



国家示范（骨干）高职院校
重点建设专业优质核心课程系列教材

Flash AS 3.0

动画程序设计案例教程

余以胜 编著



中国水利水电出版社
www.waterpub.com.cn

国家示范（骨干）高职院校重点建设专业优质核心课程系列教材

Flash AS 3.0 动画程序设计案例教程

余以胜 编著



中国水利水电出版社
www.waterpub.com.cn

内 容 提 要

Flash ActionScript (AS) 3.0 是一种功能强大的面向对象编程语言, 在网络游戏、网页特效、互动广告等领域有广泛的应用。本书以 AS 3.0 语言为基础, 以 Adobe Flash Professional CS5/CS5.5/CS6 为平台, 从程序设计的角度介绍各种常用的 Flash 动画编程技能。

本书从动画作品编程的实际需求出发, 将系统化的理论知识与精心挑选的编程案例结合起来组织内容, 注重理论教学与动手实践的融会贯通, 从而实施高职教育所倡导的“教、学、做一体化”的教学模式。

本书知识系统全面、案例丰富实用, 适合作为高职院校动漫设计与制作、多媒体技术、计算机应用技术等信息类专业的教材, 也适合作为培训机构用书, 还适合作为广大影视、游戏动画从业人员的参考书和具备 Flash 动画制作基础、程序设计基础的动画设计爱好者的自学用书。

本书提供源代码、素材、电子课件等教学资源, 可以从中国水利水电出版社网站和万水书苑免费下载, 网址为: <http://www.waterpub.com.cn/softdown/> 和 <http://www.wsbookshow.com>。

图书在版编目 (C I P) 数据

Flash AS 3.0 动画程序设计案例教程 / 余以胜编著

— 北京: 中国水利水电出版社, 2014. 7

国家示范 (骨干) 高职院校重点建设专业优质核心课程系列教材

ISBN 978-7-5170-2150-6

I. ①F… II. ①余… III. ①动画制作软件—高等职业教育—教材 IV. ①TP391.41

中国版本图书馆CIP数据核字 (2014) 第128922号

策划编辑: 石永峰 责任编辑: 李 炎 加工编辑: 袁 慧 封面设计: 李 佳

书 名	国家示范 (骨干) 高职院校重点建设专业优质核心课程系列教材 Flash AS 3.0 动画程序设计案例教程
作 者	余以胜 编著
出版发行	中国水利水电出版社 (北京市海淀区玉渊潭南路 1 号 D 座 100038) 网址: www.waterpub.com.cn E-mail: mchannel@263.net (万水) sales@waterpub.com.cn
经 售	电话: (010) 68367658 (发行部)、82562819 (万水) 北京科水图书销售中心 (零售) 电话: (010) 88383994、63202643、68545874 全国各地新华书店和相关出版物销售网点
排 版	北京万水电子信息有限公司
印 刷	北京市泽宇印刷有限公司
规 格	184mm×260mm 16 开本 16.5 印张 424 千字
版 次	2014 年 7 月第 1 版 2014 年 7 月第 1 次印刷
印 数	0001—3000 册
定 价	35.00 元

凡购买我社图书, 如有缺页、倒页、脱页的, 本社发行部负责调换
版权所有·侵权必究

前 言

动漫产业被称作 21 世纪知识经济的核心产业,是继 IT 业后又一经济增长点,涵盖了动画、漫画、游戏、制造业、版权业等诸多行业。近年来中国动漫产业的快速成长与高速发展,极大地促进了高校对动漫专业人才的招生和培养。

在国产动漫产品的数量大幅增长、质量及水平要求日益提高的前提下,将 ActionScript 动画编程的内容从“Flash 动画设计与制作”课程抽取出来,单独开设“Flash ActionScript 动画程序设计”课程,不仅是对动漫制作职业技能培养的重要补充,而且可以将“Flash 动画设计与制作”课程重新定位,使其更加专注作品本身的设计与制作技能培养。

ActionScript 3.0 实现了真正意义上的面向对象编程。目前关于 ActionScript 3.0 编程技术的书籍,基本上偏于理论;高校教材也只有少量的几本,要么知识不全面,要么案例不丰富,属于技术书籍附加少量片段代码的改良版,不适合作为高职教材使用。

本书是在编者多年讲授“Flash 动画设计与制作”课程,并一直研究和跟踪 ActionScript 编程技术等基础上完成的。依据近年来高职教育教学改革精神和职业教育教材建设的要求,以案例驱动的形式编写,突出 ActionScript 编程技能的培养,强调 Flash 作品设计和 ActionScript 编程的融合,注重实用职业技能的形成和通过案例驱动的训练方法达到教学目标。

