前 言

Preface

我们的HTML

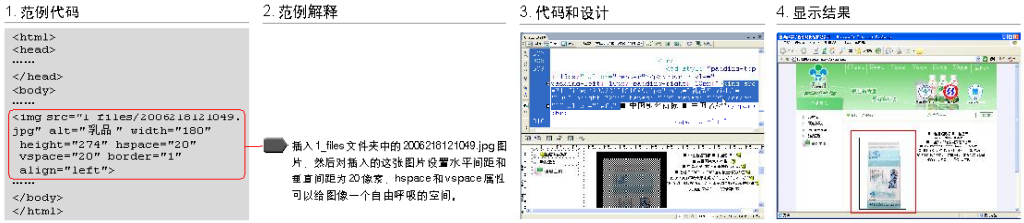
HTML 给人更多的感觉是代码，不过它与其他的程序语言有着很大的不同，它是 Hypertext Markup Language，也就是“超文本标记语言”，请大家不要忽略“标记”，它才是HTML 的精华所在，也就是<>中的内容。每个标记都有它特定的含义，精彩的 Web 页面就是靠它们描述出来的。大多数标记成对出现，中间是属性等实质性内容，后面的标记加上一个“/”，表示结束部分。这样看起来你会觉得 HTML 也不是什么高难的语言，仅此“标记组合”而已。

本书弥补了以往 HTML 图书的不足，将真实的网页设计案例融入到培训专家的讲义中，铸造了一本真正意义上从入门到精通的专家级 HTML 参考书。

本书主要特色

全书包括 13 章及附录，每个章节都配有完整的制作实例，使读者能够在真实的学习环境中掌握 HTML 语言。

- 起点较低，收获丰富——真正将 HTML 代码编写放入 Dreamweaver 8 中结合讲解，读者能直接体验从入门到精通的学习过程。
- 上手容易，轻松学习——100％网页设计真实案例讲解，让学习代码像学习可视化操作一样简单、容易。
- 代码注解，时时提示——添加注释，对每段代码中的词汇进行细致的解释，避免理解歧义的发生，让读者真正领会代码的所有含义。
- 四位一体，全面讲解——范例代码、范例解释、代码和设计、显示结果四位一体的讲解过程，把代码和制作效果直接联系在一起。



- 公开代码，方便查阅——所有实例代码及效果文件放在附赠光盘中，读者可以随时学习使用、借鉴发挥。
- 配送丰富，物超所值——16 进制 RGB 值与命名对照表、HTML 元素速查表、HTML 字符集识别等内容随附在光盘中。

看后可以学会

HTML 语言是任何高级网页制作技术的核心与基础，本书通过几十个真实的网站实例讲解了上百个语法知识点的应用。主要包括：

- HTML 语言的基本概念和特点，对 HTML 语言有一个初步的理解。
- 使用记事本和 Dreamweaver 编写 HTML 页面，迅速了解到网页的制作流程。
- HTML 网页的结构、头部信息、页面的整体设置。
- 通过 HTML 语言对字体、大小、颜色、底纹、边框等设置文本的属性。
- 使用 HTML 语言制作页面中的超级链接。
- 图片、音频、视频、动画、滚动字幕等多媒体效果在网页中的应用。
- 表单、表格、框架结构的使用，让网页更加完美。
- CSS 样式表、JavaScript 脚本的使用，是动静 Web 设计的最佳选择。

光盘附赠

1. 全书所有实例的原始文件以及完成文件。
2. 8款Dreamweaver中使用的源代码相关插件。
3. 8款优秀源代码编辑器。
4. HTML 语言电子学习手册。
5. HTML 语言授课教学幻灯片。
6. JavaScript特效代码集。
7. 网页颜色表。
8. 语法简明速查表。

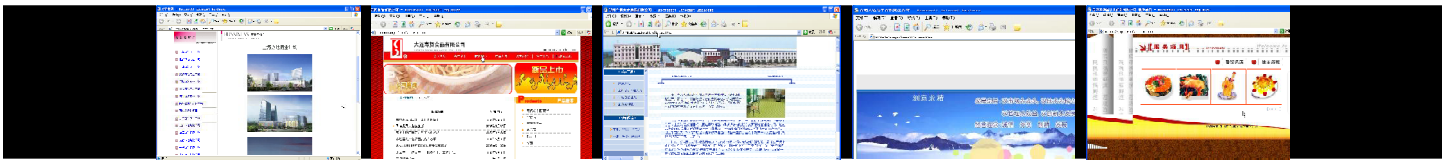
作者要说

作者在3个月的写作时间里，一直坚持不懈地投入很多精力，可以看到本书在内容的原创性、体例的特殊性和版式的创新性上下了很大的功夫，目的是希望为学习网页制作的读者提供一本讲述 HTML 语言、CSS 样式表的经典图书。从这里开始，HTML 将不再是高不可攀的代码技术，而是正在变得愈加和蔼可亲起来。

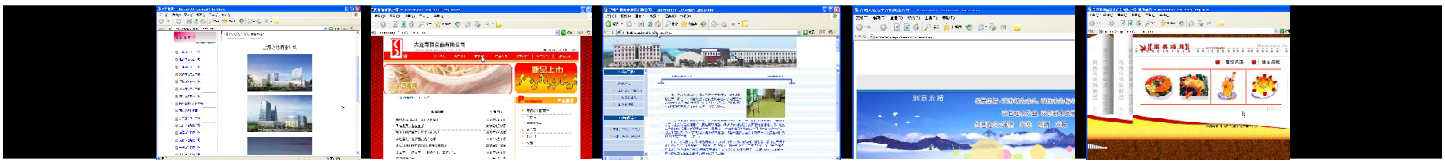
作　者

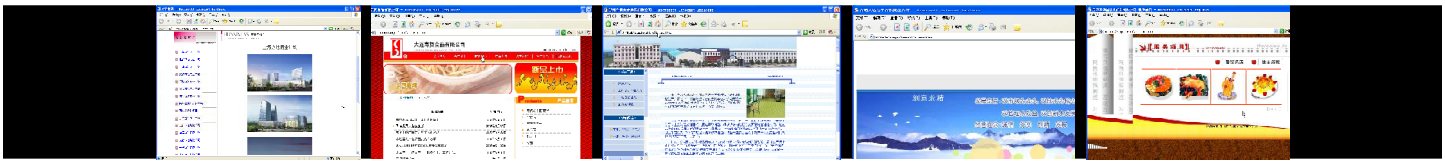
2007年1月

目 录 CONTENTS

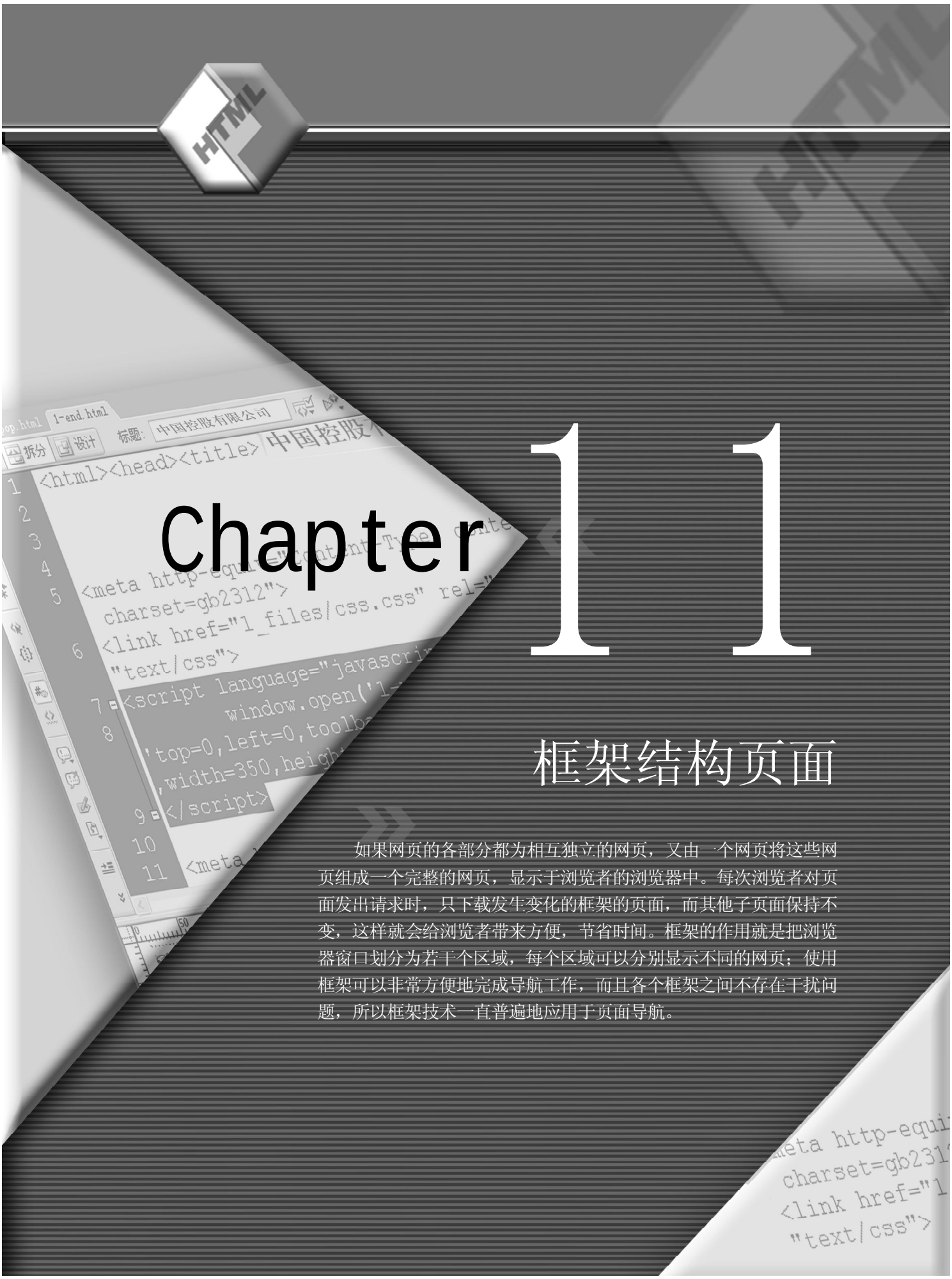


Chapter 5





附录



Chapter 11

框架结构页面

如果网页的各部分都为相互独立的网页，又由一个网页将这些网页组成一个完整的网页，显示于浏览者的浏览器中。每次浏览者对页面发出请求时，只下载发生变化的框架的页面，而其他子页面保持不变，这样就会给浏览者带来方便，节省时间。框架的作用就是把浏览器窗口划分为若干个区域，每个区域可以分别显示不同的网页；使用框架可以非常方便地完成导航工作，而且各个框架之间不存在干扰问题，所以框架技术一直普遍地应用于页面导航。

1 框架的整体结构

框架主要包括两个部分：一个是框架集，另一个就是框架。框架集是在一个文档内定义一组框架结构的HTML网页。框架集定义了在一个窗口中显示的框架数、框架的尺寸、载入到框架的网页等。而框架则是指在网页上定义的一个显示区域。要想创建一个框架文档，需要知道3个标签：<frameset>、<frame>和<noframes>，如下表所示。

标记	描述
<frameset>	框架集
<frame>	框架
<noframes>	无框架

Frameset标签

框架集（frameset）仅是一个框架的集合，而这些框架构成了浏览器的窗口。<frameset>标签的行（列）属性可以用来定义框架的行（列）数量以及初始大小。

基本语法	<frameset>	
	包含框架内容	
	</frameset>	
功能	定义框架集合	
属性及说明	属性	说明
	Class	用一个名称来标记框架集，该标记名称指向一个预定义的类，而该类是在文档级声明的或者在外部定义的样式表
	Id	为框架集创建一个标记，应用超链接时可以用这个标记来明确地引用该框架集，以便作为样式表选择器或使用其他应用程序执行自动搜索
	Style	创建框架集内容的内联样式
	Title	给框架集加上说明性的文字
	Rows	给框架集分行
	Cols	给框架集分列
	Bordercolor	框架集边框颜色
	Frameborder	框架集边框
	Framespacing	框架集间距

<frameset>标签可以用来定义框架和其他框架集的集合，并控制它们的间距和边框。框架集设置可以嵌套，并允许更复杂的布局功能设置。在框架文档中使用<frameset>标签来代替<body>标签。框架文档中不能包括除合法的<head>和<frameset>内容之外的其他任何内容。如果将包含<body>部分的传统文档与框架组合在一起，可能会导致不可预知的浏览结果。

Rows属性和Cols属性

<frameset>标签有一个必需的属性：要么是Rows，要么是Cols，它们定义了文档窗口中框架或者嵌套的框架集的行（或列）大小及数目。这两个属性都接受用引号括起来并用逗号分开的值列表，这些数值指定了框架的绝对（像素点）或相对（百分比或其它空间）宽度（对列而言），以及绝对或相对高度（对行而言）。这些属性值的数目决定了浏览器将会在文档窗口中显示多少行（或列）的框架。

与表格一样，浏览器在显示时会尽可能接近给定的框架集尺寸。但是，浏览器不会为了能够容下超出边沿的框架集而扩展主文档窗口的边界，也不会在指定的框架没有填满整个窗口时用空白区域来填满。相反，浏览器会根据一个框架相对于其他框架的大小来分配空间，这样就能够填满整个文档窗口了。

Frame 标签

框架文档不包含可以显示的内容，但有可能显示针对无框架显示支持的浏览器的提示消息。相反，一个或者多个<frameset>标签（这些标签封装了框架文本的内容）内的<frame>标签可以提供对单独文本的URL 引用，其中每个文本都占据了一个框架。

基本语法	<frame>	
	包含框架内容	
	</frame>	
功能	定义框架	
属性及说明	属性	说明
	Class	用一个名称来标记框架，该标记名称指向一个预定义的类，而该类是在文档级声明的或者在外部定义的样式表
	Id	为框架创建一个标记，应用超链接时可以用这个标记来明确地引用该框架，以便作为样式表选择器或使用其他应用程序来执行自动搜索
	Style	创建框架内容的内联样式
	Title	给框架加上说明性的文字
	Bordercolor	框架边框颜色
	Frameborder	框架边框
	Name	框架名称
	Noresize	禁止调整框架大小
	Src	框架源文件
	Marginheight	框架边缘高度
	Marginwidth	框架边缘宽度

<frame>标签只出现在<frameset>标签内。通过使用与它关联的Src属性，可以用它来设定文本内容的URL，这些文本内容从开始就显示在各个框架中。

浏览器将框架从左到右逐列（或从上到下逐行）地放置在一个框架集中，因此<frame>标签在<frameset>标签中的顺序和数目十分重要。浏览器将没有包含Src属性的<frame>标签显示为空框架。如果<frameset>标签要求的框架数超过了相应的<frame>标签定义的数目，也会显示为空框架。

Noframes标签

框架文档没有<body>标签，这是因为如果浏览器在遇到第一个<frameset>标签之前发现了任何<body>内容，它将会忽略任何框架标签。除了那些对任何无框架能力的浏览器都不可见的文件之外，框架文档可以是任何内容。<noframes>标签在一定程度上只能在框架文档的最外层的<frameset>标签内部使用。在<noframes>内部的内容以及它的结束标签（</noframes>）不能被任何支持框架能力的浏览器所显示，但会被那些不能处理框架的浏览器显示。<noframes>标签的内容可以是任何普通的主体内容，包括<body>标签。

