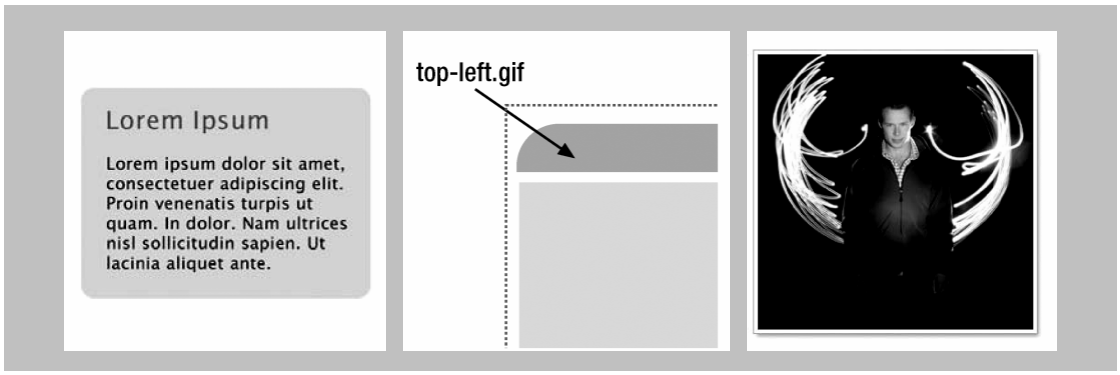
第 3 章 背景图像和图像替换



既然你已经掌握了理论，我们就开始把理论运用到实践中。当今的 Web 是一种视觉效果非常丰富的媒体。简便的图像标签使网页设计人员能够将毫无趣味的文档变成图形丰富的浏览体验。图形设计人员很快就掌握了 image 标签（原本是作为向网站中添加可视内容的方式），将它作为对页面进行视觉修饰的方式。实际上，如果没有发明 image 标签，那么可能就没有网页设计师这种职业。

不幸的是，对 image 标签的滥用导致纯修饰性的图像把页面弄乱了。好在 CSS 使我们能够在页面上显示图像，而不需要让图像成为标签的一部分。实现方法是将图像作为背景添加到现有的元素中。本章将通过一系列实际示例讲解如何使用背景图像创建各种有意思且有用的技术。

在本章中，你将学习：

- 固定宽度和可变宽度的圆角框。
- 滑动门技术。
- 山顶角。
- CSS 阴影。
- 用于 IE 5.*x* 和更高版本的 PNG 透明度支持。
- 图像替换。

3.1 背景图像基础

应用背景图像是很容易的。如果希望网站有一个好看的背景，那么只需将图像作为背景应用

于主体元素：

```
body {
 background:url(pattern.gif);
}
```

浏览器的默认行为是水平和垂直地重复显示这个图像，让图像平铺在整个页面上。可以选择背景图像是垂直平铺、水平平铺，还是根本不平铺。

目前渐变非常时髦，你可能希望在页面上应用垂直渐变。为此，需要创建一个很高但很窄的渐变图像，然后将这个图像应用于页面的主体并让它水平平铺：

```
body {
 background: #ccc url(gradient.gif) repeat-x;
}
```

因为这个渐变图像的高度是固定的，所以如果页面内容的长度超过了图像的高度，那么渐变就会突然终止。可以创建一个非常长的图像，逐渐变化到一个固定的颜色。但是，很难预测页面会有多长。实际上，只需再添加一个背景颜色。背景图像总是出现在背景颜色的上面，所以当图像结束时，颜色就会显示出来了。如果选择的背景颜色与渐变底部的颜色相同，那么图像和背景颜色之间的转换就看不出来了。

平铺图像在某些情况下很有用。但是，在大多数情况下，希望在页面上添加不进行平铺的图像。例如，假设希望在网页的开头显示一个大的品牌图像，那么只需将图像直接添加到页面上，在许多情况下这样做就够了。但是，如果图像不包含信息，是纯装饰性的，那么可能希望将图像从其余内容中分离出来。实现的方法是在 HTML 中为这个图像创建一个“钩子”，然后使用 CSS 应用这个图像。在下面的示例中，我在标记中添加一个空的 div 并且给它设置 ID branding。然后可以将这个 div 的尺寸设置为与品牌图像相同，作为背景应用图像并指定不进行平铺。

```
#branding {
  width: 700px;
  height: 200px;
  background:url(/images/branding.gif) no-repeat;
}
```

还可以设置背景图像的位置。假设希望在站点的每个标题上添加一个符号，如图 3-1 所示。可以编写下面这样的代码：

```
h1 {
  padding-left: 30px;
  background: url(/images/bullet.gif) no-repeat left center;
}
```

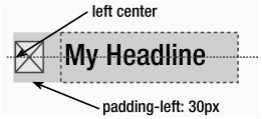


图 3-1　使用背景图像创建符号

最后两个关键字指出图像的位置。在这个示例中，图像定位在元素的左边并且垂直居中。除了使用关键字之外，还可以使用像素或百分数等单位设置背景图像的位置。

如果使用像素设置背景位置，那么图像左上角到元素左上角的距离为指定的像素数。所以，如果指定垂直和水平位置都是 20 像素，那么图像左上角出现在元素左上角下面 20 像素、左边 20 像素的地方。但是，使用百分数进行背景定位的工作方式不太一样。百分数定位并不对背景图像的左上角进行定位，而是使用图像上的一个对应点。所以，如果指定垂直和水平位置都是 20%，那么实际上将图像上距离左上角 20%的点定位到父元素上距离左上角 20%的位置（见图 3-2）。