本书的特色为案例驱动、工学结合。内容以实用案例为驱动,保证了理论教学与实践教学的一致性,适合“教、学、做一体化”教学;案例书写上更适合实践动手,包括案例任务描述、操作流程、案例小结等几部分,在每章后还有拓展训练,为本章知识巩固或编程技能的训练提高提供拓展空间。全书涉及 Flash ActionScript 3.0 编程的各个方面,知识体系完善,主要包括基本编程环境、AS 3.0 的基本语法、面向对象编程、事件侦听与用户交互、内置对象处理、几何结构处理、显示编程、声音和视频、AS 编程扩展、网络通信与数据处理等。针对各知识点,精心选择的案例完整、丰富、实用性极强,可以稍作修改即可使用,有的甚至不用修改即可应用到动画编程的职业工作实践中。本书的所有案例均在 Adobe Flash Professional CS5/CS5.5/CS6 环境下调试通过。

本书由余以胜编著,黄华林、巫志勇、宋广科、唐斌、王伟城、李龙万等老师为本书资源建设做了很多有益工作。在本书编写过程中参考了大量的计算机网络文献资料和现有技术书籍,在此,谨向这些著作者以及为本书付出辛勤劳动的同事们深表感谢!

由于编者水平有限,书中错误、疏漏或不妥之处在所难免,敬请读者和同仁给予批评指正。

编 者

2014 年 3 月

目 录

前言

第1章 动画编程简介.....1	2.4.2 操作流程.....20
1.1 动画编程语言 AS 3.0.....1	2.4.3 案例小结.....22
1.1.1 ActionScript 3.0 的优点.....2	2.5 程序结构与函数.....22
1.1.2 ActionScript 3.0 中的新功能.....2	2.5.1 条件语句.....22
1.2 Flash IDE 简介.....3	2.5.2 循环语句.....23
1.3 案例——图形自动绘制程序.....4	2.5.3 函数.....24
1.3.1 案例任务描述.....4	2.5.4 注释.....25
1.3.2 操作流程.....4	2.6 案例——数组处理程序.....26
1.3.3 案例小结.....5	2.6.1 案例任务描述.....26
1.4 案例——补间动画播放程序.....6	2.6.2 操作流程.....26
1.4.1 案例任务描述.....6	2.6.3 案例小结.....28
1.4.2 操作流程.....6	2.7 案例——求两数的最小公倍数程序.....29
1.4.3 案例小结.....8	2.7.1 案例任务描述.....29
1.5 案例——Hello World 程序.....8	2.7.2 操作流程.....29
1.5.1 案例任务描述.....8	2.7.3 案例小结.....31
1.5.2 操作流程.....9	拓展训练.....31
1.5.3 案例小结.....12	第3章 AS 3.0 中面向对象的编程.....32
拓展训练.....12	3.1 类和对象.....32
第2章 AS 3.0 基本语法.....13	3.1.1 包和命名空间.....33
2.1 常量与变量.....13	3.1.2 文档类.....33
2.1.1 常量.....13	3.1.3 元件连接类.....34
2.1.2 变量.....14	3.1.4 自定义类的编写.....34
2.2 数据与数据类型.....15	3.1.5 类的访问属性.....36
2.2.1 数据类型.....15	3.1.6 类成员的访问属性.....37
2.2.2 类型转换.....16	3.1.7 包外定义的类.....37
2.2.3 数组.....17	3.1.8 对象的创建与使用.....37
2.3 运算符与表达式.....17	3.2 案例——创建图形类程序.....38
2.3.1 运算符与表达式.....17	3.2.1 案例任务描述.....38
2.3.2 运算符的优先级和结合律.....17	3.2.2 操作流程.....38
2.3.3 主要运算符.....18	3.2.3 案例小结.....44
2.3.4 几个典型的一元运算符和关系运算符.....18	3.3 AS 3.0 中类的使用.....44
2.4 案例——类型转换程序.....19	3.4 数值计算处理.....46
2.4.1 案例任务描述.....19	3.4.1 边界情况.....46

