北京科海培训中心

以完成作品为教学导向，真正整合理论与实务

JavaScript 案例教程

- 精选数十个 JavaScript 实战案例，整理成十余种特效类型，悉心讲解其制作方法与使用技巧
- 使用者只需对案例中相关部分稍加修改，即可灵活套用到自己的网页中
- 既可供网页制作人员参考，又可用作培训教材

- 所有案例的页面效果与源代码
- 相关辅助工具

中科多媒体电子出版社

北京科海培训中心

JavaScript 案例教程

贾利峰　胡 琳　编著

中科多媒体电子出版社

2001.9

内 容 提 要

JavaScript 是一种通用的、面向对象的高级脚本语言。本书精选了数十个 JavaScript 实例，并把它们整理成图像特效、鼠标特效、状态栏特效、页面特效和文本特效等十余种类型，一种类型组成一篇，每一篇又包含数个精彩的实例，分别从不同的方面讲述了某一种特效的制作方法和技巧。每一篇的内容既相互独立，又相互支持，全书浑然一体，集中介绍了 JavaScript 在网页制作上的各种技巧。

本书构思精巧，内容充实，实例新颖，语言风格大众化，是网络爱好者及网页制作人员理想的参考资料，也可作为相关人员的培训教材使用。本书的配套光盘中提供了所有案例的效果预览页面、源代码及相关辅助工具，详情请双击光盘根目录下的 index.htm。

品　　名：JavaScript 案例教程
作　　者：贾利峰　胡 琳
责任编辑：王超辉
出　　版：中科多媒体电子出版社
印 刷 者：北京门头沟胶印厂
发　　行：新华书店总店北京科技发行所
开　　本：787×1092　1/16　印张：11.875　字数：274.5 千字
版　　次：2001 年 9 月第 1 版　2001 年 9 月第 1 次印刷
印　　数：0001~5000
盘　　号：ISBN 7-900084-16-9
定　　价：19.00 元（1CD）

前　言

随着 Internet 技术的突飞猛进，各行各业都在加入 Internet 的行列中。而 WWW 发展到今天，已成为当前 Internet 上最受欢迎、最为流行、最新型的信息资源。它利用超文本和超媒体技术结合 HyperLink（超链接）的链接功能将各种信息组织成 Web（网络结构），构成网络 Document（文档），实现 Internet 上的“漫游”。而描述 WWW 网上信息资源的是 HTML（超文本标记语言），通过 HTML 符号的描述就可以实现文字、表格、声音、图像、动画等多媒体信息的浏览。然而，采用这种超链接技术存在一定的缺陷，那就是它只能提供一种静态的信息资源，缺少动态的客户端与服务器端的交互。虽然可通过 CGI（Common Gate Interface，通用网关接口）实现一定的交互，但由于该方法编程较为复杂，因而在一段时间内妨碍了 Internet 技术的发展。而 JavaScript 的出现，有效地解决了 WWW 浏览所存在的问题，这给 Internet 用户带来了一线生机。

JavaScript 的出现，使得信息和用户之间不仅是一种显示和浏览的关系，而且还具有了一种实时的、动态的、可交互的表达能力，从而使基于 CGI 静态的 HTML 页面将要被这种可提供动态实时信息，并对客户操作进行响应的 Web 页面所取代。JavaScript 脚本正是满足这种需求而产生的语言，它深受广大用户的喜爱和欢迎，是众多脚本语言中较为优秀的一种。因此，尽快了解和掌握 JavaScript 脚本语言的基本知识和基本编程方法是广大用户的迫切要求。

现在，有些读者积累的 JavaScript 知识可能已经足够丰富，但是却很难把这些知识灵活地运用到实战当中。本书就是专门为这样的读者所写。本书没有采用传统教程的编写模式，略去了基础知识部分，而是用大量的案例来展示 JavaScript 在网页制作上的各种方法和技巧，并配备有详尽的解释，指导读者如何在这些案例中修改相关部分，以便灵活地套用到自己的网页中。读完本书，相信您对 JavaScript 的使用一定能更上一层楼。