基本语法	<noframes>	
	包含无框架内容	
	</noframes>	
功能	为不能显示框架的浏览器提供内容	
属性及说明	属性	说明
	Class	用一个名称来标记框架，该标记名称指向一个预定义的类，而该类是在文档级声明的或者在外部定义的样式表
	Id	为框架创建一个标记，应用超链接时可以用这个标记来明确地引用该框架，以便作为样式表选择器或使用其他应用程序来执行自动搜索
	Style	创建框架内容的内联样式
	Title	给框架加上说明性的文字
	Dir	文本方向
	Lang	语言信息

原始文件	Sample\Ch11\1\1.html
最终文件	Sample\Ch11\1\1-end.html
学习要点	使用<frameset>、<frame>和<noframes>标签制作纵向框架

1. 范例代码

```
<html>
<head>
-
</head>
<frameset cols="80,*">
<frame>
<frame>
</frameset>
<noframes>
<body>请使用支持框架的浏览器</body>
</noframes>
</html>
```

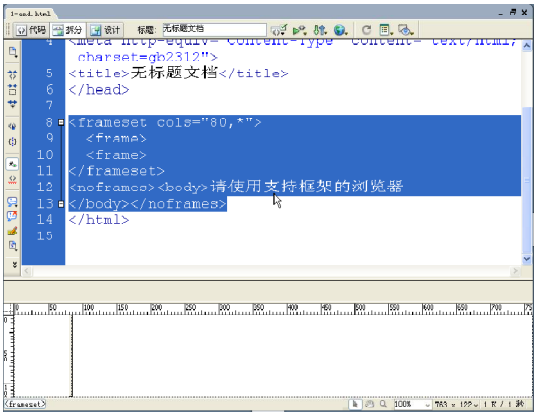
2. 范例解释

01 将窗口分割成左、右两个部分，它们的宽度分别为 80 和剩下的所有宽度。

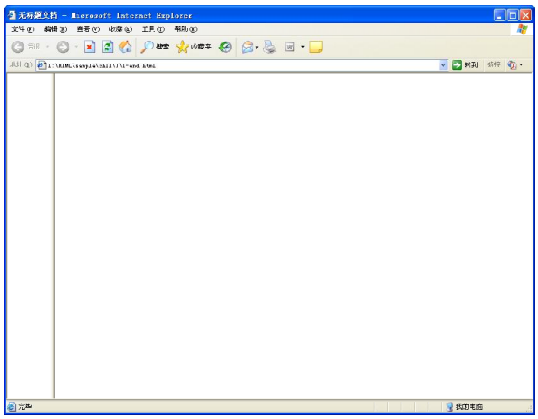
02 对应上述两个窗口。

03 声明不支持框架的浏览器的显示内容。

3. 代码和设计



4. 显示结果



原始文件	Sample\Ch11\1\2.html
最终文件	Sample\Ch11\1\2-end.html
学习要点	使用<frameset>、<frame>和<noframes>标签制作横向框架

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="80,*,80">
  <frame>
  <frame>
  <frame>
</frameset>
<noframes>
<body>请使用支持框架的浏览器</body>
</noframes>
</html>
```

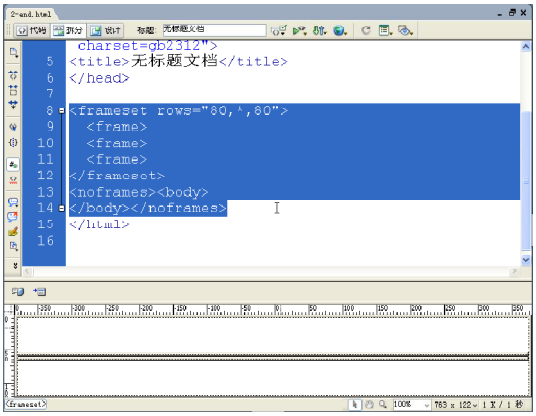
2. 范例解释

01 将窗口分割成上、中、下 3 个部分，它们的高度分别为 80、剩下的高度、80。

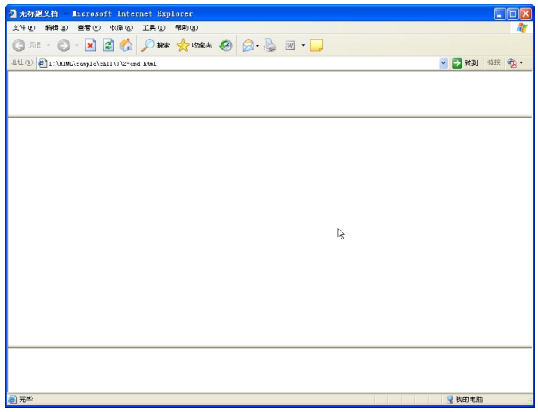
02 对应上述 3 个窗口。

03 声明不支持框架的浏览器的显示内容。

3. 代码和设计



4. 显示结果



原始文件	Sample\Ch11\1\3.html
最终文件	Sample\Ch11\1\3-end.html
学习要点	使用<frameset>、<frame>和<noframes>标签制作嵌套框架

1. 范例代码

```
<html>
<head>
…
</head>
<frameset cols="80,*">
<frame>
<frameset rows="80,*">
<frame>
<frame>
</frameset>
</frameset>
<noframes>
<body>请使用支持框架的浏览器</body>
</noframes>
</html>
```

2. 范例解释

01 将窗口分割成左、右两个部分，它们的宽度分别为80、剩下的所有宽度。

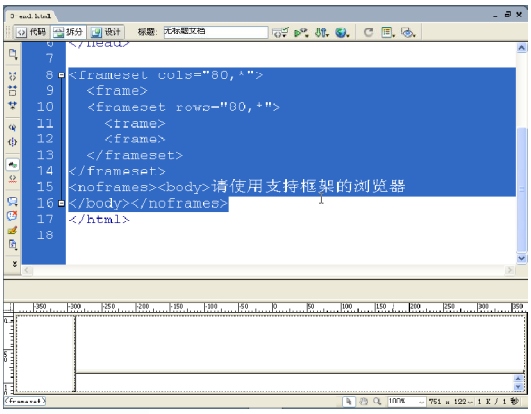
02 对应左侧的窗口。

03 将右侧的窗口分割成上、下两个部分，它们的高度分别为80、剩下的所有高度。

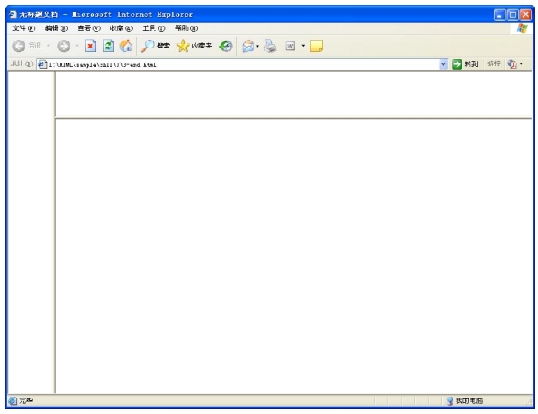
04 对应右侧上、下两个窗口。

05 声明不支持框架的浏览器的显示内容。

3. 代码和设计



4. 显示结果



2 框架的属性

架框集和框架都有很多自身的属性，正确设置这些属性能够保证框架集和框架在浏览器中正常显示。

Frame 标签的属性

Src属性

<frame>标签所使用的 Src 属性值是要显示在框架中的文档的 URL。除此之外，没有其他办法来为框架提供内容。例如，不能在这个框架文档中包含任何<body>的内容，如果<body>标签先出现，浏览器将会忽略掉框架标签，并且只显示这个<body>标签的内容，反之亦然。Src 属性引用的文本可以是任何合法的文档或者可以显示的对象，包括图像和多媒体。引用的文档本身可以由一个或者多个框架构成。这些框架显示在引用框架中，这也提供了用嵌套框架实现复杂布局的另一种途径。

原始文件	Sample\Ch11\2\1.html
最终文件	Sample\Ch11\2\1-end.html
学习要点	使用<frame>标签的src属性

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="70,202,39">
<frame src="1_files/topframe.htm">
<frameset rows="*" cols="174,*">
<frame
src="1_files/leftframe.htm">
<frame
 src="1_files/mainframe.htm">
</frameset>
<frame
src="1_files/bottomframe.htm">
</frameset>
<noframes>
<body>
</body>很抱歉，您使用的浏览器不支持框架功能，请转用新的浏览器。
</noframes>
</html>
```

2. 范例解释

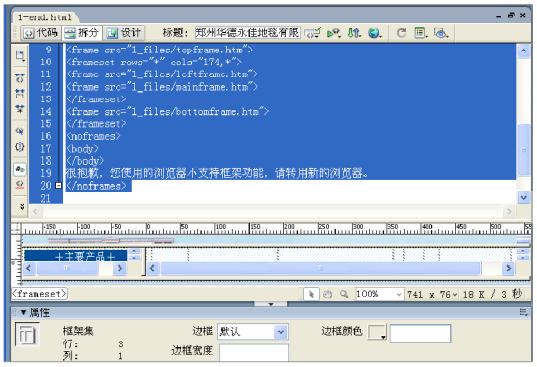
01 声明第1行的框架内容为1_files文件夹下的topframe.htm文件。

02 声明第2行左侧的框架内容为1_files文件夹下的leftframe.htm文件。

03 声明第2行右侧的框架内容为1_files文件夹下的mainframe.htm文件。

04 声明第3行的框架内容为1_files文件夹下的bottomframe.htm文件。

3. 代码和设计



4. 显示结果

Frameborder属性

可以用Frameborder 属性对一个单一的框架添加或者删除边框。值“yes”（“1”）或“no”（“0 ”）分别用来激活或者禁用框架的边框，并且不用考虑任何包含这个框架的框架集的Frameborder 属性值。Frameborder 属性值如下表所示。

Frameborder属性值及说明	Frameborder属性值	说明
	Yes(1)	出现边框
	No(0)	不出现边框

原始文件	Sample\Ch11\2\2.html
最终文件	Sample\Ch11\2\2-end.html
学习要点	使用<frame>标签的frameborder属性

1. 范例代码

```
<html>
<head>
-
</head>
<frameset  rows="70,202,39">
<frame  src="1_files/topframe.
htm"   frameborder=0>
<frameset  rows="*"  cols="174,*">
<frame  src="1_files/leftframe.
htm"   frameborder=0>
<frame  src="1_files/mainframe.
htm"   frameborder=0>
</frameset>
<frame  src="1_files/bottomframe.
htm"   frameborder=0>
</frameset><noframes><body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
</noframes>  </html>
```

2. 范例解释

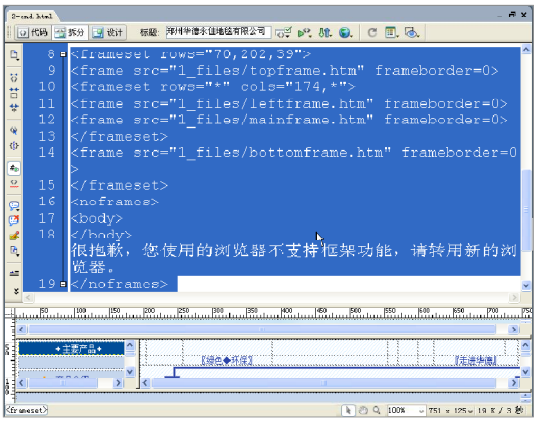
01 声明第1行的框架内容为1_files文件夹下的topframe.htm文件，框架边框为0。

02 声明第2行左侧的框架内容为1_files文件夹下的leftframe.htm文件，框架边框为0。

03 声明第2行右侧的框架内容为1_files文件夹下的mainframe.htm文件，框架边框为0。

04 声明第3行的框架内容为1_files文件夹下的bottomframe.htm文件，框架边框为0。

3. 代码和设计



4. 显示结果

Scrolling属性

对那些内容超出所分配窗口空间的框架，浏览器将会显示垂直或水平滚动条。否则这些滚动条不会出现。<frame>标签的 Scrolling 属性使用户能够控制滚动条的出现和消失。如果设置 Scrolling="yes"，浏览器就会给这个指定的框架添加滚动条(即使没有什么内容需要拖动滚动条进行浏览)； 如果设定 Scrolling 属性值为 no，将不会添加滚动条（即使是框架中的内容大于框架本身)。如果将 Scrolling 的值设定为 Auto，浏览器将根据需要添加滚动条，具体属性值如下表所示：

Scrolling属性值及说明	Scrolling属性值	说明
	Yes	出现滚动条
	No	不出现滚动条
	Auto	自动出现滚动条

原始文件	Sample\Ch11\2\3.html
最终文件	Sample\Ch11\2\3-end.html
学习要点	使用<frame>标签的scrolling属性

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="70,202,39">
<frame src="1_files/topframe.htm"
frameborder=0 scrolling="no">
<frameset rows="*" cols="174,*">
<frame src="1_files/leftframe.htm"
frameborder=0 scrolling="no">
<frame src="1_files/mainframe.htm"
frameborder=0 scrolling="auto">
</frameset>
<frame src="1_files/bottomframe.htm"
frameborder=0scrolling="no">
```

2. 范例解释

01 声明第 1 行的框架内容为 1_files 文件夹下的 topframe.htm 文件，框架边框为 0，不显示滚动条。

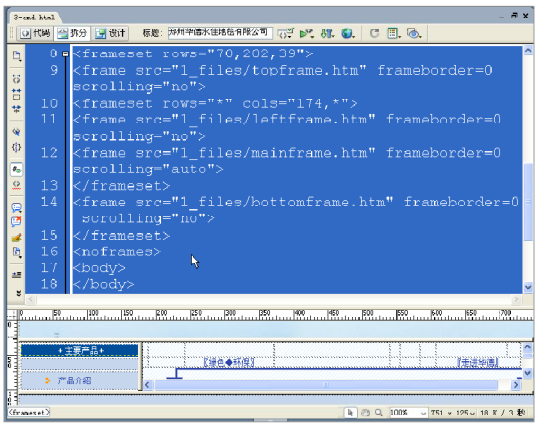
02 声明第2行左侧的框架内容为1_files文件夹下的 leftframe.htm 文件，框架边框为 0，不显示滚动条。

03 声明第2行右侧的框架内容为1_files文件夹下的 mainframe.htm 文件，框架边框为 0，自动显示滚动条。

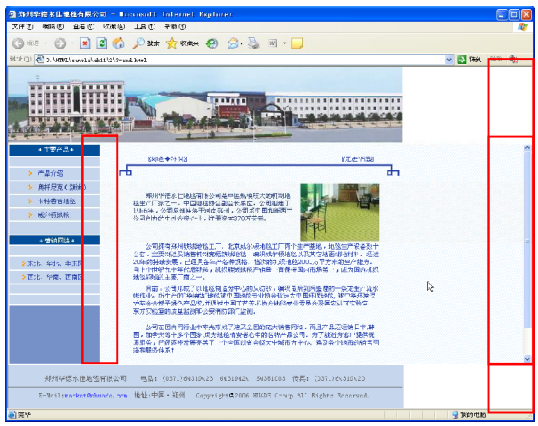
04 声明第 3 行的框架内容为 1_files 文件夹下的 bottomframe.htm 文件，框架边框为 0，不显示滚动条。