3.4.2 Math 类	47	4.7 案例——鼠标引领甲壳虫程序	77
3.5 案例——小球随机运动程序	48	4.7.1 案例任务描述	77
3.5.1 案例任务描述	48	4.7.2 操作流程	77
3.5.2 操作流程	49	4.7.3 案例小结	80
3.5.3 案例小结	51	4.8 键盘事件输入响应	80
3.6 日期和时间处理	51	4.9 案例——键盘操控甲壳虫程序	81
3.6.1 处理 Date 对象	51	4.9.1 案例任务描述	81
3.6.2 获取时间单位值	52	4.9.2 操作流程	81
3.6.3 执行日期和时间运算	52	4.9.3 案例小结	84
3.6.4 控制时间间隔	52	4.10 案例——射击飞机小游戏程序	84
3.7 案例——日期时钟程序	53	4.10.1 案例任务描述	84
3.7.1 案例任务描述	53	4.10.2 操作流程	84
3.7.2 操作流程	54	4.10.3 案例小结	90
3.7.3 案例小结	55	拓展训练	90
3.8 案例——百分秒计时器程序	56	第5章 操作 AS 3.0 对象	91
3.8.1 案例任务描述	56	5.1 处理数组	91
3.8.2 操作流程	56	5.1.1 索引数组	91
3.8.3 案例小结	58	5.1.2 关联数组	94
拓展训练	59	5.2 案例——歌曲列表管理程序	96
第4章 事件侦听及用户交互	60	5.2.1 案例任务描述	96
4.1 事件及事件侦听	60	5.2.2 操作流程	96
4.1.1 事件	60	5.2.3 案例小结	101
4.1.2 事件处理过程	60	5.3 处理字符串	101
4.1.3 事件对象	61	5.4 案例——ASCII 字符图程序	103
4.1.4 事件侦听	63	5.4.1 案例任务描述	103
4.2 常用内置事件类	64	5.4.2 操作流程	103
4.2.1 Timer 类及相关事件	64	5.4.3 案例小结	108
4.2.2 TextField 类及相关事件	65	5.5 处理影片剪辑	108
4.2.3 ENTER_FRAME 事件	65	5.6 案例——外部 SWF 库中影片处理程序	110
4.3 案例——闹钟程序	66	5.6.1 案例任务描述	110
4.3.1 案例任务描述	66	5.6.2 操作流程	110
4.3.2 操作流程	66	5.6.3 案例小结	113
4.3.3 案例小结	71	5.7 处理文本	113
4.4 鼠标事件及响应	71	5.8 案例——报刊风格文本设置程序	115
4.5 案例——击拍豆腐小游戏程序	72	5.8.1 案例任务描述	115
4.5.1 案例任务描述	72	5.8.2 操作流程	115
4.5.2 操作流程	72	5.8.3 案例小结	123
4.5.3 案例小结	76	5.9 处理错误	124
4.6 鼠标拖曳移动物体	76	5.10 案例——定制错误应用程序	125

5.10.1 案例任务描述	125	7.4 图形绘制 API	164
5.10.2 操作流程	125	7.5 案例——自选图形绘制程序	167
5.10.3 案例小结	129	7.5.1 案例任务描述	167
拓展训练	129	7.5.2 操作流程	167
第6章 处理几何结构	130	7.5.3 案例小结	169
6.1 舞台及坐标系统	130	7.6 处理位图	169
6.2 角与角度、旋转	132	7.7 案例——拼图小游戏程序	171
6.3 案例——舞台尺寸动态展示程序	132	7.7.1 案例任务描述	171
6.3.1 案例任务描述	132	7.7.2 操作流程	172
6.3.2 操作流程	133	7.7.3 案例小结	175
6.3.3 案例小结	135	7.8 颜色及其变换	176
6.4 案例——冰球位置动态展示程序	135	7.9 案例——图片颜色变换小程序	176
6.4.1 案例任务描述	135	7.9.1 案例任务描述	176
6.4.2 操作流程	136	7.9.2 操作流程	177
6.4.3 案例小结	138	7.9.3 案例小结	178
6.5 Point 类的点对象	138	7.10 过滤显示对象	178
6.6 Rectangle 类的矩形对象	140	7.11 案例——图片过滤处理程序	179
6.7 Matrix 类的转换矩阵	142	7.11.1 案例任务描述	179
6.8 案例——矩阵变换处理图片程序	143	7.11.2 操作流程	180
6.8.1 案例任务描述	143	7.11.3 案例小结	182
6.8.2 操作流程	144	拓展训练	182
6.8.3 案例小结	148	第8章 声音和视频处理	183
拓展训练	148	8.1 声音处理	183
第7章 显示编程	149	8.1.1 处理声音简介	183
7.1 显示列表	149	8.1.2 加载外部声音文件	184
7.1.1 显示列表结构	149	8.1.3 处理嵌入的声音	185
7.1.2 stage、root、this、parent	150	8.1.4 处理声音流文件	185
7.1.3 核心显示类	150	8.1.5 播放声音	186
7.2 处理显示对象	152	8.1.6 处理声音元数据	188
7.2.1 在显示列表中添加显示对象	152	8.1.7 访问原始声音数据	188
7.2.2 处理显示对象容器	152	8.2 捕获麦克风输入	188
7.2.3 遍历显示列表	153	8.2.1 访问麦克风	189
7.2.4 处理显示对象的事件	154	8.2.2 将麦克风音频传送到本地扬声器	189
7.2.5 适当选择 DisplayObject 子类	154	8.2.3 更改麦克风音频	189
7.2.6 显示对象的舞台表现	155	8.2.4 检测麦克风活动	189
7.3 案例——图形堆叠处理程序	158	8.3 案例——mp3 播放器程序	190
7.3.1 案例任务描述	158	8.3.1 案例任务描述	190
7.3.2 操作流程	159	8.3.2 操作流程	190
7.3.3 案例小结	164	8.3.3 案例小结	195