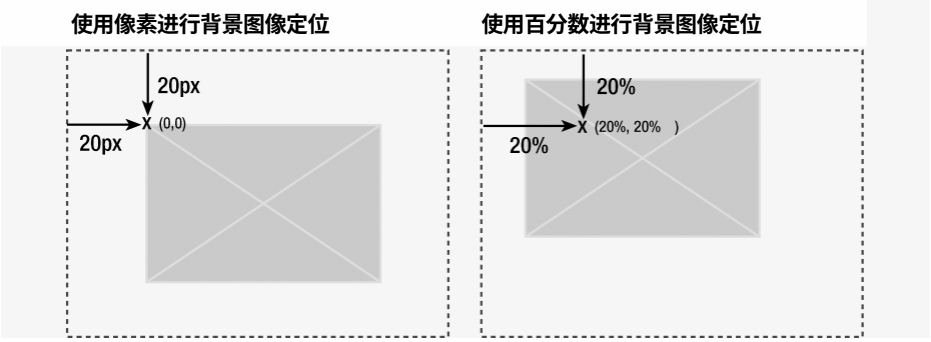


图 3-2　在使用像素进行背景图像定位时，使用图像的左上角。在使用百分数进行背景图像定位时，使用图像上的对应位置

如果希望使用百分数而不是关键字实现前面的示例，那么要将垂直位置设置为 50%，这会使符号图像垂直居中：

```
h1 {
  padding-left: 30px;
  background: url(/images/bullet.gif) no-repeat 0 50%;
}
```

规范指出，不要将像素或百分数等单位与关键字混合使用。这似乎是一个没有意义的规则，而且许多现代浏览器故意忽略了这个规则。但是，混合使用单位和关键字在某些浏览器上会导致错误，而且很可能使页面失效。因此，最好不要混合使用单位和关键字。

尽管背景图像是一个容易掌握的概念，但是它们构成了许多高级 CSS 技术的基础。

3.2　圆角框

对基于 CSS 的设计最初的批评意见之一是 CSS 太死板了，只能建立方框。为了解决这个问题，人们开始创建具有曲线的设计。圆角框很快成为最时髦的 CSS 技术之一。创建圆角框有好几种方法。每种方法各有优缺点，对这些方法的选择主要依赖于实际情况。

3.2.1　固定宽度的圆角框

最容易创建的是固定宽度的圆角框。它们只需要两个图像：一个图像用于框的顶部，另一个用于底部。例如，假设希望创建图 3-3 这样的框样式。

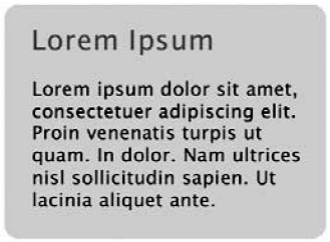


图 3-3　简单的圆角框样式

这个框的标记如下：

```
<div class="box">
  <h2>Headline</h2>
  <p>Content</p>
</div>
```

需要用图形软件创建两个图 3-4 这样的图像：一个图像用于框的顶部，另一个用于底部。这个示例以及本书中其他所有示例的代码和图像可以从 www.friendsofed.com 下载。

图 3-4　顶部和底部曲线图像

然后，将顶部图像应用于标题元素，将底部图像应用于 div 框的底部。因为这个框样式是单色的，所以可以在 div 框上添加背景颜色，从而形成框的主体。

```
.box {
  width: 418px;
  background: #effce7 url(images/bottom.gif) no-repeat  left bottom;
}

.box h2 {
  background: url(images/top.gif) no-repeat left top;
}
```

我们不希望内容碰到框的边界，所以还需要在 div 中的元素上添加一些填充：

```
.box h2 {
  padding: 10px 20px 0 20px;
}
```

```
.box p {
  padding: 0 20px 10px 20px;
}
```

这个方法对于单色而且没有边框的简单框是有效的。但是，如果希望创建像图 3-5 这样更生动的样式，那么怎么办？

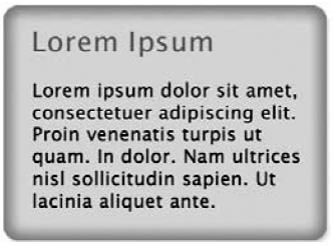


图 3-5　样式更特殊的圆角框

实际上，可以使用相同的方式，但是这一次不在框上设置背景颜色，而是设置一个重复显示的背景图像。还需要将底部曲线图像应用到另一个元素上。在这个示例中，我使用框中的最后一个段落元素：

```
.box {
  width: 424px;
  background: url(images/bg-tile.gif) repeat-y;
}

.box h2 {
  background: url(images/bg-top.gif) no-repeat left top;
  padding-top: 20px;
}

.box .last {
  background: url(images/bg-bottom.gif) no-repeat left bottom;
  padding-bottom: 20px;
}

.box h2, .box p {
  padding-left: 20px;
  padding-right: 20px;
}
<div class="box">
  <h2>Headline</h2>
  <p class="last">Content</p>
</div>
```