```
</frameset><noframes><body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
<noframes>
</html>
```

3. 代码和设计



4. 显示结果



Noresize属性

即便可以在<frameset>标签中用相应的属性显示设定框架的尺寸，用户还是能够手动改变框架中一行（或者一列）的大小。对于那些不希望用户随意修改其相对尺寸的行（或者列）来说，可将Noresize属性添加到它们的框架标签中。通过固定框架的大小以使该框架只包含图像并设置Noresize属性，就能够保证图像以所希望的方式显示出来，并且浏览器窗口的其余部分将总是显示文档中的其他框架。

原始文件	Sample\Ch11\2\4.html
最终文件	Sample\Ch11\2\4-end.html
学习要点	使用<frame>标签的noresize属性

1. 范例代码

```
<html>
<head>
…
</head>
<frameset rows="70,202,39">
<framesrc="1_files/topframe.htm"
frameborder=0 scrolling="no" noresize>
<frameset rows="*" cols="174,*">
<framesrc="1_files/leftframehtm"
frameborder=0 scrolling="no" noresize>
<framesrc="1_files/mainframehtm"
frameborder=0 scrolling="auto" noresize>
```

2. 范例解释

01 声明第1行的框架内容为1_files文件夹下的topframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小。

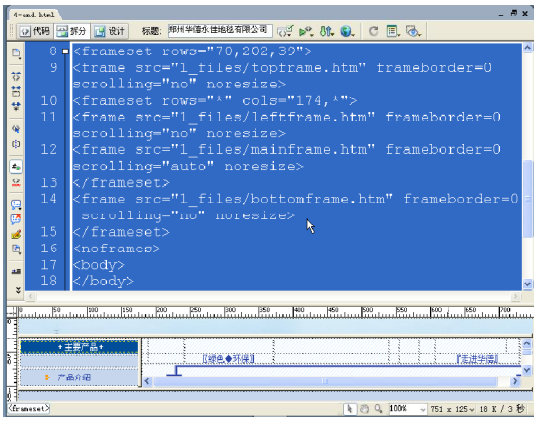
02 声明第2行左侧的框架内容为1_files文件夹下的leftframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小。

03 声明第2行右侧的框架内容为1_files文件夹下的mainframe.htm文件，框架边框为0，自动显示滚动条，禁止改变大小。

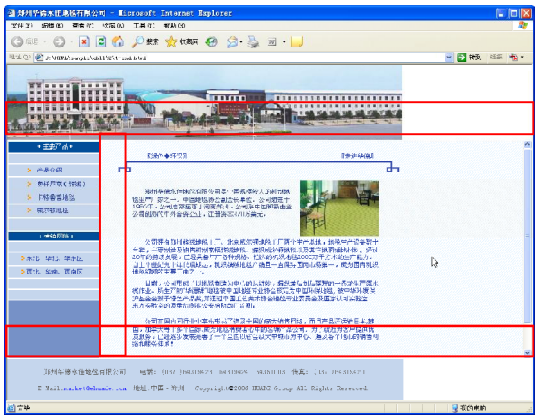
```
</frameset>
<framesrc="1_files/bottomframe.htm"
frameborder=0 scrolling="no" noresize>
</frameset><noframes><body>
</body>很抱歉，您使用的浏览器不支持框架功能，请转用新的浏览器。
</noframes>
</html>
```

04 声明第3行的框架内容为1_files文件夹下的bottomframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小。

3. 代码和设计



4. 显示结果



Marginheight和Marginwidth属性

浏览器通常在框架的边沿和其内容之间留下一小部分间隔。可以用Marginheight和Marginwidth属性修改这个边界的大小，每个属性都可以设置包围在框架内容周围的像素点的数量。

原始文件	Sample\Ch11\2\5.html
最终文件	Sample\Ch11\2\5-end.html
学习要点	使用<frame>标签的marginheight和marginwidth属性

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="70,202,39">
<framesrc="1_files/topframe.htm"
frameborder=0 scrolling="no"
noresize marginheight="0"
marginwidth="0">
<frameset rows="*" cols="174,*">
<framesrc="1_files/leftframe.htm"
frameborder=0 scrolling="no"
noresize marginheight="0"
marginwidth="0">
```

2. 范例解释

01 声明第1行的框架内容为1_files文件夹下的topframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0。

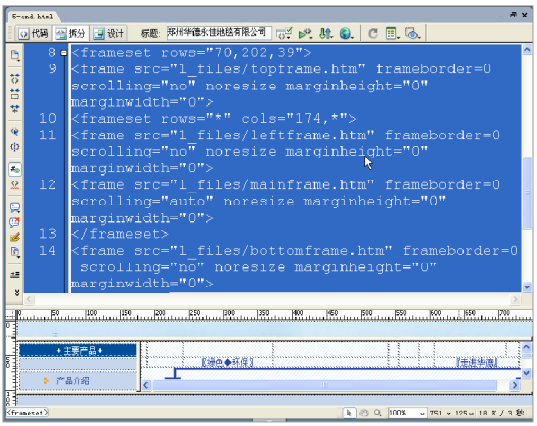
02 声明第2行左侧的框架内容为1_files文件夹下的leftframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0。

```
<frame src="1_files/mainframe.htm"
frameborder=0 scrolling="auto"
noresize marginheight="0"
marginwidth="0">
</frameset>
<frame src="1_files/bottomframe.htm"
frameborder=0 scrolling="no"
noresize marginheight="0"
marginwidth="0">
</frameset>
<noframes>
<body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
</noframes>
</html>
```

03 声明第2行右侧的框架内容为1_files文件夹下的mainframe.htm文件，框架边框为0，自动显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0。

04 声明第3行的框架内容为1_files文件夹下的bottomframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0。

3. 代码和设计



4. 显示结果

Name属性

<frame>标签中可选的Name 属性可以对该框架进行标记，以便以后被用于超文本链接锚（<a>）标签的Target 属性所引用。如果使用这种方法，就可以用另一个框架中的链接改变某个框架的内容，反之则像通常的浏览器窗口一样，用超文本链接的文档取代这个源框架中的内容。

如果在一个<a>标签内包含一个Target属性，浏览器将会载入和显示用这个标签的Href属性命名的、名称与这个目标吻合的框架或者窗口中的文档。如果这个指定名称的框架或者窗口不存在，浏览器将打开一个新的窗口，给这个窗口一个指定的标记，然后将新的文档载入那个窗口。从此以后，超链接文档就可以指向这个新的窗口。

原始文件	Sample\Ch11\2\6.html、1_files\leftframe.htm
最终文件	Sample\Ch11\2\6-end.html、1_files\leftframe-end.htm
学习要点	使用<frame>标签的name属性

6-end.html

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="70,202,39">
<frame src="1_files/topframe.htm" frameborder=0 scrolling="no" noresize marginheight="0" marginwidth="0" name="top">
<frameset rows="*" cols="174,*">
<frame src="1_files/leftframe-end.htm" frameborder=0 scrolling="no" noresize marginheight="0" marginwidth="0" name="left">
<frame src="1_files/mainframe.htm" frameborder=0 scrolling="auto" noresize marginheight="0" marginwidth="0" name="main">
</frameset>
<frame src="1_files/bottomframe.htm" frameborder=0 scrolling="no" noresize marginheight="0" marginwidth="0" name="bottom">
</frameset>
<noframes>
<body>
</body>很抱歉，您使用的浏览器不支持框架功能，请转用新的浏览器。
</noframes>
</html>
```

2. 范例解释

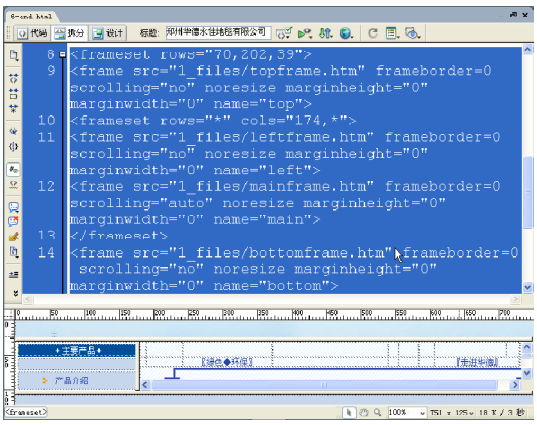
01 声明第1行的框架内容为1_files文件夹下的topframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0，框架名称为top。

02 声明第2行左侧的框架内容为1_files文件夹下的leftframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0，框架名称为left。

03 声明第2行右侧的框架内容为1_files文件夹下的mainframe.htm文件，框架边框为0，自动显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0，框架名称为main。

04 声明第3行的框架内容为1_files文件夹下的bottomframe.htm文件，框架边框为0，不显示滚动条，禁止改变大小，框架边缘高度和边缘宽度为0，框架名称为bottom。

3. 代码和设计



4. 显示结果

1_files\leftframe-end.htm

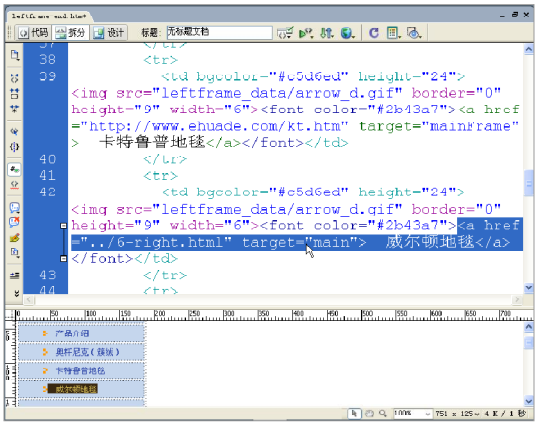
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<a href="../6-right.html"
target="main">威尔顿地毯</a>
…
</body>
</html>
```

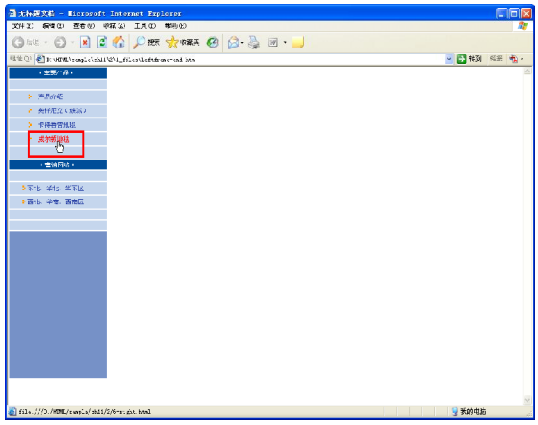
2. 范例解释

为文字“威尔顿地毯”声明到上级目录下6-right.html页面的链接，并在名为main的目标窗口中打开。

3. 代码和设计



4. 显示结果



6-end.html最终显示结果

Frameset 标签的属性

Framespacing属性

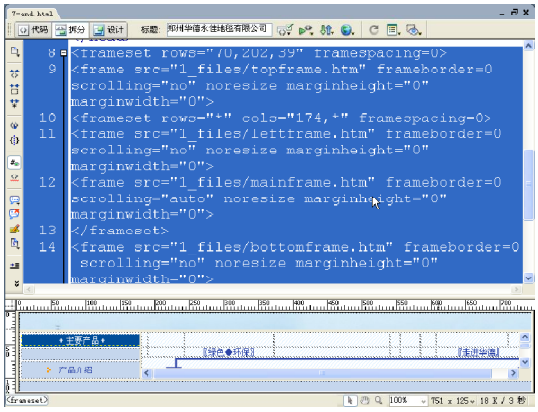
使用 Framespacing 属性能够设定框架集的边框宽度。

原始文件	Sample\Ch11\2\7.html
最终文件	Sample\Ch11\2\7-end.html
学习要点	使用<frameset>标签的framespacing属性

1. 范例代码与解释

```
<html>
<head>
-
</head>
<frameset rows="70,202,39"
framespacing=0>
<frame src="1_files/topframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frameset rows="*" cols="174,*"
framespacing=0>
<frame src="1_files/leftframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frame src="1_files/mainframe.
htm" frameborder=0
scrolling="auto" noresize
marginheight="0"
marginwidth="0">
</frameset>
<frame src="1_files/bottomframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
</frameset>
<noframes><body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
</noframes>
</html>
```

2. 代码和设计



3. 显示结果

❶ 声明分为 3 行的框架集的框架边框为 0。

❷ 声明分为 2 列的框架集的框架边框为 0。

Frameborder属性

使用 Frameborder 属性能够设定框架集的是否出现边框。

原始文件	Sample\Ch11\2\8.html
最终文件	Sample\Ch11\2\8-end.html
学习要点	使用<frameset>标签的frameborder属性

1. 范例代码

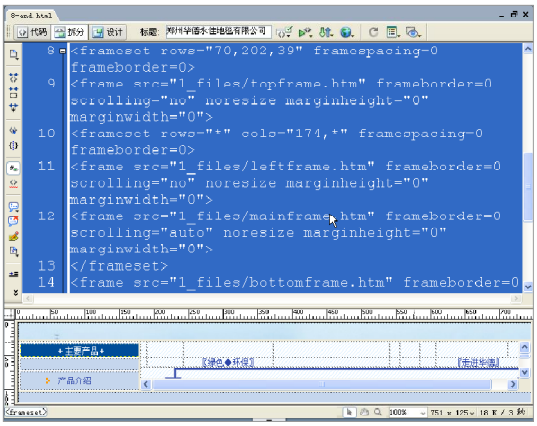
```
<html>
<head>
…
</head>
<frameset rows="70,202,39"
framespacing=0 frameborder=0>
<frame src="1_files/topframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frameset rows="*" cols="174,*"
framespacing=0 frameborder=0>
<frame src="1_files/leftframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frame src="1_files/mainframe.
htm" frameborder=0
scrolling="auto" noresize
marginheight="0"
marginwidth="0">
</frameset>
<frame src="1_files/bottomframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
</frameset>
<noframes>
<body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
</noframes>
</html>
```

2. 范例解释

01 声明分为 3 行的框架集不出现框架边框。

02 声明分为 2 列的框架集不出现框架边框。

3. 代码和设计



4. 显示结果

Bordercolor属性

使用Bordercolor属性能够设定框架集边框的颜色。

原始文件	Sample\Ch11\2\9.html
最终文件	Sample\Ch11\2\9-end.html
学习要点	使用<frameset>标签的bordercolor属性

1. 范例代码

```
<html>
<head>
-
</head>
<frameset rows="70,202,39"
frameborder=1 bordercolor=#336699>
<frame src="1_files/topframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frameset rows="*" cols="174,*"
frameborder=1 bordercolor=#336699>
<frame src="1_files/leftframe.
htm" frameborder=0
scrolling="no" noresize
marginheight="0"
marginwidth="0">
<frame src="1_files/mainframe.
htm" frameborder=0
scrolling="auto" noresize
marginheight="0"
marginwidth="0">
```