8.4 案例——声音波纹显示程序	195	9.6.1 案例任务描述	220
8.4.1 案例任务描述	195	9.6.2 操作流程	221
8.4.2 操作流程	195	9.6.3 案例小结	225
8.4.3 案例小结	197	9.7 字节数组	225
8.5 视频处理	197	9.8 案例——ZIP 文件处理程序	227
8.5.1 Video 类	198	9.8.1 案例任务描述	227
8.5.2 加载视频文件	198	9.8.2 操作流程	227
8.5.3 控制视频回放	198	9.8.3 案例小结	229
8.5.4 流式传输视频文件	199	9.9 外部 API	230
8.5.5 提示点	199	9.9.1 使用外部 API 简介	230
8.5.6 使用视频元数据	200	9.9.2 使用 ExternalInterface 类	230
8.6 捕获摄像头输入	200	9.10 案例——与 JavaScript 通信小程序	232
8.6.1 Camera 类简介	200	9.10.1 案例任务描述	232
8.6.2 在屏幕上显示摄像头内容	201	9.10.2 操作流程	233
8.6.3 设计摄像头应用程序	201	9.10.3 案例小结	234
8.6.4 连接到用户摄像头	201	拓展训练	235
8.6.5 验证是否已安装摄像头	202	第 10 章 网络及数据处理	236
8.6.6 检测摄像头的访问权限	202	10.1 网络通信	236
8.6.7 监视回放条件	203	10.1.1 网络与通信简介	236
8.7 案例——FLV 视频自动播放程序	203	10.1.2 外部数据处理	237
8.7.1 案例任务描述	203	10.1.3 与其他 Flash Player 实例连接	237
8.7.2 操作流程	203	10.1.4 套接字处理	238
8.7.3 案例小结	207	10.1.5 本地数据存储	239
拓展训练	207	10.1.6 文件的上传与下载	239
第 9 章 AS 编程扩展	208	10.2 案例——文件的上传与下载程序	240
9.1 使用组件	208	10.2.1 案例任务描述	240
9.1.1 组件类型	209	10.2.2 操作流程	241
9.1.2 组件的使用	209	10.2.3 案例小结	244
9.2 案例——使用组件播放视频程序	210	10.3 处理 XML	245
9.2.1 案例任务描述	210	10.3.1 XML 对象和 XMLList 对象	245
9.2.2 操作流程	210	10.3.2 XML 遍历	246
9.2.3 案例小结	211	10.3.3 XML 类型转换	246
9.3 上下文菜单	212	10.3.4 读取外部 XML 文档	247
9.4 案例——弹出式菜单操作程序	213	10.4 案例——从网络加载 RSS 数据程序	247
9.4.1 案例任务描述	213	10.4.1 案例任务描述	247
9.4.2 操作流程	213	10.4.2 操作流程	247
9.4.3 案例小结	216	10.4.3 案例小结	250
9.5 正则表达式	216	10.5 客户端系统环境	250
9.6 案例——用户信息校验程序	220	10.5.1 System 类	250

10.5.2	Capabilities 类	251
10.5.3	使用 IME 类	251
10.6	案例——检测系统功能程序	252
10.6.1	案例任务描述	252

10.6.2	操作流程	252
10.6.3	案例小结	254
	拓展训练	255
	参考文献	256

动画编程简介



本章导读

本章将对 ActionScript 3.0 作简单介绍, 并对 Adobe Flash Professional 中的 IDE 编程环境作简要说明, 这些初步的认识和了解, 为以后进一步学习 ActionScript 3.0 语言和训练动画编程技能打下了基础。



本章要点

- Flash ActionScript 3.0 概况
- Adobe Flash Professional IDE
- Flash 中设计与 AS 代码的集成

1.1 动画编程语言 AS 3.0

Flash 是目前最为热门的二维交互式矢量动画制作软件。ActionScript (AS) 是内置于 Flash 的编程语言, 它在 Flash 内容和应用程序中实现交互性、数据处理以及其他许多功能。

ActionScript 在 ActionScript 虚拟机 (AVM) 中执行, 运行时 AVM 包含在 Flash Player 中。ActionScript 代码通常由编译器转换为字节代码格式。字节代码是一种由计算机编写和识别的编程语言。在 Adobe Flash Professional 中内置有编译器。字节代码嵌入在 Flash Player 执行的 SWF 文件中。

ActionScript 3.0 已成了真正的面向对象的编程语言, 提供了可靠的编程模型, 方便创建拥有大型数据集和面向对象的可重用代码库的高度复杂应用程序。ActionScript 3.0 相对于早期版本改进的一些重要功能包括:

- 一个新增的 ActionScript 虚拟机, 称为 AVM2, 它使用全新的字节代码指令集, 可使性能显著提高;