本书精心整理了 JavaScript 的数十个案例，并把它们归纳为图像特效、鼠标特效、状态栏特效、页面特效和文本特效等类型，分别从各个不同的角度来揭示用 JavaScript 制作动态网页的方法和技巧。

由于编者水平有限，不足之处在所难免，恳请读者提出宝贵意见。

编者

2001 年 9 月

目　录

第 1 篇　图 像 特 效

案例1　改变图像隐现效果

【效果说明】

从本篇开始，将要讲述用 JavaScript 小程序处理图形图像的一些技巧，下面是一个最简单的例子。

打开本例的程序后，可以看到一幅模糊的图像，如图 1.1 所示。把鼠标移入图像区域，图像变得清晰了，如图 1.2 所示。而当鼠标移出后，图像又会恢复原样。

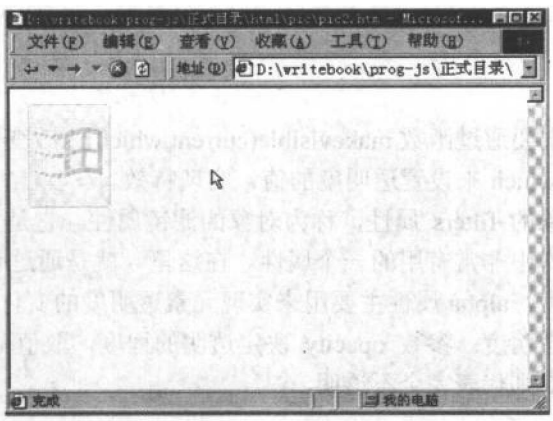

图 1.1　模糊的图像

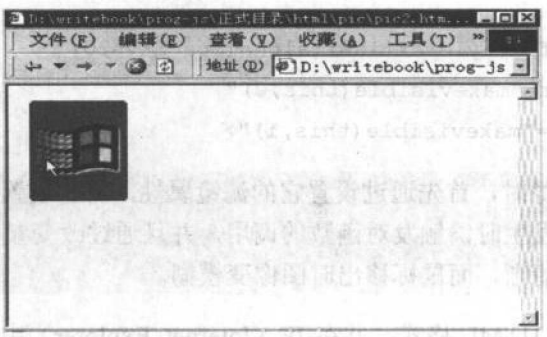

图 1.2　图像变清晰

【创作步骤与关键点解析】

下面是制作此页面的步骤。

1. 启动 FrontPage 2000，并确定已进入“网页”视图（选择“查看”菜单下的“网页”命令）。

2. 单击窗口底部的 HTML 标签，切换到 HTML 源文件编辑模式。

3. 将光盘内 sample\pic\visible.txt 中的“Part1”部分代码拷贝到<head>与</head>标记之间。

```
<script language="JavaScript">

<!-- Begin
function makevisible(current,which){
if (which==0)
   current.filters.alpha.opacity=100
else
   current.filters.alpha.opacity=20
}//  End -->

</script>
```

可以看到，程序主要通过函数 makevisible(current,which)来改变图像对象的透明度，并且通过传递来的参数 which 来设置透明度的值，实现特效。

程序中用到的对象的 filters 属性，称为对象的滤镜属性。它是页面特效制作，尤其是图形图像方面特效制作中非常有用的一个属性。在这里，就是通过它包含的一个 alpha 属性来设置图像的透明度。alpha 属性主要用来实现元素透明度的变化，并可通过指定坐标，来设置点、线、面的透明度。参数 opacity 决定透明的程度，取值范围从 0～100，其中 0 代表完全透明，而 100 则代表完全不透明。

4. 将光盘内 sample\pic\visible.txt 中的“Part2”部分代码拷贝到<body>与</body>标记之间。

```
<img src="2.gif" style="filter:alpha(opacity=20)"
    onMouseOver="makevisible(this,0)"
    onMouseOut="makevisible(this,1)">
```