图 3-6 所示为生成的框。因为没有给这个框设置高度，所以它会随着文本尺寸的增加进行垂直扩展。

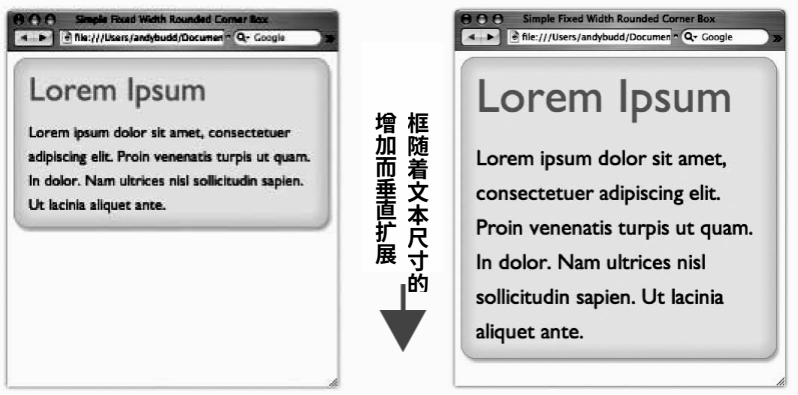


图 3-6　样式特殊的固定宽度框。框的高度会随着文本尺寸的增加而扩展

灵活的圆角框

如果加大字号，前面的示例都会垂直扩展。但是，它们不会水平扩展，因为框的宽度必须与顶部和底部图像的宽度一致。如果希望创建灵活的框，那么需要采用略有不同的方法。不用单一图像组成顶部和底部曲线，而是用两个相互重叠的图像（见图 3-7）。

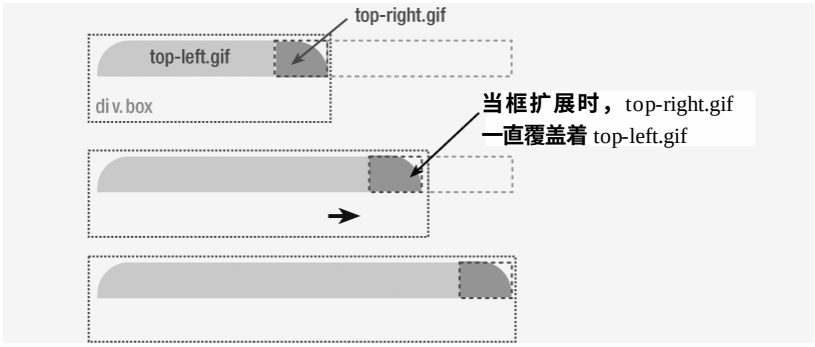


图 3-7　如何扩展顶部图像来形成灵活的圆角框

随着框尺寸的增加，大图像有更多的部分显露出来，这样就实现了框扩展的效果。这个方法有时候被称为滑动门技术（sliding doors technique），因为一个图像在另一个图像上滑动，将它的一部分隐藏起来。这个方法需要更多的图像，所以必须在标记中添加两个额外的无语义元素。

```
<div class="box">
  <div class="box-outer">
    <div class="box-inner">
    <h2>Headline</h2>
    <p>Content</p>
    </div>
  </div>
</div>
```

这个方法需要四个图像：两个顶部图像组成顶部曲线，两个底部图像组成底部曲线和框的主体（见图 3-8）。因此，底部图像的高度必须与框的最大高度相同。分别将这些图像命名为 top-left.gif、top-right.gif、bottom-left.gif 和 bottom-right.gif。

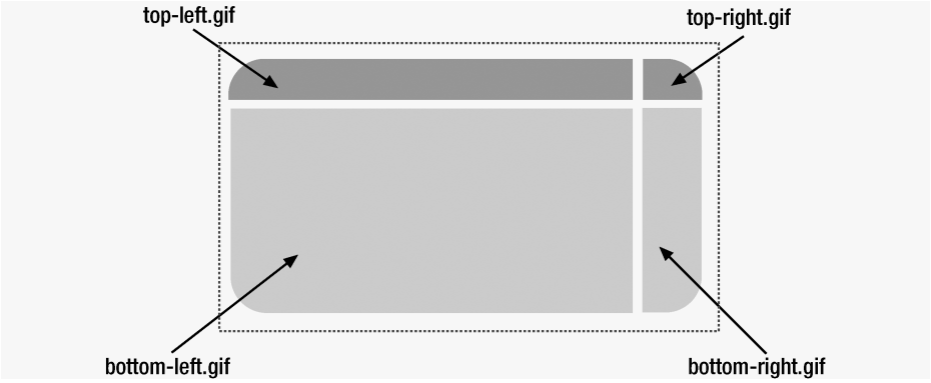


图 3-8　创建灵活的圆角框所需的图像

首先，将 bottom-left.gif 应用于主框 div，将 bottom-right.gif 应用于外边的 div。接下来，将 top-left.gif 应用于内部的 div，将 top-right.gif 应用于标题。最后，添加一些填充以便在内容周围形成一点儿空白。

```
.box {
  width: 20em;
  background: #effce7 url(images/bottom-left.gif) ➥
  no-repeat left bottom;
}
.box-outer {
  background: url(images/bottom-right.gif) no-repeat right bottom;
  padding-bottom: 5%;
}

.box-inner {
  background: url(images/top-left.gif) no-repeat left top;
}

.box h2 {
  background: url(images/top-right.gif) no-repeat right top;
  padding-top: 5%;
}

.box h2, .box p {
  padding-left: 5%;
  padding-right: 5%;
}
```

