Flash脚本基础实例

FLASH脚本特效

- 常用脚本特效欣赏

[萤火虫]

[雪花]

[弹性跟随]

[网页导航条]

[黑客帝国数字流]

学习内容

- 脚本基础知识
- 按钮元件的使用
- 导航菜单的制作
- 控制影片剪辑的脚本

FLASH脚本_AS

- Action Script是Flash的脚本语言，具有强大的交互功能，通过脚本应用，用户对动画元件的控制得到加强。目前提供了AS2.0和AS3.0两个版本。

- 动作面板是Flash提供的运行编程的专用环境——F9打开动作面板；

- 使用ActionScript的目的：
  - 交互式网站的开发；
  - 课件制作；
  - 小游戏开发、MTV、电子贺卡制作；

AS可添加在哪些对象上(AS2.0)：

- 帧——Frame必须是关键帧

写在关键帧上面的AS，当时间轴上的指针走到这个关键帧的时候，就写在这个帧上面的AS就被触发执行了。

操作方法：点选关键帧，然后打开AS面板，写入语句；或新建一层插入空白关键帧，然后打开AS面板，写入语句。

- 按钮——Button

- 影片剪辑——Movie Clip

AS2和AS3的主要区别

在AS2中，ActionScript代码可以添加到关键帧（Keyframe）、按钮（Button）或影片剪辑（Movie Clips）中，并分别称之为帧动作、按钮动作及影片剪辑动作。

在AS3中，ActionScript代码只能添加到关键帧（Keyframe），是面向对象的编程语言，而不是一个简简单单的控制影片播放的东西. 某种角度上说,随着FLASH的功能变的越来越强大,AS变的越来越复杂几乎是不可避免的。

AS基础知识

- Action Script的相关术语
- Action Script的语法
- 脚本的控制结构

Action Script相关术语

- 事件：起触发作用的事情。如鼠标的移动、按下或释放等；

  如:按钮常见事件： release、 dragOut、rollOut

- 关键字：AS中有16个关键字

  break continue delete else for function if in new return this tupeof var void while with

- 帧标签:即关键帧的名字(在时间轴上显示小红旗);

- 元件实例名称: “元件”从【库】中进入“舞台”就被称为该“元件”的“实例”；有名称的影片剪辑实例才能在动作面板中调用。

Action Script的语法

- 点语法：a1.gotoAndPlay(2);
- 括号：定义函数中的相关参数；
- 大括号：{ }形成一个完整的语句块；
- 分号：每条语句以;结束；
- 注释：//后可跟上注释；
- 字母大小写:关键字要区分大小写；

脚本的控制结构

- Flash的脚本控制结构有三种：**顺序结构、分支结构、循环结构。**

一、分支结构

1、单分支

格式:

```
if (条件){
  要执行的代码;
  }
```

功能:如果条件满足，执行相应语句;如果条件不满足,程序继续。

2、双分支

格式:

```
if (条件)
        {代码 1 }//条件满足执行这里的代码 1
else
        {代码 2 } //条件不满足执行这里的代码 2
```

功能:如果条件满足，执行相应语句1;如果条件不满足,
执行相应语句2;

3、多分支

格式： switch(变量){

```
  case 值1:语句1;break;
  case 值2:语句2;break;
  case 值3:语句3;break;
  default:语句4
}
```

二、循环结构

- for(初值；条件；下一个)

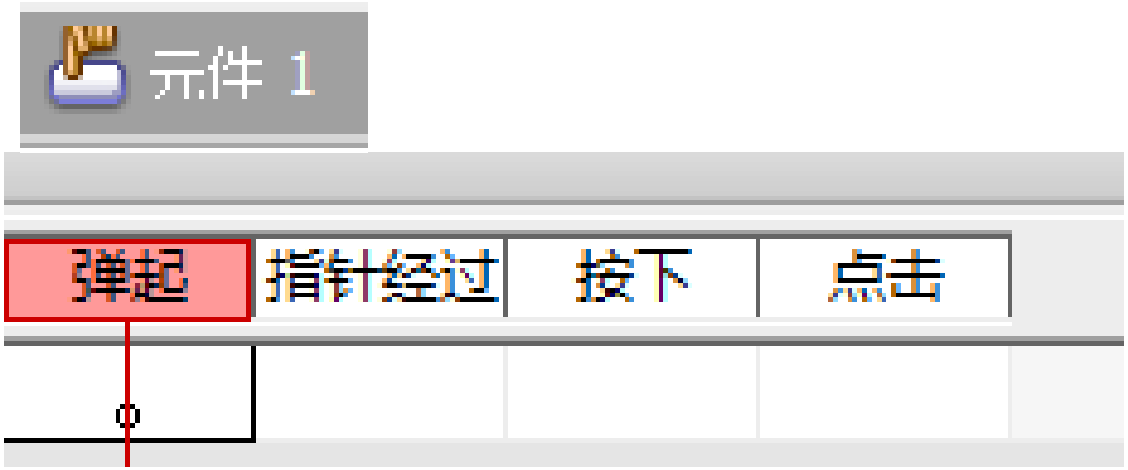
  { 循环体语句} //执行这里的代码N次

- while(条件)