- 一个更为先进的编译器代码库，可执行比早期编译器版本更深入的优化；
- 一个扩展并改进的应用程序编程接口（API），拥有对对象的低级控制和真正意义上的面向对象模型；
- 一个基于 ECMAScript for XML（E4X）规范的 XML API。E4X 是 ECMAScript 的一种语言扩展，它将 XML 添加为语言的本机数据类型；
- 一个基于文档对象模型（DOM）第 3 级事件规范的事件模型。

1.1.1 ActionScript 3.0 的优点

ActionScript 3.0 的脚本编写功能优于早期版本，它旨在方便创建拥有大型数据集和面向对象的可重用代码库的高度复杂应用程序。在 Flash Player 中运行的内容不要求使用 ActionScript 3.0。但是，拥有它可以得到只能通过 AVM2（ActionScript 3.0 虚拟机）实现的性能改善。与旧的 ActionScript 代码相比，ActionScript 3.0 代码的执行速度快 10 倍。

早期版本的 ActionScript 虚拟机 AVM1 执行 ActionScript 1.0 和 ActionScript 2.0 代码。Flash Player 9、10、11 等版本也支持 AVM1 以实现向后兼容性。

1.1.2 ActionScript 3.0 中的新功能

ActionScript 3.0 的新功能包括新增的核心语言功能，以及能够更好地控制低级对象的改进 API。

1. 核心语言功能

核心语言定义编程语言的基本构造块，例如语句、表达式、条件、循环和类型。ActionScript 3.0 包含许多加快开发过程的功能。

①运行时异常：运行时异常用于常见的错误情形；②运行时类型：类型信息在运行时保留，这样提高了性能，减少了内存使用量；③密封类：密封类只能拥有在编译时定义的一组固定的属性和方法；不能添加其他属性和方法；④闭包方法：ActionScript 3.0 使闭包方法可以自动记起它的原始对象实例；⑤ECMAScript for XML（E4X）：E4X 通过大大减少所需代码的数量来简化操作 XML 的应用程序的开发；⑥正则表达式：ActionScript 3.0 包括对正则表达式的固有支持，因此可以快速搜索并操作字符串；⑦命名空间：命名空间使用统一资源标识符（URI）以避免冲突，而且在使用 E4X 时还可以表示 XML 命名空间；⑧新基元类型：ActionScript 3.0 包含三种数值类型：Number、int 和 uint。而 ActionScript 2.0 只包含 Number 一种数值类型。

2. API 功能

ActionScript 3.0 中的 API 包含许多可用于在低级别控制对象的类。语言体系结构的设计比早期版本更为直观。虽然有太多的类需要详细介绍，但是一些重要的区别更值得注意。

①DOM3 事件模型：这种事件模型的设计允许应用程序中的对象进行交互和通信、维持其状态以及响应更改，ActionScript 3.0 事件模型的模式遵守 W3C 的 DOM 级别 3 事件规范；②显示列表 API：用于访问显示列表（包含应用程序中所有可视元素的树）的 API 由使用可视基元的类组成；③处理动态数据和内容：ActionScript 3.0 包含用于加载和处理应用程序中的资源和数据的机制，这些机制在 API 中是直观的并且是一致的；④低级数据访问：多种 API 都提供对数据的低级访问；⑤使用文本：ActionScript 3.0 包含一个用于所有与文本相关的 API 的 flash.text 包。

为了对文本进行更低级别的控制，flash.text.engine 包中的类组成了 Flash 文本引擎。这组类提供对文本的低级控制，是针对创建文本框架和组件而设计的。

1.2 Flash IDE 简介

Adobe Flash Professional IDE, 从 CS3 开始直到现在的 CS6 版本, 一直没发生过改变, 是一种集 Flash 动画设计与编程于一体的面向对象集成开发环境。

1. “动作”面板

“动作”面板是用来编辑、调试时间轴代码、类代码等所有 AS3.0 程序代码的场所。“动作”面板分为 4 个区域, 分别是脚本窗口、面板菜单、动作工具箱、脚本导航器, 如图 1-1 所示。

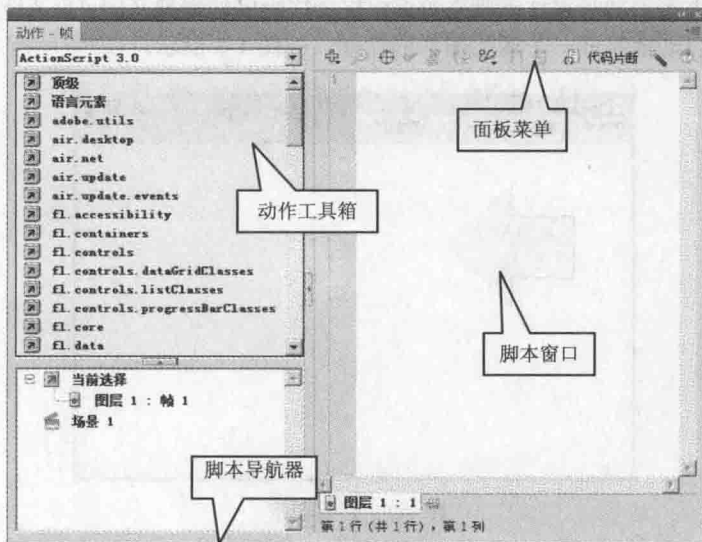


图 1-1 “动作”面板

2. “输出”面板

“输出”面板是测试程序的有效工具。程序中调用 `trace()` 函数输出相关信息到“输出”面板, 它在显示结果、测试代码的过程中经常用到。通过鼠标点击菜单【窗口】→【输出】可以将“输出”面板打开。