在这个示例中，我以 em 为单位设置框的宽度，所以在浏览器中增加文本尺寸时框会伸展（见图 3-9）。当然，可以用百分数设置宽度，这使框根据浏览器窗口的尺寸进行扩展或收缩。这是弹性和灵活布局背后的主要原则之一，在本书后面会进一步讨论这些原则。

图 3-9　灵活的圆角框会随着文本尺寸的增加进行水平和垂直扩展

添加两个额外的无语义元素是不理想的。如果只有很少的几个框，那么这是可以容忍的。但是，如果用到圆角框的地方很多，那么可以使用 JavaScript（和 DOM）添加额外元素。关于这个主题的更多细节，请参考 http://tinyurl.com/82y8l 上 456 Berea Street 的 Roger Johansson 所写的文章。

3.2.2　山顶角

山顶角（mountaintop corner）是一个简单但非常灵活的概念，是由 www.simplebits.com 的 Dan Cederholm 首创的，他是畅销图书 *Web Standards Solutions*（friends of ED，2004）的作者。假设希望创建一系列具有不同颜色的圆角框。如果使用前面的方法，就必须为每种颜色方案创建不同的角图像。如果只有几个方案，那么这个方法也可以，但是，如果想让用户创建自己的颜色方案，那么该怎么办？可能必须在服务器上动态地创建角图像，这是非常复杂的。

幸运的是，有另一个办法。并不创建有颜色的角图像，而是创建曲线形的位图角蒙板（见图 3-10）。蒙板区域盖住背景颜色，而角区域实际上是透明的。当放在有颜色的框上时，它们形成曲线形框的效果（见图 3-11）。

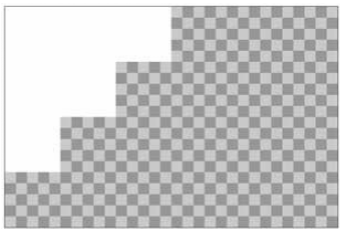


图 3-10　位图角蒙板。白色的蒙板将覆盖背景颜色，产生简单的曲线效果

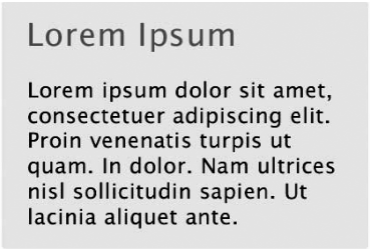


图 3-11　山顶角框

因为这些角蒙板是位图，所以对于小曲线效果最好。如果使用大曲线，那么它会出现锯齿，不好看。

基本的标记与前一个方法相似，它需要四个元素来应用四个角蒙板：

```
<div class="box">
  <div class="box-outer">
    <div class="box-inner">
    <h2>Headline</h2>
    <p>Content</p>
    </div>
  </div>
</div>
```

CSS 也非常相似：

```
.box {
  width: 20em;
  background: #effce7 url(images/bottom-left.gif) ➥
  no-repeat left bottom;
}
.box-outer {
  background: url(images/bottom-right.gif) no-repeat right bottom;
  padding-bottom: 5%;
}

.box-inner {
  background: url(images/top-left.gif) no-repeat left top;
}

.box h2 {
  background: url(images/top-right.gif) no-repeat right top;
  padding-top: 5%;
}

.box h2, .box p {
  padding-left: 5%;
  padding-right: 5%;
}
```

除了使用不同的图像之外，主要的差异是在主框 div 上添加了背景颜色。如果要修改框的颜色，只需修改 CSS 中的颜色值，不必重新创建任何新图像。这个方法只适合创建非常简单的框；但是它提供了很大的灵活性，而且可以在不同的项目中重复使用。

3.3　阴影

阴影是一种很流行、很有吸引力的设计特性，它给平淡的设计增加了深度，形成立体感。大多数人使用 Photoshop 这样的图形软件直接给图像添加阴影。但是，可以使用 CSS 产生简单的阴影效果，而不需要修改底层的图像。

这么做有几个原因。例如，你可能让非技术人员管理站点，而他不会使用 Photoshop，或者要从无法使用 Photoshop 的地方（比如网吧）上载图像。通过使用预定义的阴影样式，只需上载常规图像，它在站点上就会带着阴影显示。

使用 CSS 的最大好处之一是灵活性。如果以后想去掉阴影效果，那么只需要在 CSS 文件中修改几行代码，而不必重新处理所有图像。

3.3.1　简单的 CSS 阴影

www.1976design.com 的 Dunstan Orchard 首先描述了这个非常简单的阴影方法。它的工作原理是：将一个大的阴影图像应用于容器 div 的背景。然后使用负值的空白边偏移这个图像，从而显示出阴影。