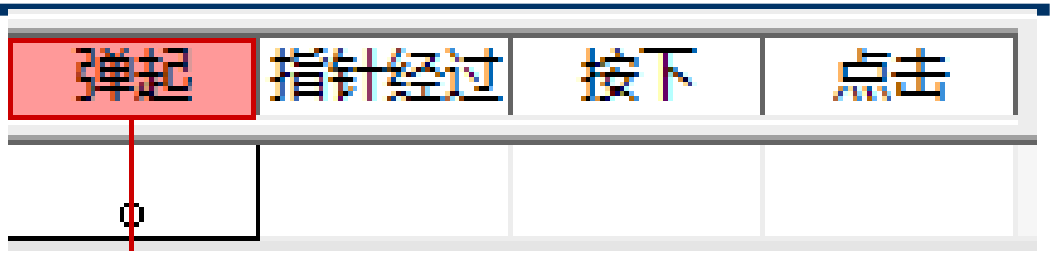
  {循环体语句} //当条件满足时一直执行这里的代码

按钮元件的使用

- 按钮的定义：
  - 按钮实际上是包含四帧的交互式影片剪辑
  - 当创建按钮元件时,Flash就会自动创建包括四帧的时间轴.



按钮的状态



- “弹起”当鼠标指针不接触按钮时，按钮处于弹起状态；
- “指针经过”当鼠标移动到按钮上面，但没有按下时，按钮所处的状态；
- “按下”当鼠标左键按下时，按钮所处的状态. 如果鼠标右键按下时，将会弹出关联菜单；
- “点击”在该状态下可以定义响应鼠标的区域，此区域在影片中是不可见的；

1、按钮制作实例

- 效果一
- 效果二

给按钮添加动作

按钮动作是按钮的灵魂，若不给按钮添加动作，按钮就毫无用处。使用给按钮添加动作的语法是：

```
On(Event){     //执行的动作   }
```

其中Event（事件）是指鼠标的各种动作，主要有：

- Press（点击）
- Release（释放）
- ReleaseOutside（释放离开）
- RollOver（指针经过）
- RollOut（指针离开）
- DragOver（拖放经过）

给按钮加动作

- 给按钮加动作
  - 实例：播放、重播、停止按钮
    - 从公用库中导入按钮；
    - 回到场景中，拖入按钮，给按钮实例加动作
    - on (release) {gotoAndplay(1);//重播
    - }

    - 给按钮加动作脚本,必须添加事件on

```
on(release)
        {stop( ); }
```

2、简单图片浏览器制作

- Flash中制作按钮元件,可任意发挥;新建图层“按钮”,放置制作好的按钮(如上一页、下一页);
- 新建图层”图片”,导入四幅图片(4个关键帧);并将此层拖入最底层;其它图层延续到第4帧;
- 在“图片”上又新建图层,命名为”图片序号”,画白色无边线矩形做为背景;左边是一个动态文本,将其变量名设为” b”,中间画一条斜线,右边是一个静态文本”4”;

2、简单图片浏览器制作

- 在最上层新建图层,命名为”脚本”;在第一帧添加如下脚本:

```
stop();
b=“1”;
```

- 上一页按钮添加如下脚本:

```
on(release){
        prevFrame();
        if(b>1){b--;} }
```

- 下一页按钮添加如下脚本:

```
on(release){
     nextFrame(); if(b<4){b++;} }
```

帧动作

- gotoAndPlay：从当前帧转到目标帧开始播放动画。
- gotoAndStop：从当前帧转到目标帧并停止播放动画。
- play：开始播放已停止了的动画。
- stop：从该帧停止当前正在播放的动画。
- nextFrame：从当前帧转到下一帧。
- nextScene：从当前场景转到下一场景。
- prevFrame：从当前帧转到前一帧。
- prevScene：从当前场景转到前一场景。
- stopAllSounds：停止正在播放的所有声音。