3. “编译器错误”面板

“编译器错误”面板是调试程序时经常使用的有效工具。通过查看面板中的错误, 可以清晰地定位程序出错位置及错误的描述, 有助于程序调试中将错误定位和改正, 如图 1-2 所示。与“输出”面板类似, 可以通过鼠标点击菜单【窗口】→【编译器错误】将“编译器错误”面板打开。

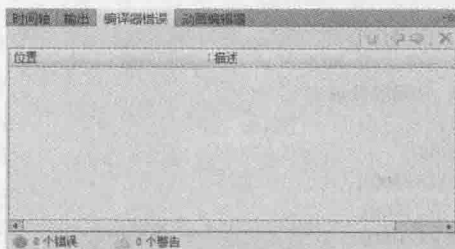


图 1-2 “编译器错误”面板

4. “脚本”窗口

“脚本”窗口是代码编辑的区域。“动作”面板是编辑时间轴代码的主要工具，更多的代码在外部的脚本文件 (*.as) 中保存。

1.3 案例——图形自动绘制程序

1.3.1 案例任务描述

本案例是一个不需要动画设计的脚本范例，程序代码以外部独立的 AS 文件存在，它实现的功能是在程序运行时自动地完成一个箭头图形的绘制，如图 1-3 所示。

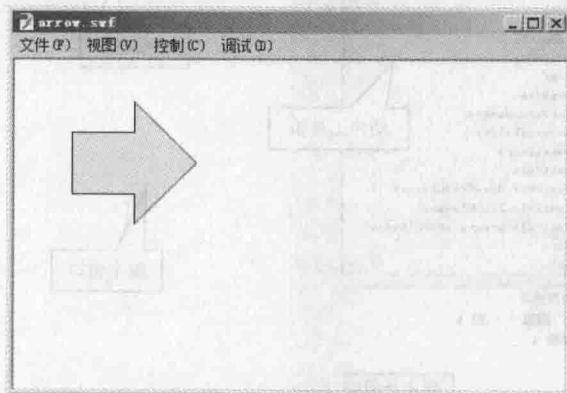


图 1-3 箭头图形

1.3.2 操作流程

(1) 创建并编辑脚本文件：打开 Adobe Flash Professional CS5/CS5.5/CS6，选择【文件】→【新建(N)...】，打开“新建文档”窗口，选择“ActionScript 文件”类型，点击【确定】按钮，创建一个空的脚本文件。在脚本窗口输入下述代码，文件命名为 Arrow.as 保存到指定的文件夹。注意，文件名与代码中的类名要一致，区分大小写。

```
package { //包定义:首先本程序需用到的内建类
    import flash.display.Sprite;
    import flash.display.Graphics;

    public class Arrow extends Sprite{ //自定义 Arrow 类
        public function Arrow() {
            init();
        }
        public function init():void { //画线并填充
            graphics.lineStyle(1,0,1);
            graphics.beginFill(0xffff00);
            graphics.moveTo(0+100,25+100);
            graphics.moveTo(0+100,25+100);
            graphics.lineTo(50+100,25+100);
            graphics.lineTo(50+100,0+100);
            graphics.lineTo(100+100,50+100);
        }
    }
}
```

```
graphics.lineTo(50+100,100+100);  
graphics.lineTo(50+100,75+100);  
graphics.lineTo(0+100,75+100);  
graphics.lineTo(0+100,25+100);  
graphics.endFill();  
}  
}
```

(2) 创建空的 Fla 文档: 打开 Adobe Flash Professional CS5/CS5.5/CS6, 创建新文档, 选择“ActionScript”类型, 创新空文档, 进入设计环境。点击【文件】→【ActionScript 设置...】, 打开“高级 ActionScript 3.0 设置”对话框, 如图 1-4 所示。在“文档类”栏中输入名称: Arrow, 并在“源路径”下填加保存 Arrow.as 的文件夹。