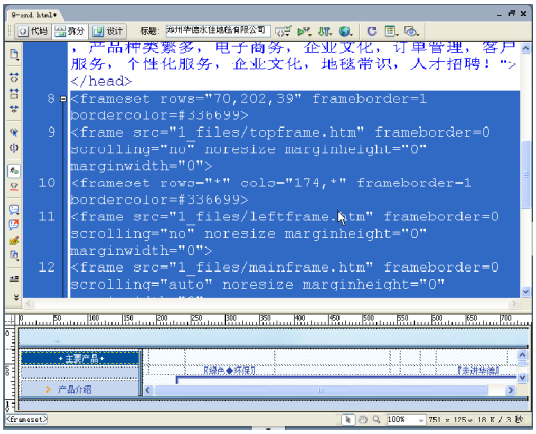
2. 范例解释

01 声明分为 3 行的框架集边框颜色为 #336699。

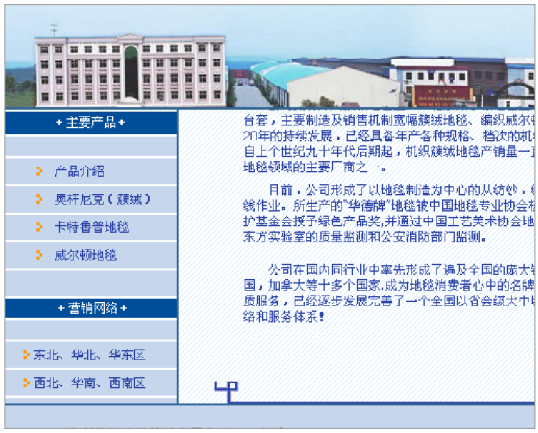
02 声明分为 2 列的框架集边框颜色为 #336699。

```
</frameset>
<frame src="1_files/bottomframe.
htm"  frameborder=0
scrolling="no"  noresize
marginheight="0"
marginwidth="0">
</frameset>
<noframes>
<body>
</body>很抱歉，您使用的浏览器不支持
框架功能，请转用新的浏览器。
</noframes>
</html>
```

3. 代码和设计



4. 显示结果



3 制作内联框架

框架集取代了传统文档的<body>，并且通过它包含的框架为用户提供了内容。HTML 4 标准允许定义存在于一个传统文档中的框架，显示成为这个文档的文本流的一部分。用<iframe>标签可以定义一个内联框架，该标签不是应用在<frameset>标签内相反它可以出现在文档中任何地方。<iframe>标签在文档中定义了一个矩形的区域，在这个区域中，浏览器会显示一个单独的文档，包括滚动条和边框。

Iframe标签

基本语法	<iframe>
	包含内联框架内容
	</iframe>
功能	定义内联框架

（续表）

	属性	说明
属性及说明	Class	用一个名称来标记内联框架，该标记名称指向一个预定义的类，而该类是在文档级声明的或者在外部定义的样式表
	Id	为内联框架创建一个标记，应用超链接时可以用这个标记来明确地引用该段落，以便作为样式表选择器或使用其他应用程序来执行自动搜索
	Style	创建内联框架内容的内联样式
	Title	给内联框架加上说明性的文字
	Frameborder	内联框架边框
	Name	内联框架名称
	Src	内联框架源文件
	Marginheight	内联框架边缘高度
	Marginwidth	内联框架边缘宽度
	Align	对齐方式
	Width	内联框架宽度
	Height	内联框架高度
	Scrolling	是否允许出现滚动条

<iframe>标签的内容可以用来为不支持内联框架的浏览器用户提供信息：支持内联框架的浏览器将忽略<iframe>标签的内容，而不支持内联框架的浏览器将忽略这个<iframe>标签，并且显示它的内容，就好像它是普通的文本内容一样。

原始文件	Sample\Ch11\3\1.html
最终文件	Sample\Ch11\3\1-end.html
学习要点	使用<iframe>标签

1. 范例代码

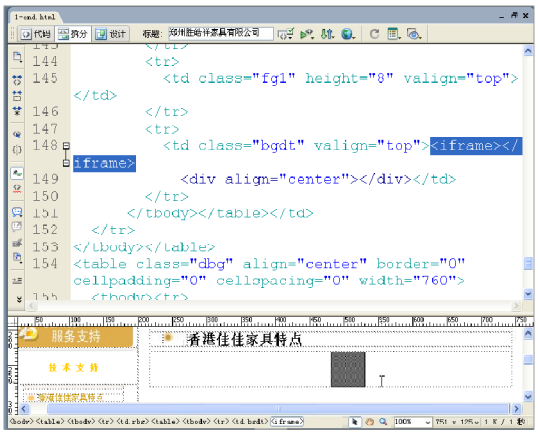
```
<html>
<head>
-
</head>
<body>
-
<iframe>
</iframe>
-
</body>
</html>
```

2. 范例解释

01 声明内联框架开始。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果

Iframe标签的属性

Src属性

<iframe>标签的Src属性值是显示在内联框架中的文档的URL。

原始文件	Sample\Ch11\3\2.html
最终文件	Sample\Ch11\3\2-end.html
学习要点	使用<iframe>标签的Src属性

1. 范例代码

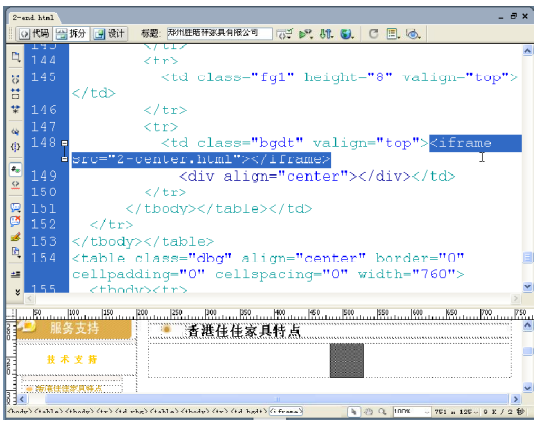
```
<html>
<head>
…
</head>
<body>
…
<iframe src="2-center.html">
</iframe>
…
</body>
</html>
```

2. 范例解释

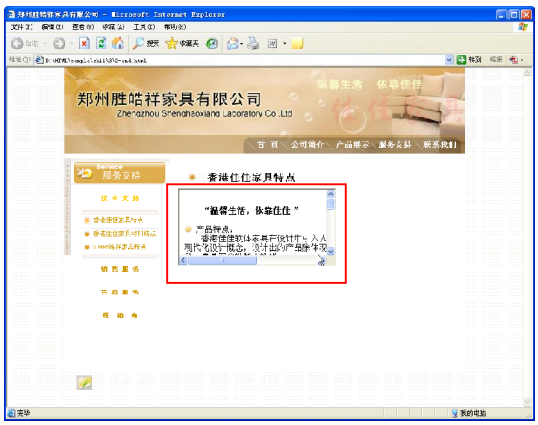
01 声明内联框架开始，载入2-center.html文件。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果



Width和Height属性

<iframe>标签的Width和Height属性可以设置内联框架显示的宽度和高度。

原始文件	Sample\Ch11\3\3.html
最终文件	Sample\Ch11\3\3-end.html
学习要点	使用<iframe>标签的width和height属性

1. 范例代码

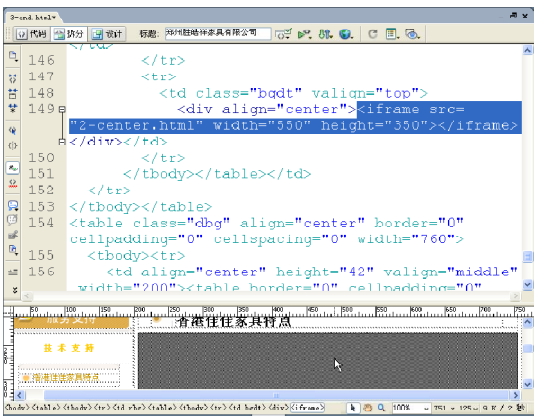
```
<html>
<head>
-
</head>
<body>
-
<iframe src="2-center.html"
  width="550" height="350">
</iframe>
-
</body>
</html>
```

2. 范例解释

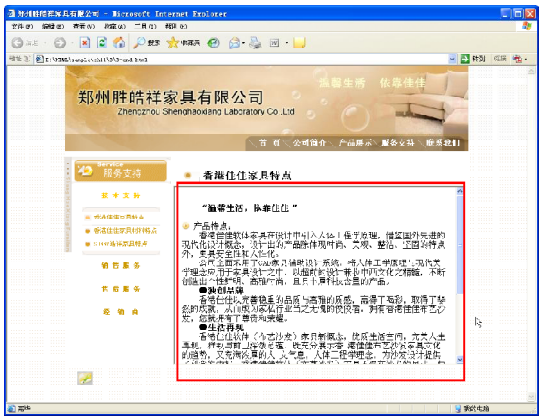
01 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果



FrameBorder属性

使用FrameBorder属性可以控制内联框架边框是否显示。

原始文件	Sample\Ch11\3\4.html
最终文件	Sample\Ch11\3\4-end.html
学习要点	使用<iframe>标签的frameBorder属性

1. 范例代码

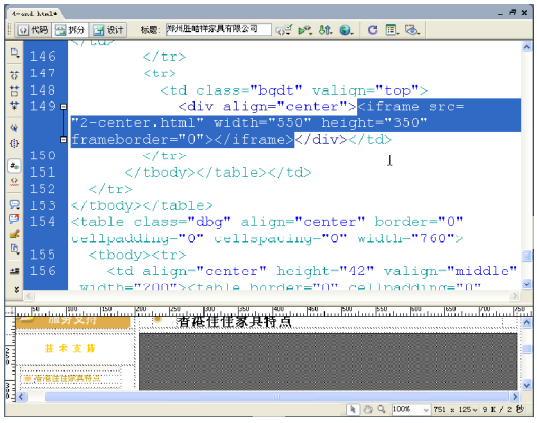
```
<html>
<head>
…
</head>
<body>
…
<iframe src="2-center.html"
width="550" height="350"
frameborder="0">
</iframe>
…
</body>
</html>
```

2. 范例解释

01 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素，框架边框为0。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果

Scrolling属性

当内联框架内的空间不够显示页面的内容时，可以通过滚动条来实现页面的滚动，以便用户查看隐藏的内容。Scrolling属性可以设定滚动条是否显示，它的属性值如下表所示。

Scrolling属性值及说明	Scrolling属性值	说明
	Yes	出现滚动条
	No	不出现滚动条
	Auto	自动出现滚动条

原始文件	Sample\Ch11\3\5.html
最终文件	Sample\Ch11\3\5-end.html
学习要点	使用<iframe>标签的scrolling属性

1. 范例代码

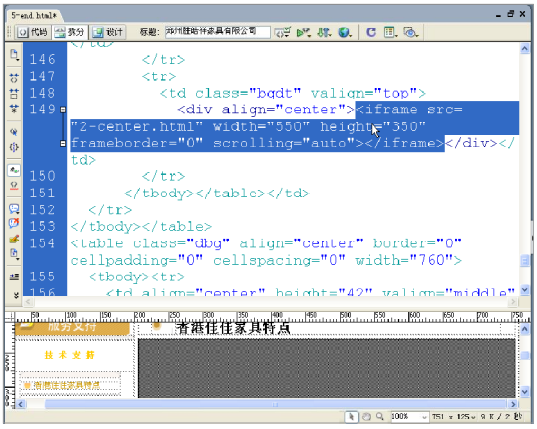
```
<html>
<head>
-
</head>
<body>
-
<iframe src="2-center.html"
width="550" height="350"
frameborder="0"
scrolling="auto">
</iframe>
-
</body>
</html>
```

2. 范例解释

01 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素，框架边框为0，滚动条为自动显示。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果

Marginheight和Marginwidth属性

浏览器通常在内联框架的边沿和其内容之间留下一小部分间隔。可以使用Marginheight和Marginwidth属性来修改这个边界的大小，这两个属性都可以设置包括在内联框架内容周围的像素点的数量。

原始文件	Sample\Ch11\3\6.html
最终文件	Sample\Ch11\3\6-end.html
学习要点	使用<iframe>标签的marginheight和marginwidth属性

1. 范例代码

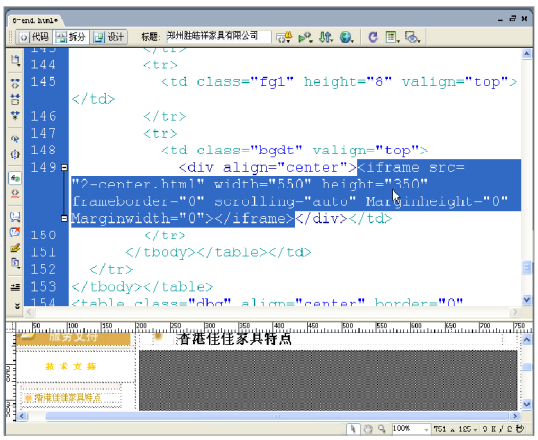
```
<html>
<head>
…
</head>
<body>
…
<iframe src="2-center.html"
width="550" height="350"
 frameborder="0"
scrolling="auto"
Marginheight="0"
Marginwidth="0">
</iframe>
…
</body>
</html>
```

2. 范例解释

01 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素，框架边框为0，滚动条为自动显示，框架边缘宽度和边缘高度都为0。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果

Align属性

使用Align属性可以设置内联框架的对齐方式：Left表示居左，Center表示居中，Right表示居右。它的属性值如下表所示。

Align属性值及说明	Align属性值	说明
	Left	居左
	Center	居中
	Right	居右

原始文件	Sample\Ch11\3\7.html
最终文件	Sample\Ch11\3\7-end.html
学习要点	使用<iframe>标签的align属性

1. 范例代码

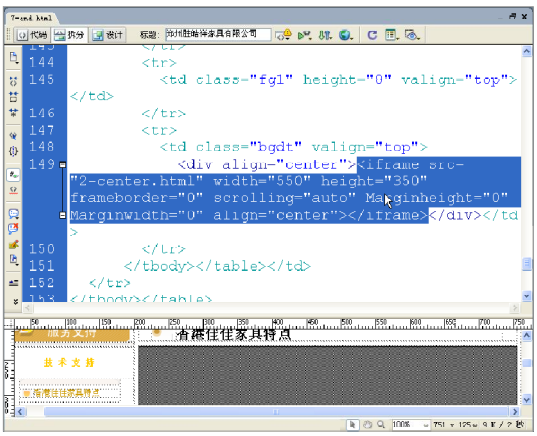
```
<html>
<head>
-
</head>
<body>
-
<iframe src="2-center.html"
width="550" height="350"
frameborder="0" scrolling="auto"
Marginheight="0" Marginwidth="0"
align="center">
</iframe>
-
</body>
</html>
```

2. 范例解释

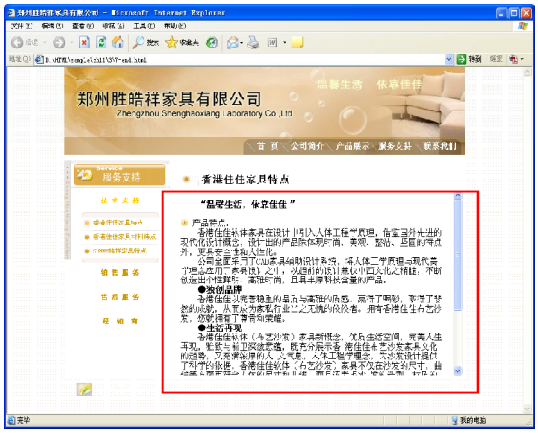
01 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素，框架边框为0，滚动条为自动显示，框架边缘宽度和边缘高度都为0，水平居中。