在页面中放置图像时，首先通过设置它的滤镜属性，将它设置成模糊的效果。然后当鼠标移入和移出事件发生时，触发对函数的调用，并且通过改变参数 which 的值来实现，当鼠标移入时图像变清晰，而鼠标移出时图像变模糊。

5. 将文件保存为 HTML 格式，并在 IE（Internet Explorer）中打开，就可以看到前面所述的效果。

注意：从本案例开始，对案例中关于程序代码的解释段落加了底纹，以便与一般文字有所区分。

案例2　图片变形扭曲

【效果说明】

本例也是一个很简单的使用 JavaScript 小程序实现图像特效的例子。

打开本例的程序后，可以看到一幅尺寸正常的图像，如图 2.1 所示。随着时间变化，图像沿 x 轴逐渐拉长、扭曲变形，如图 2.2 所示；然后又恢复原状，并如此不断往复循环。

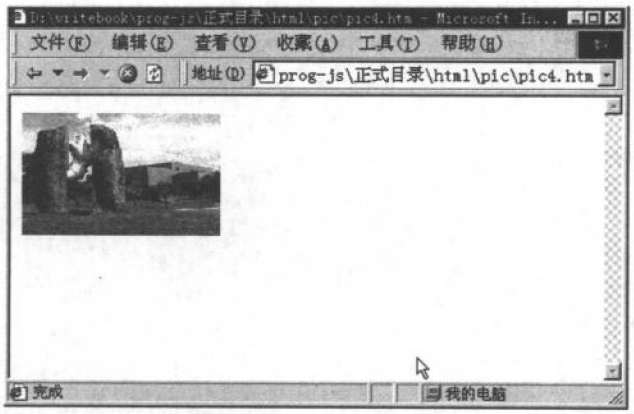

图 2.1　未变形的图像

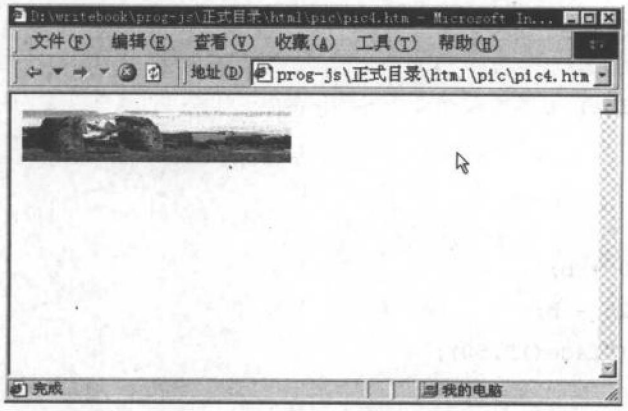

图 2.2　变形拉长的图像

【创作步骤与关键点解析】

下面是制作此页面的步骤。

1. 启动 FrontPage 2000，并确定已进入“网页”视图（选择“查看”菜单下的“网页”命令）。

2. 单击窗口底部的 HTML 标签，切换到 HTML 源文件编辑模式。

3. 将光盘内 sample\pic\fade.txt 中的“Part1”部分代码拷贝到<head>与</head>标记之间。

```
<script language="JavaScript">

var b = 1;
var c = true;

function fade(){
  if(document.all);

  if(b==100) {
     c = false
  }

  if(b==10) {
    c = true;
  }

  if(c == true) {
    b++;
  }

  if(c == false) {
    b--;
  }

  u.width=150 + b;
  u.height=125 - b;
  setTimeout("fade()",50);
}
</script>
```

程序代码一开始，先为两个变量设置初值 b=1，c=true。其中，b 的作用是改变图像的宽高的值，而 c 是用来决定 b 是否增减。

function fade( )为主要函数，用来实现动态改变图像的宽高值，其过程如下：

b 首先以 1 为单位递增；当 b=100 时，则 c=false，此时，b 以单位 1 递减，当 b=10 时，则 c=true，这时 b 又以单位 1 递增；

接着，函数把图像对象的宽度 width 和高度 height 对 b 分别做加减，就可以实现图像的扭曲变化。

最后，需要注意的是语句 setTimeout("fade()",50)，这个函数的功能是每隔第二个参数的时间就执行第一个参数所指的表达式，具体到这里来说，就是每隔 50 毫秒就调用函数 fade( )一次，如此可实现图像变形效果的往复循环。