首先需要创建阴影图像。我使用 Adobe Photoshop 创建阴影图像。创建一个新的 Photoshop 文件，其尺寸与图像的最大尺寸一样。为了保险，我创建一个 800×800 像素的文件。打开背景层并且填充一种颜色，阴影将放在这种颜色上面。我让背景层保持白色。创建一个新的层并且填充上白色。现在，将这个层向左上方移动 4 或 5 个像素，然后对这个层应用 4 或 5 像素宽的阴影。保存这个文件并将它命名为 shadow.gif（见图 3-12）。

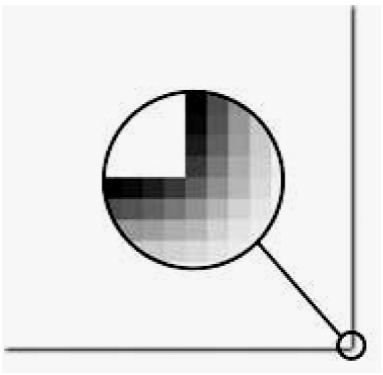


图 3-12　800×800 像素的 shadow.gif，在放大图中可以看到 5 像素宽的阴影

这种技术的标记非常简单：

```
<div class="img-wrapper"><img src="dunstan.jpg" width="300"➥
height="300" alt="Dunstan Orchard" /></div>
```

> 一定要将这些代码放在一行上，而且在 div 和图像之间不能有空格。IE 5.5有一个空格 bug，如果代码在不同的行上，那么这个 bug 会在图像和阴影之间造成间隙。

为了产生阴影效果，首先需要将阴影图像应用于容器 div 的背景。因为 div 是块级元素，所以它们会水平伸展，占据所有可用空间。在这种情况下，希望 div 包围图像。可以显式地设置容器 div 的宽度，但是这么做会限制这种技术的灵活性。可以对 div 进行浮动，让它在现代浏览器上产生“收缩包围”的效果。Mac 上的 IE 5.*x* 不支持这种技术，可以对 Mac 上的 IE 5.*x* 隐藏这些样式。关于对各种浏览器隐藏规则的更多信息，请参见第 8 章，那里会讨论各种招数和过滤器。

```
.img-wrapper {
  background: url(images/shadow.gif) no-repeat bottom right;
  clear: right;
  float: left;
}
```

为了露出阴影图像并产生阴影效果（见图 3-13），需要使用负值的空白边偏移这个图像：

```
.img-wrapper img {
  margin: -5px 5px 5px -5px;
}
```

图 3-13　应用了阴影的图像

还可以给图像加上边框和一些填充，从而产生类似照片边框的效果（见图 3-14）：

```
.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
  margin: -5px 5px 5px -5px;
}
```

图 3-14 最终结果

这种技术对于大多数符合标准的现代浏览器都是有效的。但是，为了在 IE 6 中产生正确的效果，还需要添加两个简单的规则：

```
.img-wrapper {
  background: url(images/shadow.gif) no-repeat bottom right;
  clear: right;
  float: left;
  position: relative;
}

.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
  display: block;
  margin: -5px 5px 5px -5px;
  position: relative;
}
```

阴影效果现在可以在 IE 6 中实现了。在 IE 5.*x* 中，不显示图像上的填充，但是这是个小问题，可以不用理会。

3.3.2　来自 Clagnut 的阴影方法

www.Clagnut.com 的 Richard Rutter 提供了一个相似的创建阴影的方法。他的技术不使用负值的空白边，而是使用相对定位来偏移图像：

```
.img-wrapper {
  background: url(images/shadow.gif) no-repeat bottom right;
  float:left;
  line-height:0;
}

.img-wrapper img {
  background:#fff;
  padding:4px;
  border:1px solid #a9a9a9;
  position:relative;
  left:-5px;
  top:-5px;
}
```

在 IE 5.*x* 中仍然不显示图像上的填充，但是浏览器对这个方法的支持大体上很好。

3.3.3　模糊阴影

前两种方法提供了创建阴影效果的简单方式。但是，对它的主要批评意见是阴影的边缘看起来很硬。如果在 Photoshop 这样的图形软件中创建阴影效果，那么边缘会淡入背景，产生更加自然的效果。在图 3-15 中可以看到这两种效果的对比。

图 3-15　一些人不喜欢前面的技术产生的生硬边缘，希望更有真实感

幸运的是，通过使用 PNG 和蒙板并添加一个无语义的 div，可以实现这种效果。这个方法创建一个具有 alpha 透明度的 PNG 来盖住阴影图像的边缘。

首先需要建立蒙板 PNG。创建一个新的 800×800 像素的 Photoshop 文件。删除背景层的内容，然后在左边缘上建立一个 5 像素宽的选区。在这个选区中填入从白色到透明的渐变。在图像顶部建立 5 像素高的选区，同样填入渐变。最终的图像在顶部和左边形成模糊的白色边缘，见图 3-16。现在，将这个文件保存为 24 位的 PNG 并且命名为 mask.png。