02 声明内联框架结束。

3. 代码和设计



4. 显示结果



Name属性

<iframe>标签中可选的Name属性可以对该内联框架进行标记，以便被用于超文本链接锚（<a>）标签的Target属性所引用。如果使用这种方法，就可以用另一个框架中的链接改变这个内联框架的内容。

原始文件	Sample\Ch11\3\8.html、8-center.html
最终文件	Sample\Ch11\3\8-end.html
学习要点	使用<iframe>标签的name属性

1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<a href="2-center.html"
target="content">
香港佳佳家具特点
</a>
…
<a href="8-center.html"
target="content" class="wzll">
香港佳佳家具材料特点
</a>
…
<iframe src="2-center.html"
width="550" height="350"
frameborder="0" scrolling="auto"
Marginheight="0" Marginwidth="0"
align="center" name="content">
</iframe>
…
</body>
</html>
```

2. 范例解释

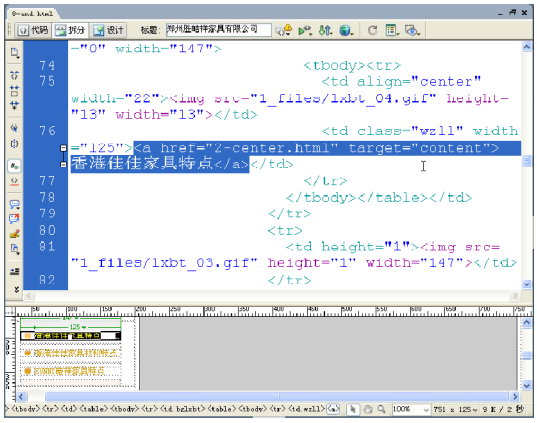
01 为“香港佳佳家具特点”文字制作到2-center.html文件的链接，设置目标窗口为content。

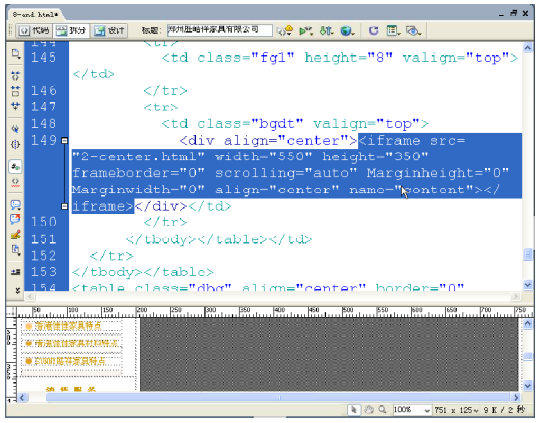
02 为“香港佳佳家具材料特点”文字制作到8-center.html文件的链接，设置目标窗口为content。

03 声明内联框架开始，载入2-center.html文件，宽度为550像素，高度为350像素，框架边框为0，滚动条为自动显示，框架边缘宽度和边缘高度都为0，水平居中，并设置名称为content。

04 声明内联框架结束。

3. 代码和设计



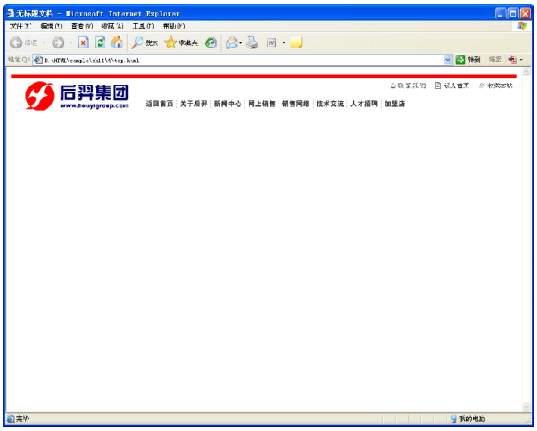


4. 显示结果

4 实例制作

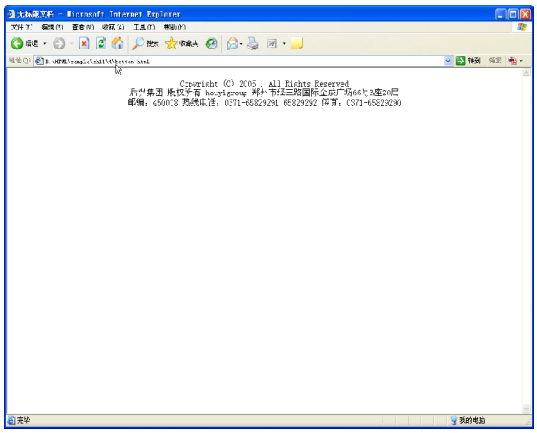
下面通过一个实例练习使用框架搭建页面结构的方法。读者可在编写该实例的过程中体会如何使用框架标签并设置框架各种属性。编写前先准备三个文件，它们分别是top.html、middle.html和bottom.html文件。

原始文件	Sample\Ch11\4\top.html、middle.html、bottom.html
最终文件	Sample\Ch11\4\1.html
学习要点	使用框架标签及其相关属性



原始效果top.html

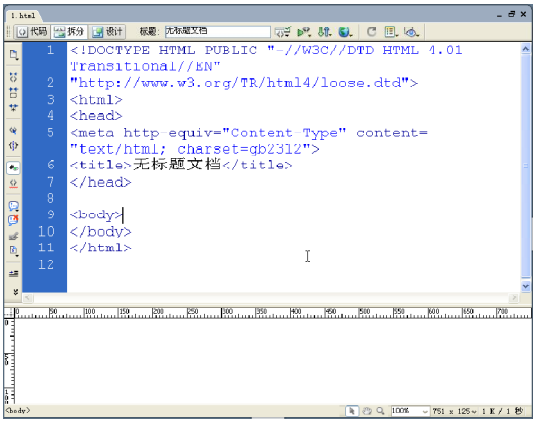
原始效果middle.html



原始效果bottom.html

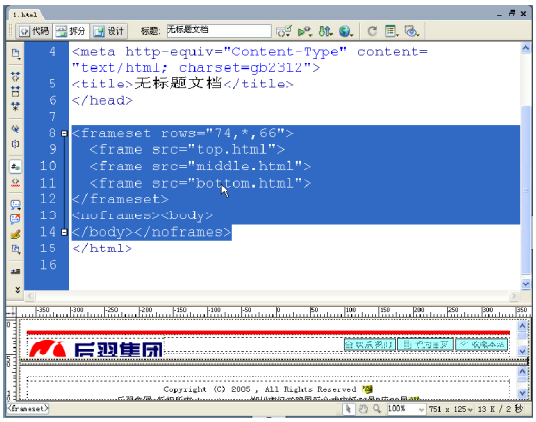
最终效果

1 在Dreamweaver中建立新页面，单击“拆分”按钮，切换到“代码视图和设计视图”环境。



2 删除<body></body>标记，在代码视图中插入以下代码：

```
<frameset rows="74,*,66">          ❶
<frame src="top.html">
<frame src="middle.html">          ❷
<frame src="bottom.html">
</frameset>
<noframes>
<body></body>                      ❸
</noframes>
```



❶ 声明3行的框架，高度分别为74、剩余的高度、66。

❷ 对应这3个窗口，并分别载入top.html、middle.html、bottom.html。

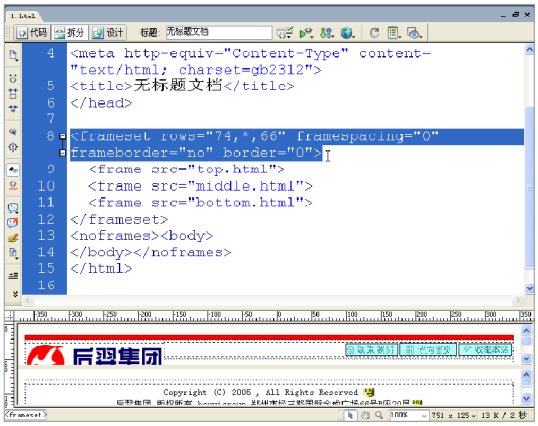
❸ 声明不支持框架的浏览器的显示内容。

3 此时按下 F12 快捷键预览页面，可以看到框架中已经载入了 top.html、middle.html 和 bottom.html页面内容。

4 选择<frameset>标签，并设置框架边框的间距均为0，可在代码视图中修改为如下代码：

```
<frameset rows="74,*,66"
framespacing="0"
frameborder="no" border="0">
```

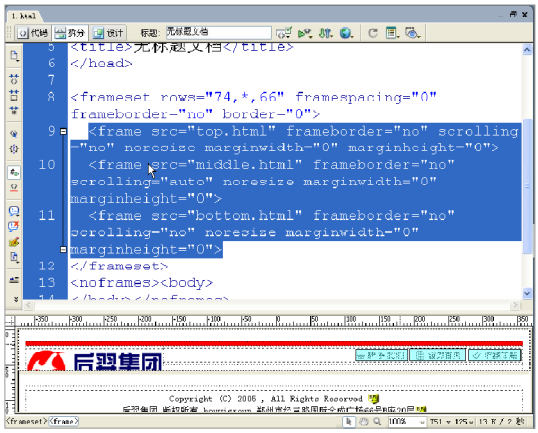
❶ 声明框架边框和间距都为 0。



5 此时按下 F12 快捷键预览页面，可以看到框架集中的边框已经被去除。

6 选择<frame>标签，在代码视图中修改为如下代码：

```
<frame src="top.html" frameborder="no" scrolling="no" noresize marginwidth="0" marginheight="0">
<frame src="middle.html" frameborder="no" scrolling="auto" noresize marginwidth="0" marginheight="0">
<frame src="bottom.html" frameborder="no" scrolling="no" noresize marginwidth="0" marginheight="0">
```

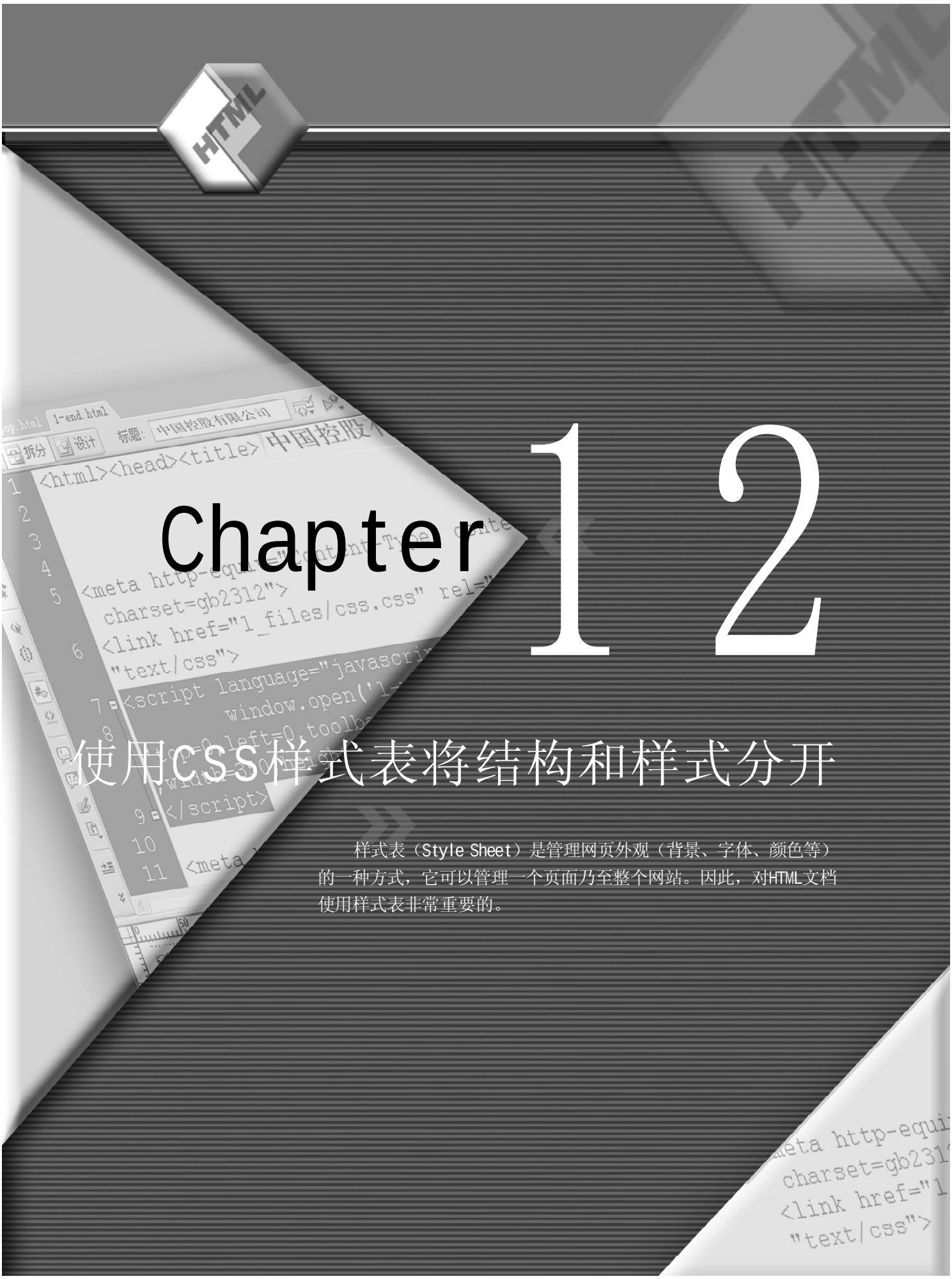


❶ 设置框架没有边框，没有滚动条，边缘宽度和边缘高度都为0，禁止改变框架大小。

❷ 设置框架没有边框，自动出现滚动条，边缘宽度和边缘高度都为0，禁止改变框架大小。

❸ 设置框架没有边框，没有滚动条，边缘宽度和边缘高度都为0，禁止改变框架大小。

7 按下F12快捷键预览页面，可以看到制作完成的框架网页效果。



Chapter 12

使用css样式表将结构和样式分开

样式表（Style Sheet）是管理网页外观（背景、字体、颜色等）的一种方式，它可以管理一个页面乃至整个网站。因此，对HTML文档使用样式表非常重要的。

1 什么是样式表

从一开始，HTML 就比较重视内容而不是形式：该标准鼓励设计者注重提供高质量的可读信息，而把表现形式的问题留给浏览器去处理。样式表则用多种额外的效果扩展了表现形式，包括颜色、字体等方面的选择，甚至加入了声音，这样用户就可以更好地区分文档中的元素。更重要的是，样式表允许用户独自控制文档中所有标签的表现属性——不论是单篇网页还是整个网站的表现形式。

1996 年初，万维网联盟（W3C）提出了一个定义 HTML 级联样式表（Cascading Style Sheets，CSS）的建议草案，这个建议草案很快就成为了一个被广泛采纳的标准。1998 年，W3C 扩展了其原有的规范建立了 CSS2，除熟悉的屏幕浏览器之外，还包括各种媒介的表现形式标准，以及许多其他增强的功能。

层叠样式表 CSS 可以使用 HTML 标签或命名的方式定义，除可控制一些传统的文本属性外（例如字体、字号、颜色等），还可以控制一些比较特别的 HTML 属性（如对象位置、图片效果、鼠标指针等）。层叠样式表可以一次控制多个文档中的文本，并且可随时改动层叠样式表 CSS 的内容，以自动更新文档中文本的样式。

总的来说，层叠样式表 CSS 能够完成下列工作：弥补 HTML 对网页格式化功能的不足例如段落间距、行距、字体变化和大小、页面格式的动态更新、排版定位等。

2 样式表的基本语法

样式表类型

简单地说，样式是一个规则，告诉浏览器如何表现特定 HTML 标签中的内容。每个标签都有一系列相关的样式属性，它们的值决定了浏览器将如何显示这个标签。一条规则定义了标签中一个或几个属性的特定值。表达一个样式表的基本写法有 3 种：内联样式表、文档级样式表、外部样式表。在文档中可以使用一种或者多种样式表。浏览器会将每个样式的定义合并在一起，或者重新定义标签内容的样式特性。

内联样式表

内联样式表是连接样式和标签的最简单方式，只需在标签中包含一个 Style 属性，后面再跟一系列属性及其属性值即可，浏览器会根据样式属性及其值来表现标签中的内容。

原始文件	Sample\Ch12\1\1.html
最终文件	Sample\Ch12\1\1-end.html
学习要点	使用内联样式表

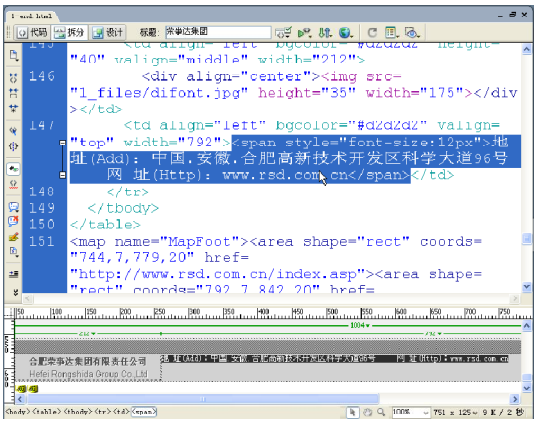
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<span style="font-size:12px">地 址(Add)：中国.安徽.合肥高新技术开发区科学大道96号 网 址(Http)：www.rsd.com.cn</span>
-
</body>
</html>
```