4. 将光盘内 sample\pic\visible.txt 中的“Part2”部分代码拷贝到<body>与</body>标记之间。

```
<img src="1.gif" name="u" border="0" alt="很好玩的">
```

在页面中只放置了一幅图像，并把它命名为 u，当然也可以用其他的名字来代替。不过要注意将前面程序中用到该名字的相应地方也要改正过来。

5. 将文件保存为 HTML 格式，并在 IE 浏览器中打开就可以看到前面所述的效果。

案例3　雪　景

【效果说明】

相对前面的两个比较简单的例子来说，本例中介绍的这个“雪景”特效就有一定的难度了。

打开页面本例的程序后，可以看到片片雪花在窗口中飘飘落下，如图 3.1 所示，是不是很美丽？而且，用户可以通过简单修改一下程序，使雪花变得“漫天飞舞”，如图 3.2 所示。很有意思吧！

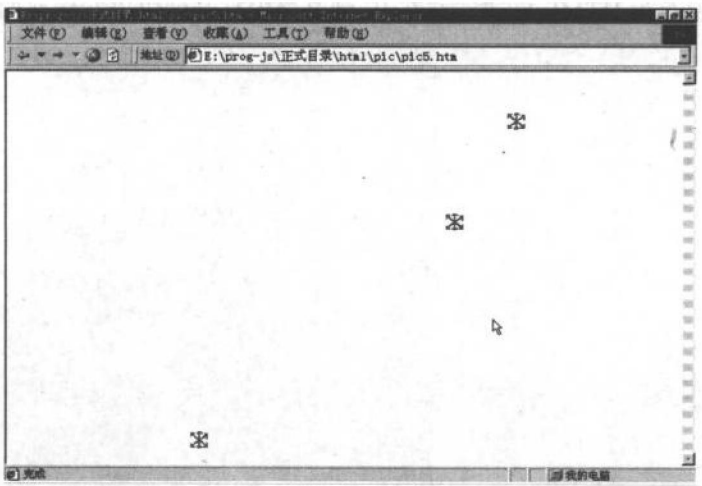

图 3.1　雪景（1）

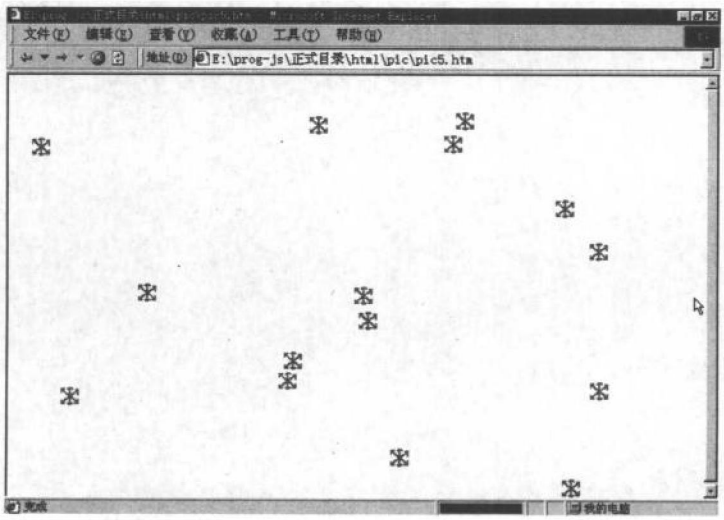

图 3.2　雪景（2）

【创作步骤与关键点解析】

下面是制作此页面的步骤。

1. 启动 FrontPage 2000，并确定已进入“网页”视图（选择“查看”菜单下的“网页”命令）。

2. 单击窗口底部的 HTML 标签，切换到 HTML 源文件编辑模式。

3. 将光盘内 sample\pic\snow.txt 中的“Part1”部分代码拷贝到<body>与</body>标记之间，下面逐步讲解代码的作用。

先看看最开始的一段代码：

```
var no = 12; // number of hearts
var heart = "heart.gif";
```

上述代码首先设定了雪花的数目，用户可以在这里进行修改，随自己的喜好选择雪花的数目。同时，这里也选定了雪花的图片。当然，用户可以选择自己喜欢的图片，使之呈现雪花飘舞的效果。

