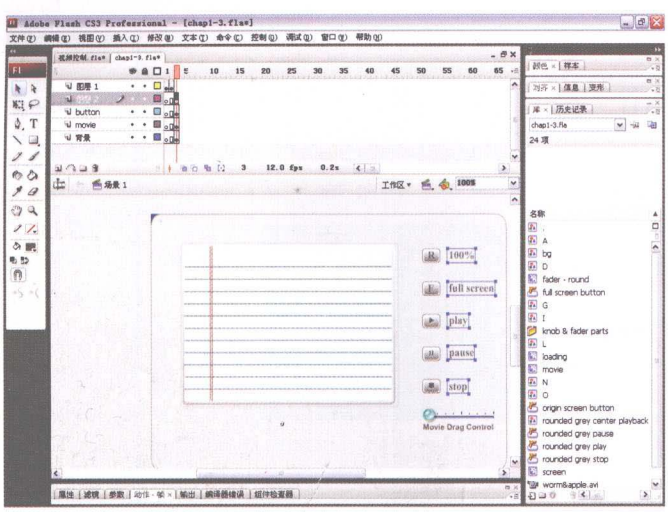


图 1-3-3　粘贴帧

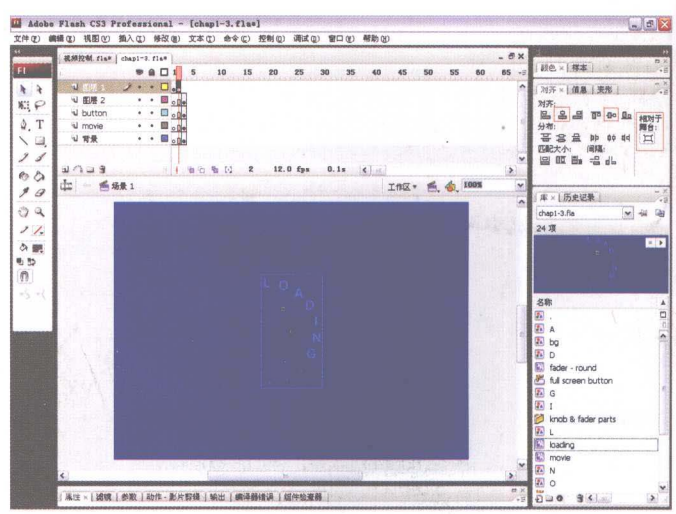


图 1-3-5　居中对齐

STEP4：选择图层 1 的第 2 帧，按【F9】打开动作面板，在其中输入如下代码：

```
if(_framesloaded>=_totalframes){
gotoAndPlay(3);}
else{
gotoAndPlay(1);
}
```

或者使用如下代码：

```
text=getTimer()/1000;
if(text>=10){
  gotoAndPlay(3);
}else{
  gotoAndPlay(1);
}//loading 动画运行 10 秒后，继续播放第三帧。
```

按【Ctrl+Enter】测试影片，此时我们发现影片在 1、2、3 帧之间不断闪烁，这是由于视频短片被嵌套在了“movie”层的“movie”影片剪辑中，而主时间轴上实际只有 3 帧，在代码测试的时候只判断主时间轴的帧数，而不判断影片剪辑中的帧数。所以我们只要将主实践轴的帧延长到“movie”中视频播放完毕即可。