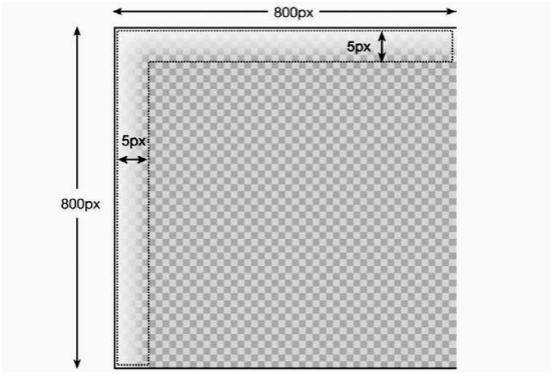


图 3-16 这个 PNG 的透明边缘将盖住阴影图像的角，形成柔和的角

不幸的是，老版本的 IE 不支持 PNG alpha 透明度。为了适应这些浏览器，需要创建一个替代图像。我创建一个简单的 GIF 蒙板，它的左边和顶部填充 5 像素宽的白色。

这种技术的标记如下：

```
<div class="img-wrapper">
 <div>
  <img src="dunstan.jpg" width="300" height="300" alt="Dunstan" />
 </div>
</div>
```

要实现这个效果，首先需要将阴影图像应用于 `img-wrapper div`，将它对准右下角：

```
.img-wrapper {
  background: url(images/shadow.gif) no-repeat right bottom;
  float: left;
}
```

接下来，将蒙板图像应用于内层 `div` 的右上角。这会将蒙板图像盖在阴影图像的上面，蒙住生硬的左边缘和上边缘，形成柔和的边缘。目前，这两个背景图像都被主图像盖住了。要进行偏移，只需在内层 `div` 的底部和右边应用一些填充：

```
.img-wrapper div {
  background: url(images/mask.png) no-repeat left top !important;
  background: url(images/mask.gif) no-repeat left top;
  padding: 0 5px 5px 0;
}
```

你会注意到，我将 PNG 和 GIF 都应用于这个规则。这是为了同时适应支持 PNG alpha 透明度的新型浏览器和不支持它的 IE 版本。通过使用 `!important` 招数，比较现代的浏览器将显示 PNG，而 IE 用户会看到 GIF。关于这个招数的更多信息，请参阅第 8 章。

如果浮动元素包含块级元素，那么 Mac 上的 IE 5.2 不对它们进行“收缩包围”。为了解决这个问题，只需同时浮动第二个和第一个 `div`：

```
.img-wrapper div {
  background: url(images/mask.png) no-repeat left top !important;
  background: url(images/mask.gif) no-repeat left top;
  padding: 0 5px 5px 0;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}
```

最后，在图像元素上添加边框效果：

```
.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
}
```

把所有这些步骤组合在一起，完整的 CSS **像下面这样：**

```
.img-wrapper {
  background: url(images/shadow.gif) no-repeat right bottom;
  float: left;
}

.img-wrapper div {
  background: url(images/mask.png) no-repeat left top !important;
  background: url(images/mask.gif) no-repeat left top;
  padding: 0 5px 5px 0;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}

.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
}
```

最终的效果应该像图 3-17 **这样。**

图 3-17　**最终效果**

如果愿意的话，采用这样的效果就可以了，为好浏览器提供 PNG，为其他浏览器提供 GIF。不幸的是，正如大家知道的，IE 占据的市场份额非常大，所以能够实际看到这种模糊阴影的人非常少。

幸运的是，IE 5.5 和更高版本提供了一些专有的 CSS，可以实现 PNG 透明度：

```
filter:progid:DXImageTransform.Microsoft.AlphaImageLoader➥
(src='images/mask.png', sizingMethod='crop');
```

可以将这些代码添加到现有的 CSS 文件中，并且使用 IE 专有招数对好浏览器隐藏它。但是，这会使 CSS 文件失效。另外，除非绝对必要，应该设法避免使用招数。更合适的方法是将规则放在一个单独的 CSS 文件中，然后对除 IE 之外的其他浏览器隐藏它。为此，创建一个新的 CSS 文件 `ie55.css` 并且添加以下代码：

```
.img-wrapper div {
  filter:progid:DXImageTransform.Microsoft.AlphaImageLoader➥
  (src='img/shadow2.png', sizingMethod='crop');
  background: none;
}
```

第一个规则使用 IE 专有的 `AlphaImageLoader` 过滤器在 IE 5.5 和更高版本中显示具有 alpha 透明度的 PNG。原来的背景图像仍然会显示，所以第二个规则隐藏原来的背景图像。

IE 还有另一段称为有条件注释（conditional comment）的专有代码，它允许向 IE 的特定版本提供特定样式表。在这个示例中，只希望让 IE 5.5 和更高版本看到这个新的样式表，所以在页面的开头添加以下代码：

```
<!--[if gte ie 5.5000]>
<link rel="stylesheet" type="text/css" href="ie55.css"/>
<![endif]-->
```

> 目前不需要为有条件注释太担心，在第 8 章中将详细讨论它们。

这样就行了。所有现代浏览器以及 IE 5.5 和更高版本都会显示柔和的阴影。其他浏览器将显示硬角的阴影。

创建能够在所有浏览器中工作的基本页面，然后为比较现代的浏览器添加高级样式或功能，这种概念称为渐进式改进（progressive enhancement）。与之相反，确保页面的样式或功能在老式浏览器中不会造成严重的问题，这称为平稳退化（graceful degradation）。这两个概念在基于标准的设计中非常重要。