帧动作

帧添加动作的代码，是在时间轴的关键帧上。

【方法】

在时间轴上选择要添加动作的关键帧，然后打开动作面板，选择动作代码即可 。

例子：在动画时间轴上添加stop动作。

第一步：制作运动动画

第二步：演示动画

第三步：在时间轴末尾关键帧添加stop动作

第四步：演示动画

请同学们比较分析两次演示动画的不同

goto语句详解

- 语句包括:gotoAndPlay(scene,frame)
  gotoAndStop(scene,frame)
  nextFrame( )　prevFrame( )
- 通过时间跳转到某一位置播放，即播放到该帧时直接跳转到某一指定帧；
- 通过交互进行跳转，即当有一个事件发生后，才跳转，所以此类跳转必须写在相关事件内，on内，此on又必须添加到某对象内；
- goto语句可以添加在主时间轴，影片剪辑，按钮中；

- 例：

  gotoAndPlay（5）

  由目前播放的帧直接跳到同一场景内第5帧，由第5帧继续播放影片。

  gotoAndStop（5）

  由目前播放的帧直接跳到同一场景内第5帧，停止播放影片。

  gotoAndStop（“场景2”,5）

  由目前播放的帧直接跳到Scene2场景内的第5帧继续播放影片。

on详解

- 作用：给按钮加动作时一定要包含在on命令的大括号中
- on后跟的事件包括：
  - press
  - release
  - releaseOutside
  - rollOver
  - rollOut
  - dragOver
  - dragOut
  - keyPress”<Left>”
  - 此处Left还可以更改为Right Home End Delete Insert Enter Backspace Up Down PageUp PageDown Tab Escape Space

- press——点击--鼠标指针在按钮上时按下鼠标键；
- release——释放--鼠标指针在按钮上时，释放鼠标按键；
- rollOver——指针经过--鼠标指针移到按钮上面；
- rollOut——指针离开--鼠标指针从按钮上移出；
- releaseOutside——释放离开--鼠标指针在按钮上时按下鼠标按键，移出按钮外后才释放按键；
- dragOut——拖放离开--鼠标指针在按钮上时按下鼠标键，然后拖出按钮外；
- dragOver——拖放经过--鼠标指针在按钮上时按下鼠标键，然后拖出按钮外，接着又拖回按钮上；
- keyPress——按键--按下指定的键盘键

loadMovie详解(幻灯浏览)

- 作用:将SWF、JPEG、GIF或PNG从URL加载到影片剪辑中;
- 语法:loadMovie(url,目标,方法)
- 加载的对象与源文件最好在同一目录下
- 例:个人主页中导航按钮中用到了加载SWF;

- 实例——01在影片剪辑中加载图片.fla
  - 制作一空影片剪辑元件，拖入场景图层1的第1帧中，实例名称取为mc
  - 在第1帧加脚本 stop(); loadMovie("001.jpg",mc); 后四帧添加关键帧加相同脚本，更换图片名称
  - 新建图层2，添加两个按钮，控制上一帧，下一帧的跳转（略）

3、强化练习——幻灯片效果

- 目的：制作图片加载过程中淡入淡出的效果；
- 制作一空影片剪辑元件”载入图片”，拖入场景图层1的第1帧中，实例名称取为 mc
- 在第1帧加脚本loadMovie(“\\校园风光\\1.jpg”,mc); 分别在第15、25、35帧中添加关键帧，再将第1帧和第35帧中的实例的透明度改为0%，第1-15帧，第25-35帧之间加动作补间；
- 其它三幅图片载入效果类同，做在同一层 ；
- 新建图层2，添加四个按钮，控制五幅图片的显示，每个按钮上的动作如下（每个按钮对应一行脚本）；

```
on (release) {gotoAndPlay(1);}
on (release) {gotoAndPlay(36);}
on (release) {gotoAndPlay(71);}
on (release) {gotoAndPlay(106);}
```

loadMovieNum详解

- 作用:将图片、SWF从URL加载到层中;
- 例___01加载图片到层中.fla
  - 新建文档，保存后，在第1帧加如下代码（对应的图片保存在同目录下；）

```
loadMovieNum("001.jpg",1);
loadMovieNum("002.jpg",2);
onEnterFrame=function(){
   if(_level1){
      with(_level1){_y=200; }
      delete onEnterFrame;
}}
```

将第一幅图加载到第一层；
将第二幅图加载到第二层；

蓝色脚本表示对层的位置的控制；
如果没有此段代码，两幅图会重叠；
此代码表示：如果存在第一层，
那第一层的对象的Y轴偏移200像素！（还可以添加X轴的位置，或透明度的属性等等）

- unloadMovie详解
  - 作用：卸载影片剪辑中的对象；
  - 语法：unloadMovie(目标)
  - 如：unloadMovie(mc);　mc为影片剪辑的实例名称
- unloadMovieNum详解
  - 作用：卸载层中的对象；
  - 语法：unloadMovieNum(级别)
  - 如：unloadMovieNum(1);　1表示第一层的的对象