在场景中选中“movie”实例，双击鼠标进入编辑状态，查看其帧数为 849 帧，如图 1-3-6 所示。

返回主场景中，将“背景”层、“movie”层、“button”层和“图层 2”四层都延长至 851 帧，如图 1-3-7 所示。

保存文件，测试影片。

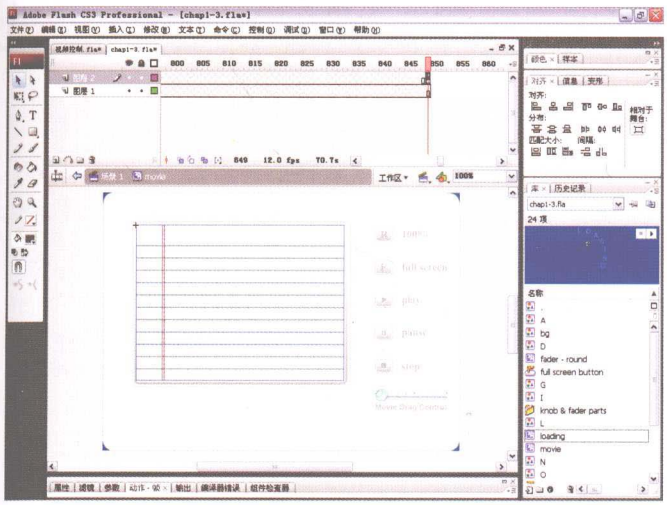


图 1－3－6　查看“movie”实例的帧数

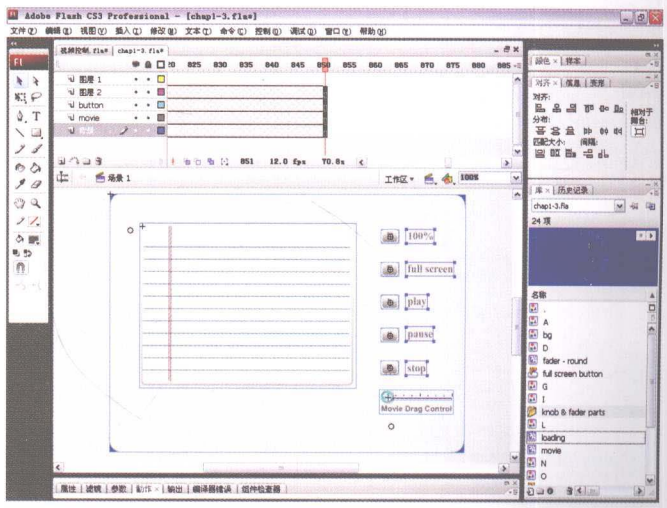


图 1－3－7　延长时间轴至 851 帧

子课题三　精确显示进度的 Loading 动画的制作

精确显示进度的 Loading 动画可以使用户一目了然地把握装载和等待的时间。

例 4：精确显示进度的 Loading 动画。

为方便测试，一般制作 Loading 动画是在主动画完成之后制作。如果在主动画制作之前就有制作 Loading 动画的打算，我们可以把主场景的前 10 帧提前预留出来，如果没有预留，我们可以通过下例中的方法实现。

【操作步骤】

打开要添加 Loading 动画的主动画，仍以“视频控制.fla”为例，将其另存为 chap1-4.fla。如图 1-4-1 所示。

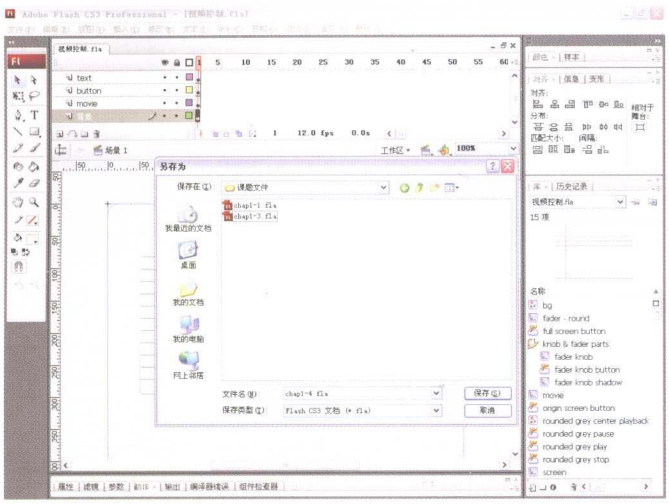


图 1－4－1　另存 fla 文件

在时间轴上右击 >【选择所有帧】，按住鼠标左键将所有帧往后移动 10 帧，至第 11 帧。再新建三个层，从上到下分别命名为 action、文字、loading。如图 1-4-2、图 1-4-3 所示。