2. 范例解释

设置“地 址(Add)：中国.安徽.合肥高新技术开发区科学大道96号 网 址(Http)：www.rsd.com.cn”文字内联样式的字号为12像素。

3. 代码和设计



4. 显示结果

文档样式表

如果将一系列表现规则罗列在 HTML 文档的开头，样式表的真正作用就会更加明显。将文档样式表放在<head>内的<style>标签和</style>结束标签之间，就会影响文档中所有相同标签的内容。<style>和</style>标签之间的所有内容都将被看作是样式规则的一部分，会被浏览器应用于显示的文档中。

基本语法	<style>	
	包含样式内容	
	</style>	
功能	定义文档样式	
属性及说明	属性	说明
	Dir	文本方向
	Title	给样式加上说明性的文字
	Lang	语言信息
	Media	文档要使用的媒体类型
	Type	样式类型

浏览器需要一种方法来区分文档中到底使用了哪种样式表，因此可以在<style>标签中使用Type属性。级联样式表全部都是text/css类型；JavaScript样式表使用的类型则是text/javascript。

为了帮助浏览器计算出表现文档的最佳方式，HTML 4标准支持<style>标签使用Media属性。其属性值代表文档要使用的媒介类型，默认值为Screen（计算机显示器），其他值如下表所示：

	Media属性值	说明
Media属性值及说明	Screen	计算机显示器
	Tv	电视
	Projection	剧场
	Handheld	PDA和手提电话
	Print	打印
	Braille	触感设备
	Embossed	盲文设备
	Aural	音频
	All	所有媒体

原始文件	Sample\Ch12\1\2.html
最终文件	Sample\Ch12\1\2-end.html
学习要点	使用文档样式表

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
a {text-decoration: none;}
td {font-family: "宋体";
font-size: 12px;}
-->
</style>
…
</head>
<body>
…
</body>
</html>
```

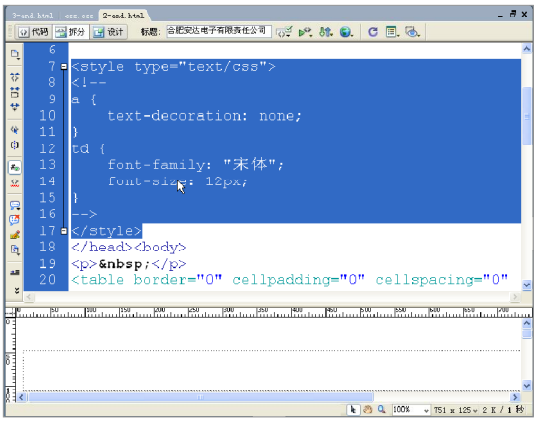
2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

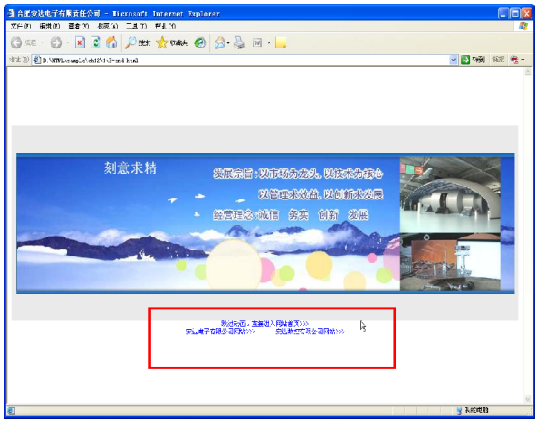
02 声明样式内容（主要包括链接的文字没有下划线，单元格中的文字为宋体，字号为12像素）。

03 文档样式表结束。

3. 代码和设计



4. 显示结果



外部样式表

设计者可以在分离的文档中放置样式定义（将其 MIME 类型定义为 text/css 的文本文件），这样就把“外部”样式表引入了文档中。同一种样式表可以应用于多个文档中。由于外部样式表是一个独立的文件，并由浏览器通过网络进行加载，所以可以随意存储和使用，甚至可以使用其他样式表。

基本语法	<link href="url">	
功能	定义当前文档和其他文档的关系	
属性及说明	属性	说明
	Dir	文本方向
	Title	给语句加上说明性的文字
	Lang	语言信息
	Target	链接的目标窗口
	Type	链接类型
	Class	用一个名称来标记语句，该标记名称指向一个预定义的类，而该类是在文档级声明的或者在外部定义的样式表
	Id	为语句创建一个标记，应用超链接时可以用这个标记来明确地引用该语句，以便作为样式表选择器或使用其他应用程序来执行自动搜索
	Style	创建语句内容的内联样式
	Charset	指定作为目标文档中所使用的字符编码
	Href	要链接的文件的路径
	Media	文档要使用的媒体类型
	Rel	指定从源文档到目标文档的关系
	Rev	指定从目标文档到源文档的关系

<link>标签可以为当前文档和Web上的某个文档建立一种联系。用于指定样式表的<link>及其必需的Href 和 Type 属性，都必需出现在文档的<head>标签中。样式表的 URL 可以是基于文档基本 URL 的绝对或相对 URL。

原始文件	Sample\Ch12\1\3.html、2_files\css.css
最终文件	Sample\Ch12\1\3-end.html
学习要点	使用外部样式表

1. 3-end.html范例代码

```
<html>
<head>
…
<link href="2_files/css.css"
rel="stylesheet" type="text/
css">
…
</head>
<body>
…
</body>
</html>
```

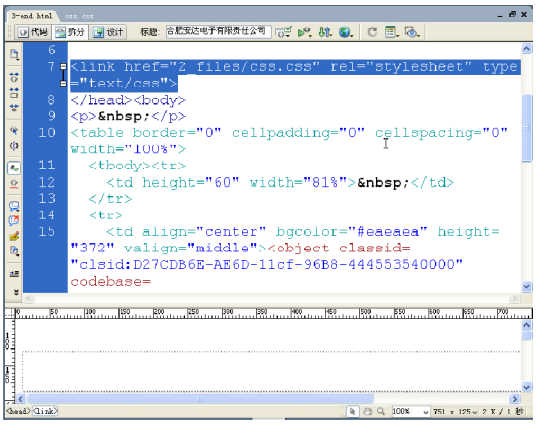
链接至外部的位于2_files文件夹下的css.css，类型为样式表。

2. Css.css范例代码

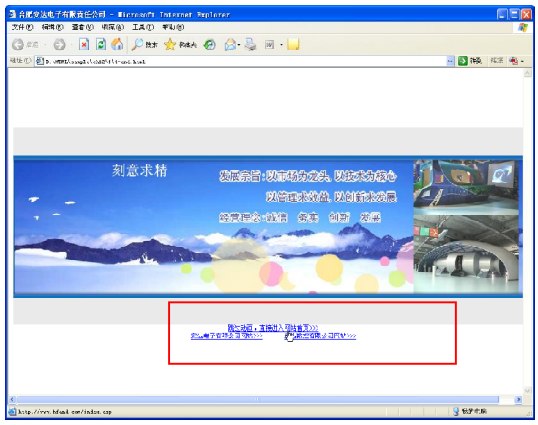
```
a {
    text-decoration: none;
}
td {
    font-family: "宋体";
    font-size: 12px;
}
```

放置的样式表的具体内容。

3. 代码和设计



4. 显示结果



样式表语法

样式表的语法包含以下几种类型：HTML 重定义标签样式表、类样式表、伪类样式表、混合类样式表。

HTML重定义标签样式表

HTML 重定义标签样式表的样式规则至少由 3 个基本部分组成。一是 CSS 样式；二是一个选择符（selector），这是受样式规则影响的 HTML 或 XHTML 标记元素的名称；三是由大括号（{}）括起来并用分号分隔的一个或多个样式“属性：值（property：value）”对的列表，例如：

```
td {color: #000099; font-size: 9pt }
```

在这个示例中，td 是 HTML 标签，Color 是样式属性，#000099 是值，看起来简洁明了。属性至少需要一个值，但许多属性包括了两个或更多个值；当有多个值时，可以用空格将它们分开，有些属性则需要用逗号来将多个值分开。

原始文件	Sample\Ch12\1\4.html
最终文件	Sample\Ch12\1\4-end.html
学习要点	使用HTML重定义标签样式表

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
td
{font-family: "宋体";
    font-size: 12px;}
-->
</style>
-
</head>
<body>
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

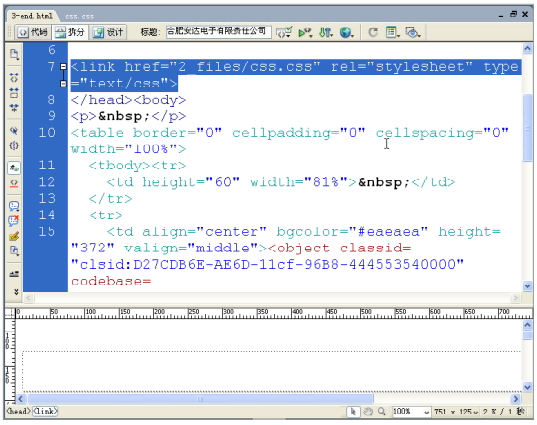
02 声明 td 标签样式。

03 声明字体为宋体。

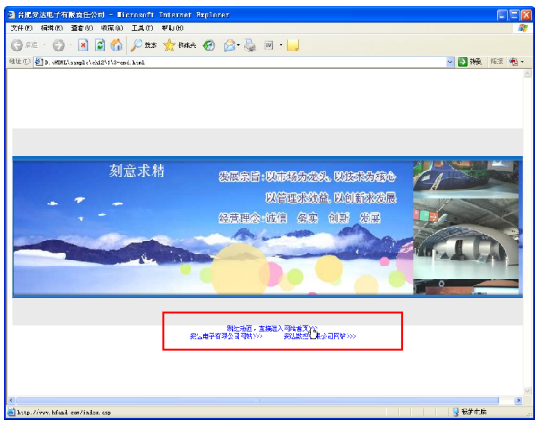
04 声明字号为 12 像素。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果



类样式表

使用类样式能够在文档样式表（或外部样式表）中为同一个元素创建多种不同的样式。在文档后面通过设置与样式相关的Class属性，并将其中一个预定义的样式指定为Name值，就可以选择要对该标签的特定情况应用何种样式。

```
.bg {background-image: url(bg.gif);}
<body class="bg">
```

在这个示例中，.bg是类，background-image是样式属性，url(bg.gif)是值。Class="bg"是在<body>标签中对样式的应用。

原始文件	Sample\Ch12\1\5.html
最终文件	Sample\Ch12\1\5-end.html
学习要点	使用类样式表

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
.bg {background-image:url
(images/bg3.jpg)}
-->
</style>
</head>
<body class="bg">
…
</body>
</html>
```

2.范例解释

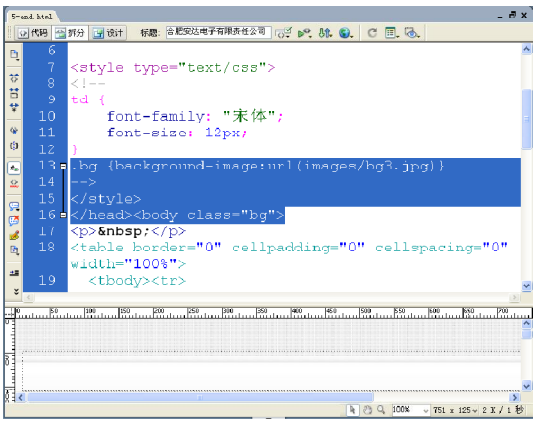
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式bg，设置了背景图像为images文件夹下的bg3.jpg。

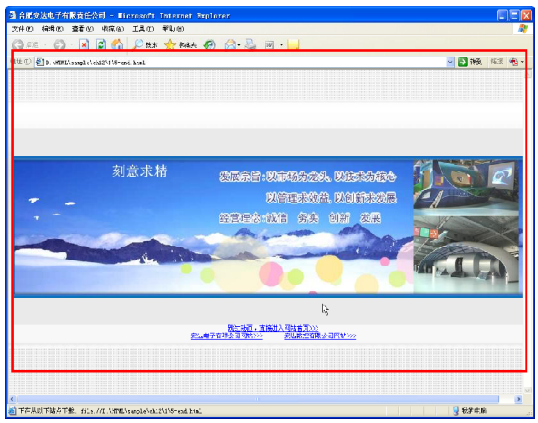
03 声明文档样式表结束。

04 将bg类样式引用在body标签中，这样整个页面就将使用bg3.jpg作为背景图片。

3. 代码和设计



4. 显示结果



伪类样式表

除了传统的样式类之外，CSS2 标准还定义了伪类，它允许为特定的标签状态定义显示样式（例如在用户选择超链接时改变显示样式）。创建伪类样式表的方法和常规类相似，但有两个显著的不同之处：它们在连接到标签名时使用的是冒号而不是句点；它们有预先定义好的名称，而不能随便命名。

```
a:link {color: #FF3366;font-family: "宋体";text-decoration: none;}
a:visited {font-family: "宋体";color: #339900;text-decoration: none;}
a:hover {color: #FF6600;font-family: "宋体";text-decoration: underline;}
a:active{font-family: "宋体";color: #339900;text-decoration: none;}
```