3.3.4 洋葱皮阴影

我要演示的最后一种阴影方法使用与圆角框方法非常相似的技术。但是，并不是使用蒙板盖住阴影的末尾，而是创建两个阴影末尾 GIF，然后将它们覆盖在主阴影图像的末尾上。为此，需

要向标记中添加两个无语义的 div，作为应用这些图像的钩子。

基本的 HTML 如下所示：

```
<div class="img-wrapper">
 <div class="img-outer">
  <div class="img-inner">
   <img src="images/dunstan.jpg" width="300" height="300" ➥
        alt="Dunstan" />
  </div>
 </div>
</div>
```

与前面一样，将主阴影图像作为背景应用于主 div：

```
.img-wrapper {
  background:url(images/shadow.gif) no-repeat right bottom;
  float: left;
}
```

也与前面一样，对这个 div 进行浮动，让它产生收缩包围。

现在，可以将左下角图像应用于外层 div 的左下角，将右上角图像应用于内层 div 的右上角。在内层 div 的底部和左边添加一些填充，产生阴影效果。为了确保主图像容器在 Mac 上的 IE 5.2 中产生收缩包围，还需要浮动这两个 div：

```
.img-outer {
  background:url(images/bottom-left2.gif) no-repeat left bottom;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}

.img-inner {
  background:url(images/top-right2.gif) no-repeat top right;
  padding: 0 5px 5px 0;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}
```

最后与前面一样，可以在图像上添加边框和一些填充，产生照片式边框：

```
.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
  display: block;
}
```

最终的 CSS 如下所示：

```
.img-wrapper {
  background:url(images/shadow.gif) no-repeat right bottom;
  float: left;
}
```

```
.img-outer {
  background:url(images/bottom-left2.gif) no-repeat left bottom;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}

.img-inner {
  background:url(images/top-right2.gif) no-repeat top right;
  padding: 0 5px 5px 0;
  float: left; /* :KLUDGE: Fixes problem in IE5.2/Mac */
}

.img-wrapper img {
  background-color: #fff;
  border: 1px solid #a9a9a9;
  padding: 4px;
  display: block;
}
```

这种方法非常容易理解，而且它创建的阴影在许多浏览器上都可以工作。它的缺点是需要添加两个额外的无语义 div。这是因为 CSS 当前不允许在一个元素上添加多个背景图像。以后 CSS 3 将提供这个功能，所以使用多个元素只是一种过渡性方法，以后应该很容易从文档中去掉这些额外的标记。如果希望标记更简洁，那么可以使用 JavaScript 或生成的内容来添加这些额外元素。

3.4 图像替换

HTML 文本具有很多优点。搜索引擎可以读取它，开发人员可以复制并粘贴它，如果在浏览器中增加文本字号，它还会放大。因此，尽可能使用 HTML 文本而不是文本的图像是一种好想法。遗憾的是，页面设计人员对于字体只有有限的选择能力。另外，尽管可以使用 CSS 在某个范围内控制版式，但是某些效果对于文本是不可能实现的。因此，在某些情况下还是需要使用文本的图像。

由于不愿意将这些图像直接嵌入页面中，CSS 作者发明了图像替换（image replacement）的概念。可以像平常一样将文本添加到文档中，然后使用 CSS 隐藏文本并在它的位置上显示一个背景图像。这样的话，搜索引擎仍然可以搜索 HTML 文本，而且如果禁用 CSS，文本仍然会显示。

在各种缺陷暴露出来之前，这个概念看起来非常棒。但是，一些比较流行的图像替换方法对于屏幕阅读器[1]是无效的，而且如果关闭图像但是打开 CSS，就会出现内容缺失。因此，许多 CSS 作者停止使用图像替换方法，恢复为使用一般文本。虽然我也主张尽可能避免图像替换，但是仍然相信在某些情况下它是合适的，比如由于公司品牌策略的要求，需要使用特定的字体。为此，你应该很好地掌握各种技术并且了解它们的局限性。

1. 所谓屏幕阅读器，是指将屏幕显示内容转为声音或布莱叶盲文显示，以帮助视力障碍者使用电脑的软件。

3.4.1　FIR

由 Todd Fahrner 开创的 FIR（Fahrner 图像替换）是最早且可能最流行的图像替换技术。我对这个方法进行解释是因为它在历史上具有重要意义，而且它是最容易理解的方法之一。但是，这个方法有一些严重的可访问性问题（稍后会提到），因此应该避免使用。

基本概念非常简单。把要替换掉的文本放在 span 标签中：

```
<h2>
  <span>Hello World</span>
</h2>
```

然后将替换图像作为背景图像应用于标题元素：

```
h2 {
  background:url(hello_world.gif) no-repeat;
  width: 150px;
  height: 35px;
}
```

并且将 span 的 display 值设置为 none，从而隐藏 span 的内容：

```
span {
  display: none;
}
```

这个方法似乎很有效，但是最后一个规则会造成问题。许多流行的屏幕阅读器会忽略那些 display 值设置为 none 或 hidden 的元素。因此会完全忽略这个文本，造成严重的可访问性问题。所以，这种试图改进站点可访问性的技术实际上产生了相反的效果。因此，最好不要使用这种技术。