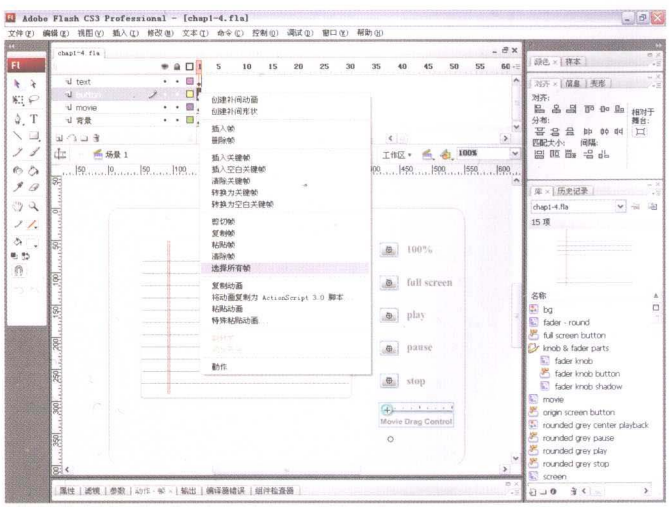


图 1－4－2　选择所有帧

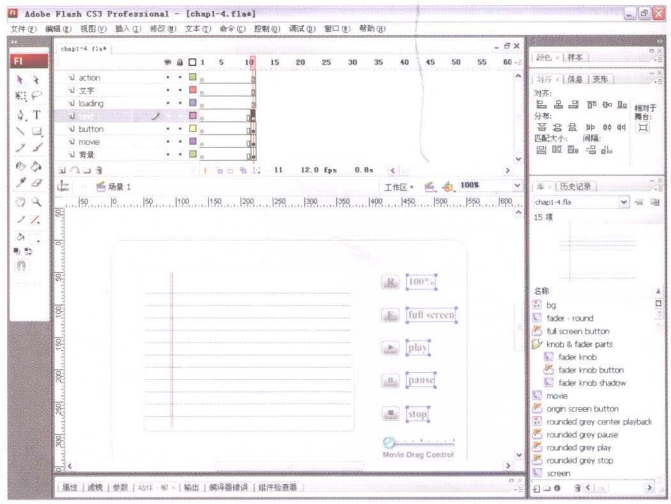


图 1－4－3　新建三个图层

根据例 3 知道，“视频控制.fla”中视频长度为 849 帧，因此我们将“背景”层、“movie”层、“button”层和“text”层都延长至 859 帧，如图 1-4-4 所示。

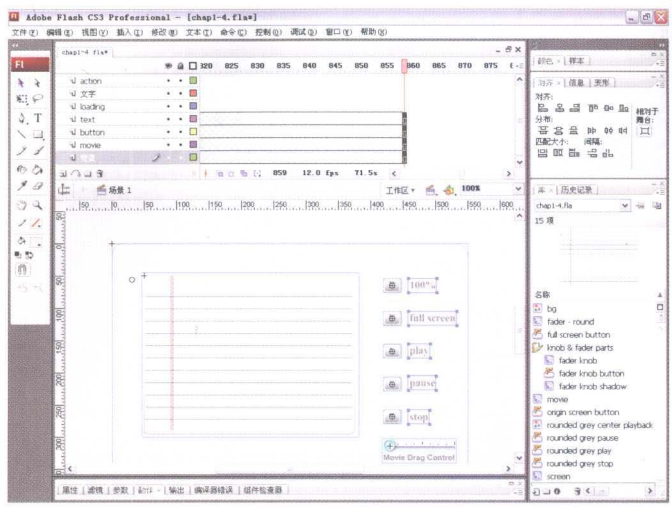


图 1－4－4　延长时间轴至 859 帧

下面制作进度条变化的影片剪辑，在这里我们从第 1 帧到 100 帧制作一个逐渐变长的进度条。方法如下：

按【Ctrl+F8】新建一个影片剪辑，命名为 loadingbar。

选择矩形绘画工具，将填充色设置为线性渐变色，笔触颜色设置为黑色。在 loadingbar 的第 1 帧画出一个细长的矩形。如图 1-4-5 所示。

按住【SHIFT】键把四个外边框同时选中，剪切。在原有图层上面新建图层，选中新图层的第 1 帧，点【编辑】>【粘贴到当前位置】（或者按【Ctrl+Shift+V】快捷键，将外边框复制到图层 2 中的原来的位置，这样外边框和矩形块就分离到两个图层中了。