在这个示例中，a:link设定正常状态下链接文字的样式；a:active设定鼠标单击时链接的外观；a:visited设定访问过的链接外观；a:hover设定光标放置在链接文字之上时文字的外观。

原始文件	Sample\Ch12\1\6.html
最终文件	Sample\Ch12\1\6-end.html
学习要点	使用伪类样式表

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
-
a:link {
    color: #104C81;
    text-decoration: none;}
a:hover {
    color: #000102;
    text-decoration: none;}
a:visited {
    color: #104C81;
    text-decoration: none;}
a:active {
    color: #104C81;
    text-decoration: none;}
-->
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明默认链接样式。

03 声明默认链接样式内容。

04 声明光标上滚链接样式。

05 声明光标上滚链接样式内容。

06 声明访问过后的链接样式。

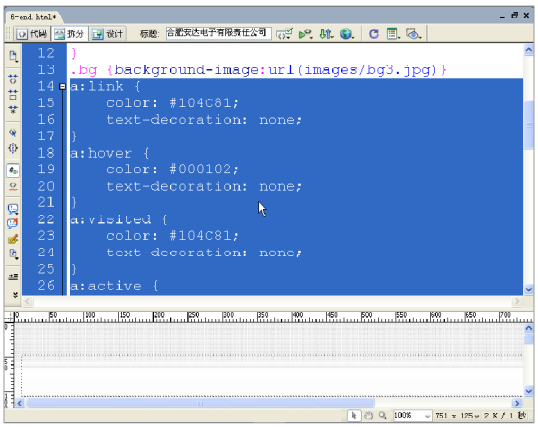
07 声明访问过后的链接样式内容。

08 声明激活状态的链接样式。

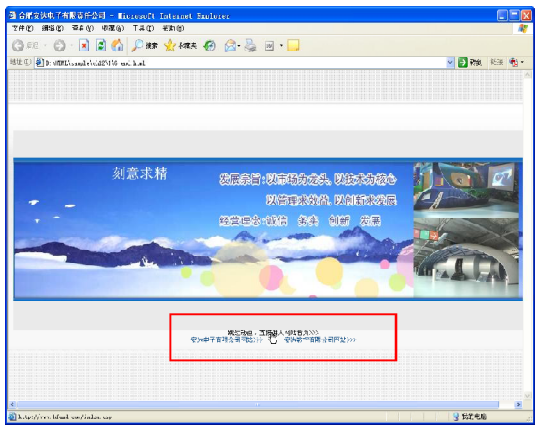
09 声明激活状态的链接样式内容。

10 声明文档样式表结束。

3. 代码和设计



4. 显示结果



混合类样式表

只要把伪类名追加到选择符的类名后面，就可以混合使用伪类与常规类。

```
a.left:link {color: #FF3366;font-family: "宋体";text-decoration: none;}
a.left:visited {font-family: "宋体";color: #339900;text-decoration:
none;}
a.left:hover {color: #FF6600;font-family: "宋体";text-decoration:
underline;}
a.left:active{font-family: "宋体";color: #339900;text-decoration: none;}
```

在这个示例中，left为常规类样式，a.left:link为伪类与常规类的混合类样式表。

原始文件	Sample\Ch12\1\7.html
最终文件	Sample\Ch12\1\7-end.html
学习要点	使用混合类样式表

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
a.copyright:link {
    color: #CC0000;
    text-decoration: none;}
a.copyright:hover {
    color: #000102;
    text-decoration: underline;}
a.copyright:visited {
    color: #104C81;
    text-decoration: none;}
a.copyright:active {
```

2.范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式copyright默认链接样式。

03 声明默认链接样式内容。

04 声明类样式copyright光标上滚链接样式。

05 声明光标上滚链接样式内容。

06 声明类样式copyright访问过后的链接样式。

07 声明访问过后的链接样式内容。

08 声明类样式copyright激活状态的链接样式。

```
    color: #104C81;
    text-decoration: none;}
-->
</style>
</head>
<body class="bg">
-
<a href="http://www.hfand.com/
index.asp" target="_blank"
class="copyright">安达电子有限公司
网站&gt;&gt;&gt;</a>
<a href="http://www.hfand.com/
shukong/index.asp"
target="_blank"
class="copyright">安达数控有限公司
网站&gt;&gt;&gt;</a>
-
</body>
</html>
```

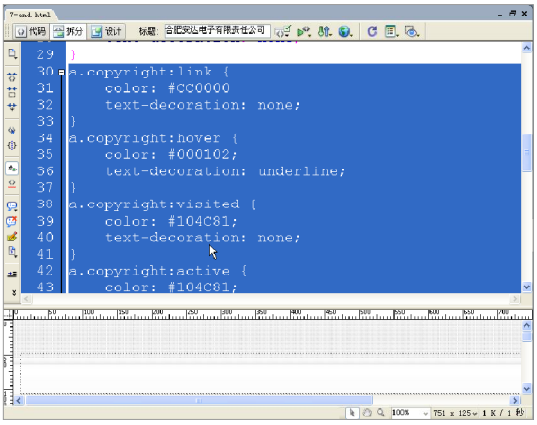
09 声明激活状态的链接样式内容。

10 声明文档样式表结束。

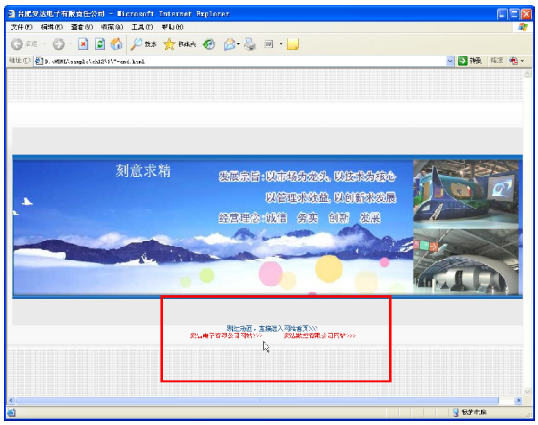
11 将类样式 copyright 应用到链接文字“安达电子有限公司网站>>>”上。

12 将类样式 copyright 应用到链接文字“安达数控有限公司网站>>>”上。

3. 代码和设计



4. 显示结果



3 样式表的属性

CSS 标准规范的核心是许多属性，它将这些属性集合在一起分为如下 8 组：字体、颜色和背景、文本、边框、方框、列表、滤镜、鼠标等。

字体属性

HTML 的<font>标签只能解决字体属性的部分问题，因为每次文本字体改变时都需要一个不同的<font>标签，样式表可以改变这种情况。CSS2 标准提供了 7 种字体属性，使用它们便可以修改受影响标签内文本的外观，具体如下表所示。

	字体属性	说明
字体属性及说明	Font-family	用一个指定的字体名或一个种类的字体族科
	Font-size	字体显示的大小
	Font-style	定义显示的字体样式
	Font-weight	定义字体重量
	Font-variant	设置英文大小写转换
	Font	组合设置字体属性

Font-family属性

Font-family（字体系列）属性采纳的是以逗号分开的字体名称列表。浏览器使用列表中命名的第一种字体在客户端机器上显示文字，当然该字体必需已经安装在该客户机上，并且可以使用。

原始文件	Sample\Ch12\2\1.html
最终文件	Sample\Ch12\2\1-end.html
学习要点	使用font-family属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font-family: "黑体";}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

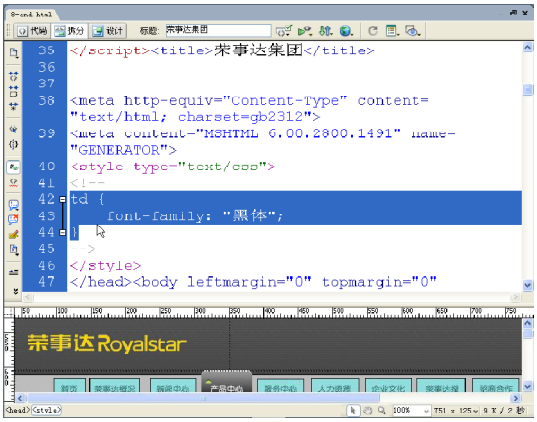
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

03 设置字体为黑体。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Font-size属性

Font-size属性允许使用相对或绝对长度值、百分比及关键字来定义字体大小。

原始文件	Sample\Ch12\2\2.html
最终文件	Sample\Ch12\2\2-end.html
学习要点	使用font-size属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font-family: "黑体";
    font-size:12px;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

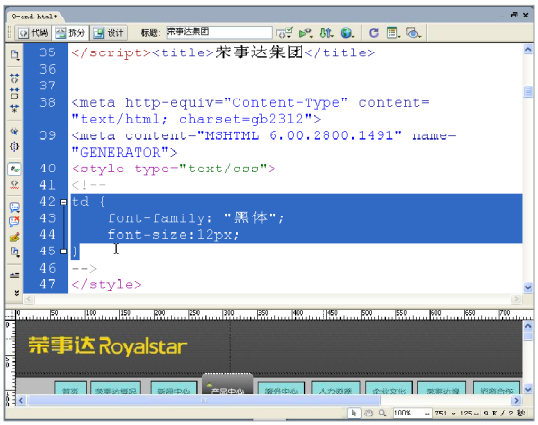
02 声明 td 单元格标签样式。

03 设置字体为黑体。

04 设置字号为 12 像素。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Font-style属性

使用Font-style属性可使文本倾斜，它的默认样式为Normal，也可以设置成Italic或Oblique。

Font-style属性值及说明	Font-style属性值	说明
	Normal	正常值
	Italic	斜体
	Oblique	偏斜体

原始文件	Sample\Ch12\2\3.html
最终文件	Sample\Ch12\2\3-end.html
学习要点	使用font-style属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font-family: "黑体";
    font-size:12px;
font-style:italic;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

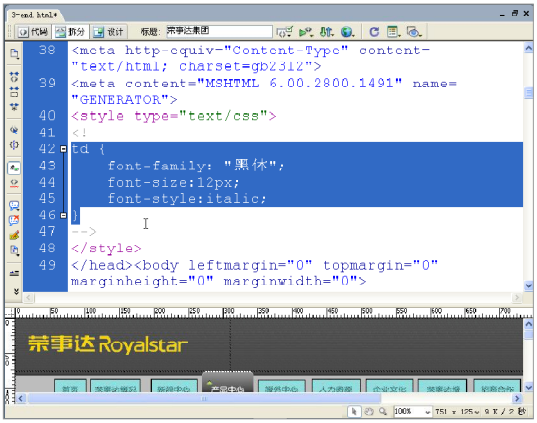
03 设置字体为黑体。

04 设置字号为 12 像素。

05 设置文字样式为斜体。

06 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Font-weight属性

Font-weight 属性控制着书写字母的粗细，它的默认值是Normal；也可以指定为Bold来得到字体的粗体版本；或使用Bolder（lighter）值来得到比父元素字体更粗（更细）的版本。它的属性值如下表所示：

Font-weight属性值及说明	Font-weight属性值	说明
	Normal	正常值
	Bold	粗体，字体粗细约为700
	Bolder	粗体再加粗，字体粗细约为900
	Lighter	比默认字体还细
	100 — 900	共有100到900九个级别，数字越小字体越细，数字越大字体越粗

原始文件	Sample\Ch12\2\4.html
最终文件	Sample\Ch12\2\4-end.html
学习要点	使用font-weight属性

1. 范例代码

```
<html>
<head>
<style type="text/css">
<!--
td {
    font-family: "黑体";
    font-size:12px;
font-style:italic;
font-weight:bold;}
-->
</style>
</head>
<body>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

03 设置字体为黑体。

04 设置字号为12像素。

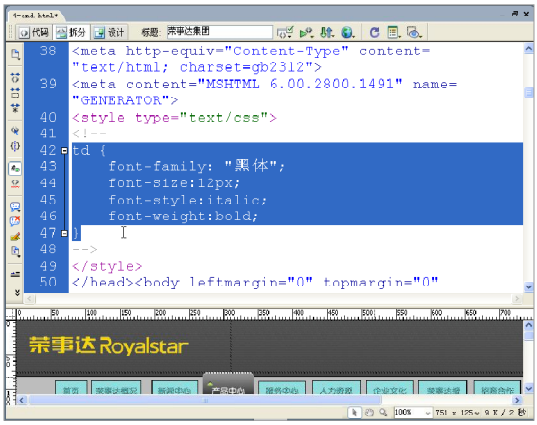
05 设置文字样式为斜体。

06 设置文字重量为粗体。

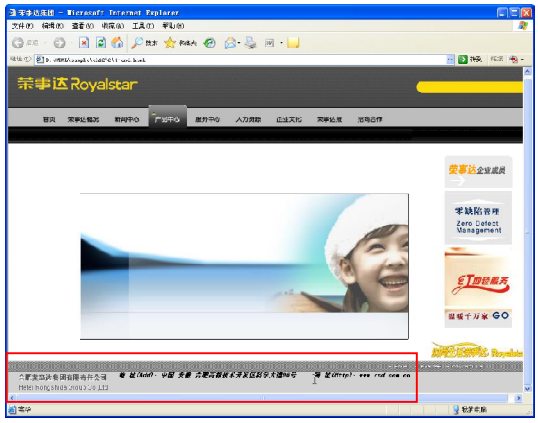
07 声明文档样式表结束。

```
…
</body>
</html>
```

3. 代码和设计



4. 显示结果



Font-variant属性

使用Font-variant属性可以选择所需字体的某种变形，它的默认值是Normal，表示字体的常规版本；也可以指定small-caps来选择字体的一个版本，在这个版本中，小写字母都会被替换为小的大写字母。Font-variant的属性值如下表所示：

Font-variant属性值及说明	Font-variant 属性值	说明
	Normal	正常值
	Small-caps	将小写英文字体转换为大写英文字体

原始文件	Sample\Ch12\2\5.html
最终文件	Sample\Ch12\2\5-end.html
学习要点	使用font-variant属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font-family: "黑体";
    font-size:12px;
font-style:italic;
font-weight:bold;
    font-variant:small-caps;}
-->
</style>
</head>
```

2.范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

03 设置字体为黑体。

04 设置字号为12像素。

05 设置文字样式为斜体。

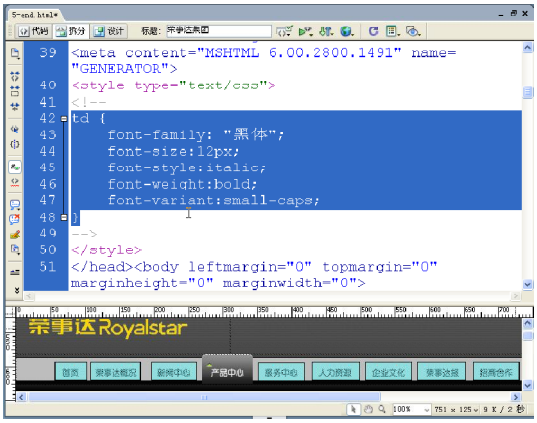
06 设置文字重量为粗体英文字体。

07 设置将小写英文字体转换为大写。

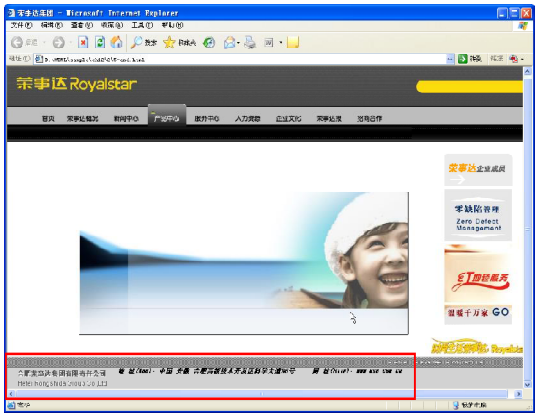
08 声明文档样式表结束。