3.4.2　Phark

www.phark.net 的 Mike Rundle 发明了一种适合屏幕阅读器的图像替换技术，而且不需要添加额外的无语义 div：

```
<h2>
  Hello World
</h2>
```

Phark 方法并不使用 display 属性来隐藏文本，而是对标题应用一个非常大的负值文本缩进：

```
h2 {
  text-indent: -5000px;
  background:url(hello_world.gif) no-repeat;
  width: 150px;
  height:35px;
}
```

这个方法效果很好，解决了屏幕阅读器的问题。但是，与 FIR 方法一样，这个方法在关闭图

像但是打开 CSS 的情况下是无效的。这是一种很少见的情况，可能只有使用非常慢的连接或者使用手机作为调制解调器的人才会这样设置。站点访问者能够打开图像，但是他们可能选择不打开图像。要记住某些人可能看不见被替换的文本，所以对于重要信息或导航信息，最好避免使用这个方法。

3.4.3 Gilder/Levin 方法

这种方法是由 Tom Gilder 和 Levin Alexander 一起发明的，可能是最可靠的方法。它对屏幕阅读器是有效的，而且在关闭图像但是打开 CSS 的情况下会显示文本。它的原理是将一个图像盖在文本上而不隐藏文本。这样的话，当图像关闭时，就会看到下面的文本。

要使用这种技术，需要在要替换的元素中添加一个额外的空 span：

```
<h2>
  <span></span>Hello World
</h2>
```

然后将这个元素的尺寸设置为与图像尺寸相同，并且将元素的 position 设置为 relative：

```
h2 {
  width: 150px;
  height: 35px;
  position: relative;
}
```

这为被包含的 span 元素建立了新的定位上下文，从而可以将它绝对定位在文本的上面。将尺寸设置为父元素的 100%并且将替换图像作为背景应用于 span，这样被替换的文本就会被图像完全盖住：

```
h2 span {
  background: url(hello_world.gif) no-repeat;
  position: absolute;
  width: 100%;
  height: 100%;
}
```

在使用这种技术时，必须使用具有实色背景的图像；否则文本就会露出来。这种技术的缺点是需要添加无语义的 span。

3.4.4 IFR 与 sIFR

图像替换试图解决的主要问题之一是在大多数计算机上缺少可用的字体。为了避免将文本换成文本的图像，Mike Davidson 和 Shaun Inman 一起发明了一种更新颖的方法。

Flash 允许将字体嵌入 SWF 文件，所以他们并不把文本换成图像，而是用 Flash 文件替换文本。进行这一替换的方法是使用 JavaScript 搜索文档，找到特定元素或者具有特定类名的元素中

的所有文本。然后，JavaScript 将文本替换为一个小的 Flash 文件。接下来是真正精明的部分。这种技术并不为每段文本创建单独的 Flash 文件，而是将被替换的文本放回一个重复的 Flash 文件中。因此，触发图像替换所要做的只是添加一个类，Flash 和 JavaScript 会完成余下的工作。另一个好处是 Flash 文件中的文本是可搜索的，这意味着可以轻松地复制它。

Shaun Inman 公开了他的 Flash 图像替换方法，并且将它命名为 IFR(Inman Flash 替换)。IFR 是一种非常轻量的方法。关于这个方法的细节（包括源代码）可以在 www.shauninman.com/plete/2004/04/ifr-revisited-and-revised 上找到。

Mike Davidson 对这个方法进行了扩展，创建了 sIFR（可伸缩 Inman Flash 替换）方法。这个方法允许多行文本替换和改变文本字号。

要想在站点上使用 sIFR，首先需要从 www.mikeindustries.com/sifr 下载最新的版本。在站点上安装 sIFR 是非常简单的，但是应该先阅读文档。首先需要做的是打开 Flash 文件、嵌入希望使用的字体并且导出视频。为了让 sIFR 能够正确地工作，接下来需要应用包含的打印和屏幕样式，或者创建自己的样式。现在，将 sifr.js JavaScript 文件添加到希望使用 sIFR 的每个页面中。可以对这个文件进行许多配置，可以指定要替换的元素、文本颜色、填充、大小写以及其他许多样式属性。完成配置之后，将所有文件上传到服务器上，就会看到原来的字体换成了动态的 Flash 内容。

这些技术的主要问题涉及装载时间。页面必须完全装载，然后 JavaScript 才能替换文本。因此，在所有文本被替换为 Flash 内容之前常常有短暂的闪烁（见图 3-18）。

图 3-18　在页面装载之后，fortymedia.com 上的标题才会显示。这明确说明这个站点正在使用 sIFR

尽管这不是个大问题，但是会被访问者注意到，会给人留下页面装载慢的印象。另外，如果进行许多 Flash 替换，一些页面感觉起来有点儿迟钝。最好将替换减少到最少，只对主要标题使用这种技术。

3.5 小结

在本章中，学习了如何将背景图像应用于元素，从而产生各种有意思的技术，比如灵活的圆角框和纯 CSS 阴影；还看到了如何在 IE 中实现 PNG 支持以及进行图像替换的几种方法。

在下一章中，将学习如何组合使用背景图像和链接来实现一些有意思的交互效果。