注意：把图片的路径设定好，否则，可能什么也看不到。

```
var flag;
var ie4up = (document.all) ? 1 : 0;
var dx, xp, yp;    // coordinate and position variables
var am, stx, sty;  // amplitude and step variables
var i, doc_width = 800, doc_height = 600;
dx = new Array();
xp = new Array();
yp = new Array();
amx = new Array();
amy = new Array();
stx = new Array();
sty = new Array();
flag = new Array();
```

上面的这一段代码设定了程序中要用到的一些变量，并对它们进行了初始化。其中，ie4up 用来判断用户浏览器是否是 IE4.0 以上版本，而 flag 是用来决定雪花飘动的左右方向的一个变量，dx、xp、yp 和 amx、amy、stx、sty 则分别是用来设定雪花位置和移动幅度等值的变量。显然，每一片雪花都需要独自使用一组变量来描述，所以需要用数组来保存这些变量。

```
if(ie4up){
doc_width = document.body.clientWidth+document.body.scrollLeft;
doc_height = document.body.clientHeight+document.body.scrollTop;
}
```

```
for (i = 0; i < no; ++ i) {
dx[i] = 0;                        // set coordinate variables
xp[i] = Math.random()*(doc_width-30)+10;  // set position variables
yp[i] = Math.random()*doc_height;
amy[i] = 12+ Math.random()*20;          // set amplitude variables
amx[i] = 10+ Math.random()*40;
stx[i] = 0.02 + Math.random()/10; // set step variables
sty[i] = 0.7 + Math.random();     // set step variables
flag[i] = (Math.random()>0.5)?1:0;

if (ie4up) {
if (i == 0) {
document.write("<div id=\"dot"+ i +"\" style=\"POSITION: ");
document.write("absolute; Z-INDEX: "+ i +"; VISIBILITY: ");
document.write("visible; TOP: 15px; LEFT: 15px;\"><img src=\"");
document.write(heart+ "\" border=\"0\"></div>");
} else {
document.write("<div id=\"dot"+ i +"\" style=\"POSITION: ");
document.write("absolute; Z-INDEX: "+ i +"; VISIBILITY: ");
document.write("visible; TOP: 15px; LEFT: 15px;\"><img src=\"");
document.write(heart+ "\" border=\"0\"></div>");
}
}
}
```

在上面的程序中，程序先获取了页面实际的宽度和高度：doc_width 和 doc_height，它们分别是用当前窗口的宽度 document.body.clientWidth 和高度 document.body.clientHeight 再加上因滚屏而造成的横向位移 document.body.scrollLeft 和纵向位移 document.body.scrollTop 而得到的。

然后用一个循环语句分别为每个雪花元素的变量赋初值，这里用到了一个数学方法 Math.random( )，它可以产生一个 0～1 之间的伪随机值，这样，每片雪花的位置、移动、速度都是随机的。

最后，用标记 为每个雪花图片建立放置的容器，并为它们按序编号，指定路径和一些基本属性。

```
function snow() {
doc_width = document.body.clientWidth+document.body.scrollLeft;
doc_height = document.body.clientHeight+document.body.scrollTop;

for (i = 0; i < no; ++ i) {  // iterate for every dot
```

```
if (yp[i] > doc_height-50) {
xp[i] = 10+ Math.random()*(doc_width-amx[i]-30);
yp[i] = document.body.scrollTop;
stx[i] = 0.02 + Math.random()/10;
sty[i] = 0.7 + Math.random();
flag[i]=(Math.random()<0.5)?1:0;
}

if (flag[i])
dx[i] += stx[i];
else
dx[i] -= stx[i];

if (Math.abs(dx[i]) > Math.PI) {
yp[i]+=Math.abs(amy[i]*dx[i]);
xp[i]+=amx[i]*dx[i];
dx[i]=0;
flag[i]=!flag[i];
}

document.all["dot"+i].style.pixelTop = yp[i] +
amy[i]*(Math.abs(Math.sin(dx[i])+dx[i]));
document.all["dot"+i].style.pixelLeft = xp[i] +
amx[i]*dx[i];
}
setTimeout("snow()", 10);
}
```