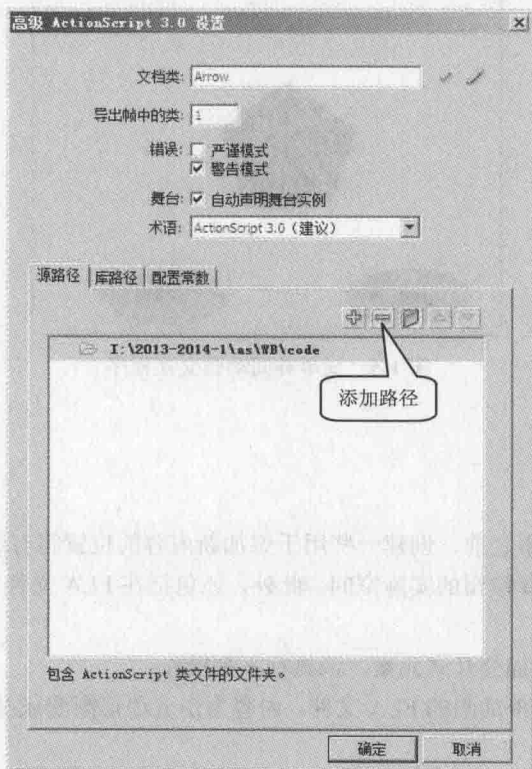


图 1-4 高级 ActionScript 设置

(3) 设置完成后, 文件命名为 Arrow fla 保存到与 Arrow.as 相同的文件夹下, 按 Ctrl+Enter 组合键测试影片, 可以看到绘制的箭头效果。

1.3.3 案例小结

在本案例中, 使用独立存在的 Arrow.as 脚本文件, 被另一个设计内容空白的 Arrow fla 设计文件通过文档类的方式调用, 集成在一起, 形成最后的 Flash 作品。

Arrow.as 文件的类代码可以被重复使用, 任何其他影片作品文档都可调用它, 原因在于它是独立存在的。

1.4 案例——补间动画播放程序

1.4.1 案例任务描述

本案例首先设计一段常规的形状补间动画，然后添加动画中的交互行为，包括两个可以单击的按钮：一个用于启动动画，另一个用于导航到单独的 URL（某个网站，比如中国水利水电出版社）。交互行为的动作代码为存放在时间轴上的帧代码。运行效果如图 1-5 所示。

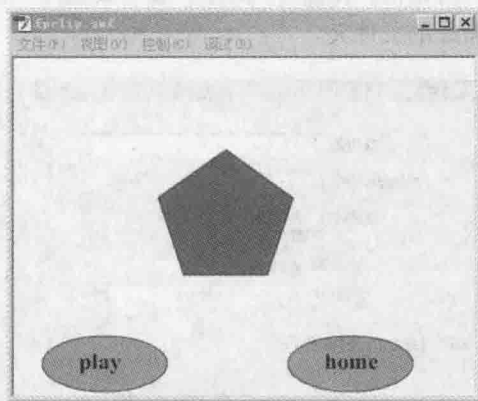


图 1-5 简单补间动画交互程序

1.4.2 操作流程

1. 准备添加交互

向动画添加交互式元素之前，创建一些用于添加新内容的位置将有助于创建 FLA 文件。此任务包括在舞台上创建可放置按钮的实际空间。此外，还包括在 FLA 文件中创建“空间”，以分隔不同的项目。

要创建 FLA 文件以添加交互式元素，请执行下列操作：

(1) 创建一个包含简单动画的 FLA 文件，内容为由五边形渐变成圆形的形状补间动画，文件名可为 `fpclip.fl`。

(2) 确定两个按钮在屏幕上的显示位置。一个按钮用于启动动画，另一个按钮用于链接到作者的包或主页。如果需要，在舞台上为新内容清除或添加一些空间。如果该动画还没有启动按钮，可以在第一帧上创建一个启动屏幕。这时，可能希望移动动画，使其从第二帧或后面的帧启动播放。

(3) 在时间轴中的图层上添加一个新图层，并将其命名为 `buttons`。这是将要在其中添加按钮的图层。

(4) 在 `buttons` 图层之上创建一个新图层，并将其命名为 `actions`。这是将要在其中向应用程序添加 ActionScript 代码的图层。最后完成的时间轴，如图 1-6 所示。

2. 创建和添加按钮

接下来，将实际创建和放置构成交互式应用程序中心的按钮。要创建按钮并将其添加到 FLA 文件，请执行下列操作：

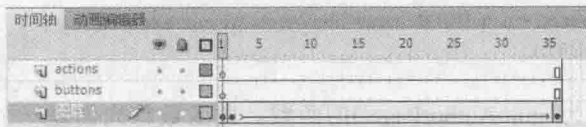


图 1-6 简单补间动画的时间轴

(5) 使用绘图工具在 **buttons** 图层上创建第一个按钮 (“play” 按钮) 的可视外观, 例如, 绘制一个顶部带文本的水平椭圆。

(6) 使用 “选择” 工具选择单个按钮的所有图形部分。

(7) 在主菜单中, 选择【修改】→【转换为元件】。

(8) 在对话框中, 选择 “按钮” 作为元件类型, 将该元件命名为 **play**, 单击 “确定” 按钮。

(9) 在 “库” 中双击 **play** 按钮, 对 **play** 按钮的 “指针”、“按下” 等帧加入关键帧, 更换不同的颜色。

(10) 返回场景后, 选择按钮, 在属性检查器中为按钮的实例命名为 **playButton**。

(11) 重复步骤 (5) 至 (10), 创建可将查看者链接至某个网站的按钮 **home**。将该按钮的实例命名为 **homeButton**。

3. 编写代码

虽然此应用程序的 **ActionScript** 代码全都是在同一个位置输入的, 但是功能可分为三组。代码的这三组功能分别是:

- 一旦 **SWF** 文件开始加载 (当播放头进入第 1 帧时), 就停止播放头。
- 侦听一个事件, 该事件在用户单击 **play** 按钮时开始播放 **SWF** 文件。
- 侦听一个事件, 该事件在用户单击 **home** 按钮时将浏览器定向至相应的 **URL**。

要创建代码, 使得在播放头进入第 1 帧时停止播放头, 请执行下列操作:

(12) 在 **actions** 图层的第 1 帧上选择关键帧。

(13) 若要打开 “动作” 面板, 请从主菜单中选择【窗口】→【动作】。

(14) 在 “脚本” 窗口中, 输入以下代码:

```
stop();
```

接着编写程序代码, 使得单击 **play** 按钮时启动动画播放, 请执行下列操作:

(15) 在前面步骤中输入的代码的末尾添加两个空行。

(16) 在脚本底部输入以下代码:

```
function startMovie(event:MouseEvent):void {
    this.play();
}
```

此代码定义一个名为 **startMovie()** 的函数, 调用该函数会导致主时间轴开始播放。

(17) 接下来, 输入以下代码行:

```
playButton.addEventListener(MouseEvent.CLICK, startMovie);
```

此代码行将 **startMovie()** 函数注册为 **playButton** 的 **CLICK** 事件的侦听器。也就是说, 只要单击名为 **playButton** 的按钮, 就会调用 **startMovie()** 函数。

接着编写代码, 使得单击 **home** 按钮时将浏览器定向至相应的 **URL**, 请执行下列操作:

(18) 在前面步骤中输入的代码的末尾添加两个空行。

(19) 在脚本底部输入以下代码:

```
function gotoAuthorPage(event:MouseEvent):void {
```

```
var targetURL:URLRequest = new URLRequest("http://www.waterpub.com.cn");
navigateToURL(targetURL);
}
```

此代码定义一个名为 gotoAuthorPage() 的函数。此函数首先创建一个表示 URL http://www.hxedu.com.cn 的 URLRequest 实例，然后将该 URL 传递给 navigateToURL() 函数，使用户的浏览器打开此 URL。

(20) 在上一步中添加的代码的下一行中，输入以下代码：

```
homeButton.addEventListener(MouseEvent.CLICK, gotoAuthorPage);
```

此代码将 gotoAuthorPage() 函数注册为 homeButton 的 CLICK 事件的侦听器。也就是说，只要单击名为 homeButton 的按钮，就会调用 gotoAuthorPage() 函数。

(21) 随时记得保存文件。至此，再对 fpclip fla 最终源文档保存。

4. 测试应用程序

应用程序的功能已实现，测试一下应用程序，看是否达到预期效果。测试应用程序：

(22) 从主菜单中选择【控制】→【测试影片】。Flash Professional 将创建 SWF 文件，并在 Flash Player 窗口中打开该文件。

(23) 试用这两个按钮，确保它们按预期的方式工作。

(24) 如果按钮不起作用，请检查下列事项：

- 这两个按钮是否具有不同的实例名？
- addEventListener() 方法调用使用的名称是否与按钮的实例名相同？
- addEventListener() 方法调用中使用的事件名称是否正确？
- 为各个函数指定的参数是否正确？

上述错误和大多数其他错误都会导致出现错误消息。选择【测试影片】命令时，或测试项目时单击此按钮，都会出现错误消息。在“编译器错误”面板中查看编译器错误（第一次选择【测试影片】时出现的错误）。检查“输出”面板看看是否有在播放内容（如单击按钮）时发生的运行时（runtime）错误。

1.4.3 案例小结

本案例将各段 ActionScript 代码合并后形成一个完整的应用程序，实现如何利用现有线性动画以及添加某些交互式元素。

程序代码全部放置在时间轴的帧上，不存在或没有用到外部 AS 文件中的代码。

1.5 案例——Hello World 程序

1.5.1 案例任务描述

本案例 ActionScript 应用程序是一个标准“Hello World”应用程序，设计要求较为简单：应用程序名为 HelloWorld；将显示一个包含“Hello World!”字样的文本字段；应用程序主要实现一个名为 Greeter 的面向对象的类，存在于外部 AS 文件中。这种设计允许在 Flash Professional 中使用该类。

首先创建该应用程序的基本版本，即标准输出“Hello World!”字符串。然后添加功能，使用