控制影片剪辑的脚本

- 把AS写在影片剪辑本身上，选中影片剪辑,必须添加事件onClipEvent

  onClipEvent (事件)

  {//需要执行的脚本程序}

- 括号里的“事件”其实是个触发器，当事件发生时，执行该事件后面花括号中的语句。

onClipEvent触发事件

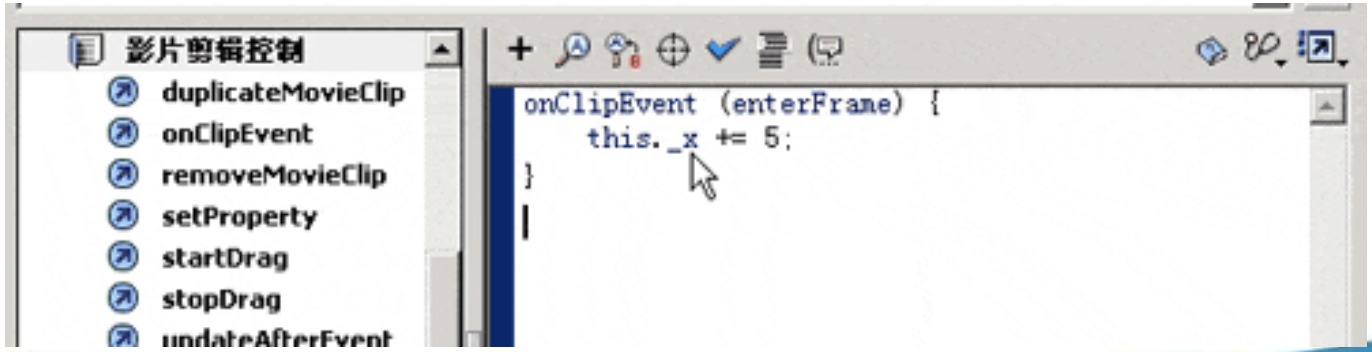
- **onClipEvent(load)** 当影片片断第一次加载到时间轴时，会触发本事件一次
  onClipEvent(enterFrame) 当影片片断加载时间轴时，不论是放映或停止状态或显示与否，都会不断触发本事件。所以只要此片断被加载后，此事件会一直不断地执行，直到影片片断被删除为止。
  onClipEvent(unload) 当影片片断被删除时，会触发本事件一次。
  onClipEvent(mouseDown)
  当鼠标左键被按下时，会触发本事件一次。

  onClipEvent(mouseUp) 当被按下的鼠标左键被放开时，会触发本事件一次。
  onClipEvent(mouseMove) 只要在场景中移动鼠标，就会不断触发本事件。
  onClipEvent(keyDown) 当键盘被按下时，会触发本事件。
  onClipEvent(keyUp) 当已按下的键盘被松开时，会触发本事件一次

影片剪辑控制实例

- 1、 新建立一个电影剪辑元件，里面随便画一个圆。之后把这个影片剪辑拖放到舞台之中（也就是创建一个此影片剪辑的实例）。
- 2、 选中这个MC，按F9打开动作面板，按照图01显示选择onClipEvent，之后在显示的事件中选择enterFrame，然后在里面编写脚本如下：

- onClipEvent (enterFrame) {// enterFrame的意思是以影片帧频不断地触发此动作
- this._x += 5; //this代表这个影片剪辑自身。_x表示影片剪辑的X轴坐标。
- }



控制影片剪辑属性

1、影片剪辑在场景中的位置（效果）

- 由“_x”和“_y”属性决定的。

若每播放一帧，影片剪辑向右和向下移动10像素的位置：

```
onClipEvent(enterFrame){
this._x+=10;
this. _y+=10;
}
```

- 2、控制影片剪辑的旋转(效果)

  使用“_rotation”属性，-180~180 度之间，如：_rotation+=10；
- 3、控制影片剪辑的透明度和可见性

  使用“_alpha”属性，0~100 之间，如：mc._alpha=50；_alpha+=10；

  使用“_visible”属性，true或false之间；

4、控制影片剪辑的大小（效果）

使用“_width”和“_height”属性，及表示纵、横向缩放百分比的“_xscale”和“_yscale”属性，如：

```
onClipEvent(enterFrame){
  _width =_ width /2;
  _height =_ height /2;
}
onClipEvent(enterFrame){
_xscale =100*（_root._xmouse-_x）/100;
_yscale =100*（_root._ymouse-_y）/100;
}
```