最后讲解一下主函数 snow()。

由于页面可能处于不停翻转状态中，因此，程序一开始就再次获取了变量 doc_width 和 doc_height，以备后面使用。

然后，以雪花数目“no”为循环上限，其目的是为了改变每个雪花的位置、移动方向等属性。开始作了一个判断“yp[i] > doc_height-50”，这是为了知道雪花是否已经飘到屏幕最下方。如果是，就把雪花的纵坐标值设到屏幕最上方的位置：yp[i] = document.body.scrollTop。

接下来，用变量 flag[i]来决定雪花飘动的方向是向左还是向右。在这里，程序通过一个判断“Math.abs(dx[i]) > Math.PI”：当 dx[i]的绝对值大于 π 之后，就作一次变向——将 dx[i]的值重新置 0，将 flag[i]取非。

做完各项工作之后，将每个雪花改变后的位置值赋给相应的保存有图片的容器，这样，

屏幕上的雪花才会真正改变位置。

最后，用函数 setTimeout("snow()", 10)来每隔 10 毫秒调用主函数 snow( )，实现循环。

4. 将<body>标记改成<body onload="snow()">，以便在页面装载完毕时调用 snow( )函数。

5. 将文件保存为 HTML 格式，并在 IE 浏览器中打开就可以看到前面所述的效果。

案例4　相片选择器

【效果说明】

在本例中，通过一个下拉列表框来控制图片的显示，它可以被用来制作相片保存簿，只要选中相应的文字就可以在窗口中看到相片了。

打开本例的程序后，可以看到一个“请选择相片”的下拉列表框，如图 4.1 所示。单击下三角按钮，可以看到选择相片的文字说明。任选中一个，如图 4.2 所示，会打开一个新窗口，并在其中显示所选的相片，如图 4.3 所示。单击新窗口中的“关闭窗口”按钮可以关闭它。

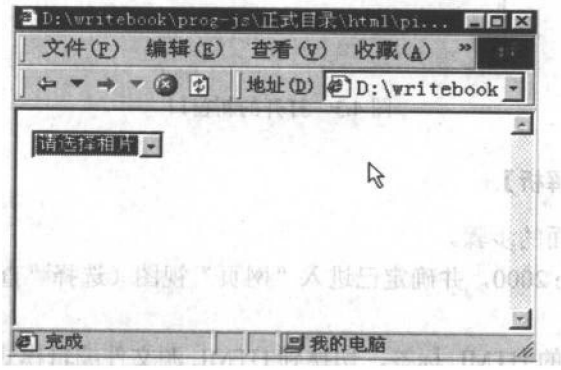

图 4.1　初始的窗口

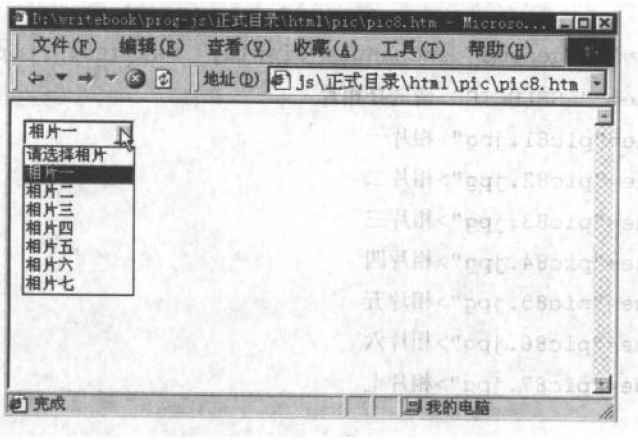

图 4.2　选择相片