```
<body>
…
</body>
</html>
```

3. 代码和设计



4. 显示结果



Font属性

当同时为一个标签的文本指定了多个与字体相关的属性时，完整的字体规范使用起来可能有一些不便；为了减少这种麻烦，并避免出现难以辨认的情况，应该使用易于理解的Font 属性，然后把所有属性组成一个声明的集合。字体属性的分组和排序在该集合中是很重要的：必需先用字体样式、粗细和变体属性；然后是用字体大小，最后用字体系列列表结束。在所有属性的集合中，字体大小和字体系列是必需的；其他的则可以省略。

原始文件	Sample\Ch12\2\6.html
最终文件	Sample\Ch12\2\6-end.html
学习要点	使用font属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font:italic bold small-caps
12px 黑体;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

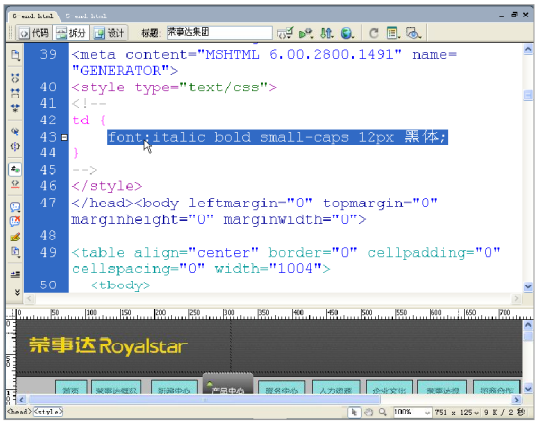
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

03 设置文字字体属性为斜体、粗体、小写英文字体转换为大写英文字体、12 像素、黑体。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果

颜色和背景属性

文档中的每个元素都有一种前景颜色和一种背景颜色。在某些情况下，背景不是颜色，而是一幅色彩丰富的图片。Color 和 Background 样式属性控制着这些颜色和图像，它们的属性如下表所示。

	颜色和背景属性	说明
颜色和背景属性及说明	Color	定义颜色
	Background-color	设置一个元素的背景颜色
	Background-image	设置一个元素的背景图片
	Background-repeat	设置一个指定的背景图片如何被重复
	Background-position	设置水平和垂直方向上的位置
	Background	组合设置背景属性

Color属性

Color 属性设置的是标签内容的前景颜色（例如文本文字的颜色）。它的值可能是一种颜色名，也可能是一个十六进制的 RGB 组合，或一个十进制的 RGB 组合。

原始文件	Sample\Ch12\2\7.html
最终文件	Sample\Ch12\2\7-end.html
学习要点	使用color属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
td {
font:italic bold small-caps
12px 黑体;
color:#990000;}
-->
</style>
</head>
<body>
-
</body>
</html>
```

2.范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

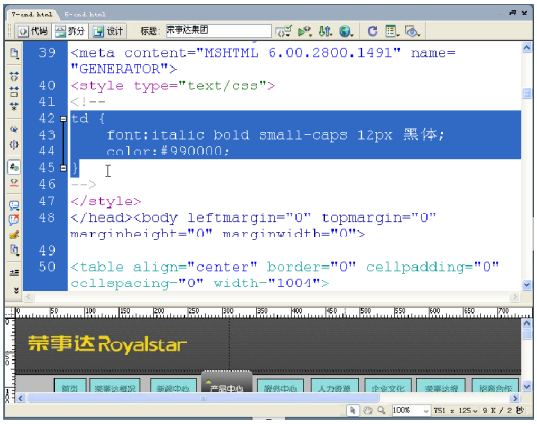
02 声明 td 单元格标签样式。

03 设置文字字体属性为斜体、粗体、小写英文字体转换为大写英文字体、12 像素、黑体。

04 设置颜色为 #990000 的红色。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Background-color属性

使用Background-color属性设定页面元素的背景颜色。

原始文件	Sample\Ch12\2\8.html
最终文件	Sample\Ch12\2\8-end.html
学习要点	使用background-color属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
    font:italic bold small-caps
12px 黑体;
color:#990000;
background-color:#EEEEEE;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

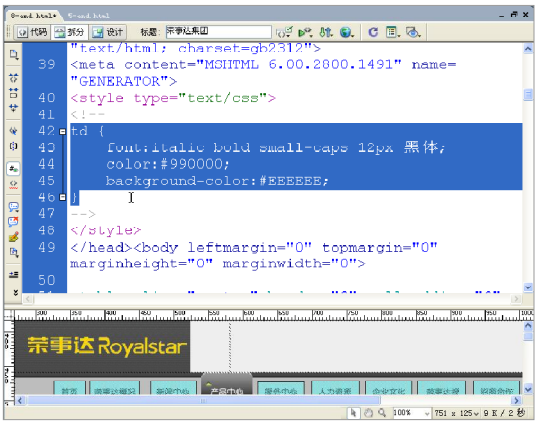
03 设置文字字体属性为斜体、粗体、小写英文字体转换为大写英文字体、12像素、黑体。

04 设置颜色为#990000的红色。

05 设置背景颜色为#EEEEEE的灰色。

06 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Background-image属性

使用Background-image属性可以在元素内容后面放置一张图片。它的值可以是一个URL，也可以是关键字none（默认值）。

原始文件	Sample\Ch12\2\9.html
最终文件	Sample\Ch12\2\9-end.html
学习要点	使用background-image属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!-
-
body{
background-image:url(images/bg3.
jpg);}
-->
</style>
</head>
<body>
-
</body>
</html>
```

2.范例解释

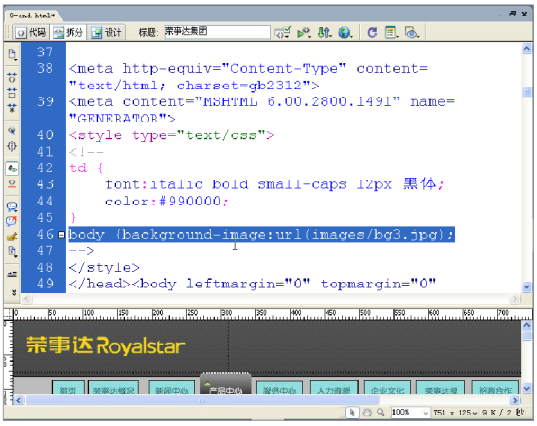
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明body主体标签样式。

03 设置背景图像为images文件夹下的bg3.jpg。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Background-repeat属性

浏览器通常会平铺背景图像来填充分配的区域，也就是在水平和垂直方向上重复显示该图像，使用Background-repeat属性可以改变这种“repeat（重复）”（默认值）行为。如果需要图像只在水平方向重复而垂直方向不重复，可使用repeat-x值；如果只想在垂直方向上重复，可使用repeat-y值；如果要禁止重复，可以使用no-repeat值。Background-repeat的属性值如下表所示。

Background-repeat 属性值及说明	Background-repeat 属性值	说明
	Repeat	背景图像平铺
	Repeat-x	背景图像以 x 轴方向平铺
	Repeat-y	背景图像以 y 轴方向平铺
	No-repeat	背景图像不平铺

原始文件	Sample\Ch12\2\10.html
最终文件	Sample\Ch12\2\10-end.html
学习要点	使用background-repeat属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!-
…
body{
background-image:url(images/bg3.
jpg);
background-repeat:repeat-y;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

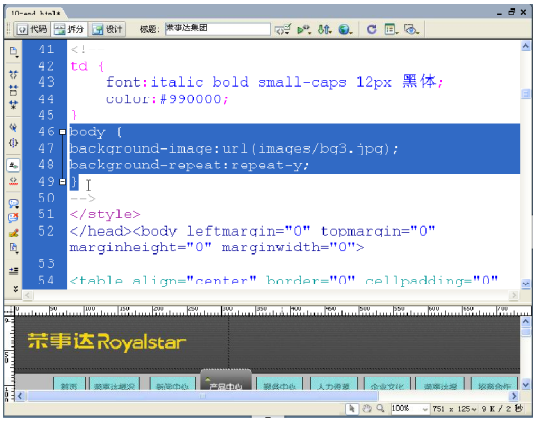
02 声明 body 主体标签样式。

03 设置背景图像为 images文件夹下的bg3.jpg。

04 设置背景图像纵向平铺。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Background-position属性

默认情况下，能够识别样式的浏览器将从分配的显示区域的左上角开始显示背景图像，并将图像平铺（如果需要的话）至同一区域的右下角。使用 Background-position 属性时，可以将背景图像的默认起始位置向下（或向）右移动一个绝对（长度）或相对（百分比或关键字）的偏移量距离，另外还可以为 Background-position 属性指定一个或两个值。如果是一个值，它将同时应用于垂直和水平位置；如果是两个值，那么第一个值表示水平偏移，第二个值表示垂直偏移。Background-position的属性值如下表所示。

	Background-position属性值	说明
Background-position 属性值及说明	Value	以百分比的形式(x百分比. y百分比)或绝对单位的形式(x . y)设定背景图像的位置
	Top	背景图像垂直居顶
	Center	背景图像垂直居中
	Bottom	背景图像垂直居底
	Left	背景图像水平居左
	Center	背景图像水平居中
	Right	背景图像水平居右

原始文件	Sample\Ch12\2\11.html
最终文件	Sample\Ch12\2\11-end.htmll
学习要点	使用background-position属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!-
-
body{
background-image:url(images/bg3.
jpg);
background-position:top left;}
-->
</style>
</head>
<body>
-
</body>
</html>
```

2.范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

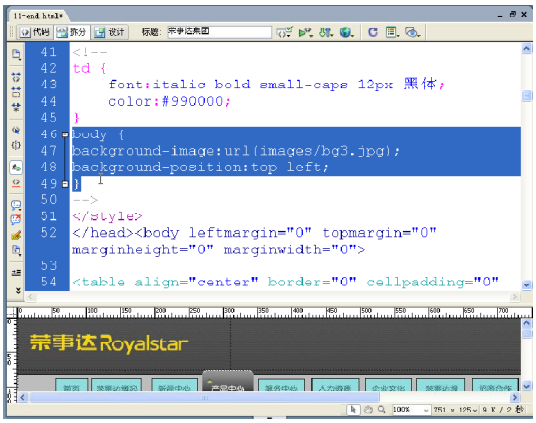
02 声明 body 主体标签样式。

03 设置背景图像为 images文件夹下的bg3.jpg。

04 设置背景图像的位置居左，居顶。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Background属性

与各种字体属性一样，许多CSS2背景属性编写起来很麻烦，而且阅读时也很困难。因此与Font属性一样，它们也要使用一个一般的Background属性。Background属性能够以任何顺序接受任何或所有Background-color、Background-image、Background-repeat和Background-position属性的值。如果没有为某些属性指定值，那些属性将被明确地设置为其默认值。

原始文件	Sample\Ch12\2\12.html
最终文件	Sample\Ch12\2\12-end.html
学习要点	使用background属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!-
…
body {
background:url(images/bg3.jpg)
repeat-y left top;}
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

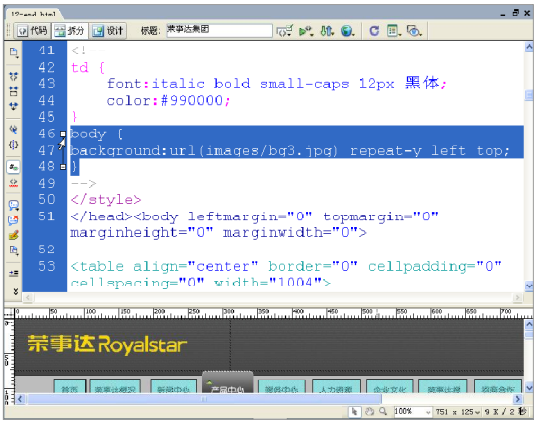
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明body主体标签样式。

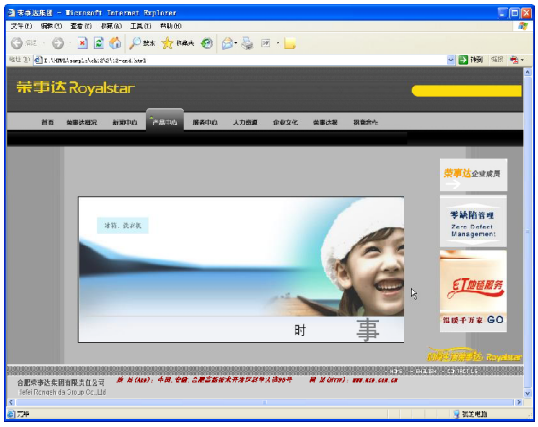
03 设置背景图像为images文件夹下的bg3.jpg，纵向平铺，位置居左，居顶。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



文本属性

样式表可以对字体属性和文本属性加以区分：前者控制文本的大小、样式和外观；而后者控制文本对齐和呈现给用户的方式。文本属性及其说明如下表所示 ：

	文本属性	说明
文本属性及说明	Letter-spacing	定义一个附加在字符之间的间隔数量
	Word-spacing	定义一个附加在单词之间的间隔数量
	Text-decoration	文本修饰属性允许通过5个属性中的一个来修饰文本
	Text-align	设置文本的水平对齐方式，包括左对齐、右对齐、居中、两端对齐
	Text-indent	文字的首行缩进
	Line-height	行高属性接受一个控制文本基线之间的间隔值
	Text-transform	控制英文文字大小写

Letter-spacing属性

Letter-spacing（字间距）属性在文本字符之间加入了额外的空格，浏览器在显示文本时可以表现出来。设置这个属性时可以使用长度值也可以使用默认关键字normal，表明浏览器应该使用正常的字符间隔。它的属性值及说明如下表所示。

	Letter-spacing属性值	说明
Letter-spacing 属性值及说明	Normal	正常值
	<length>	长度单位

原始文件	Sample\Ch12\2\13.html
最终文件	Sample\Ch12\2\13-end.html
学习要点	使用letter-spacing属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!--
td {
…
    letter-spacing:1em;}
…
-->
</style>
</head>
<body>
…
</body>
</html>
```

2.范例解释

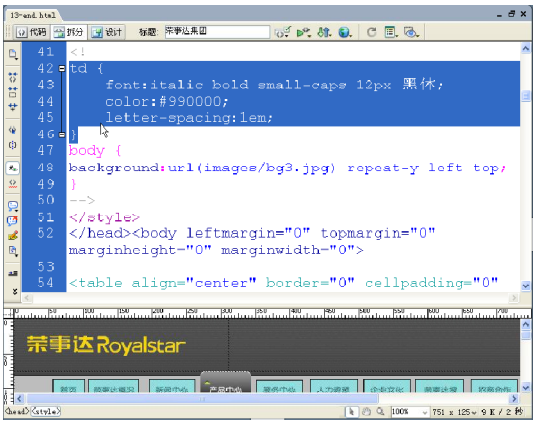
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

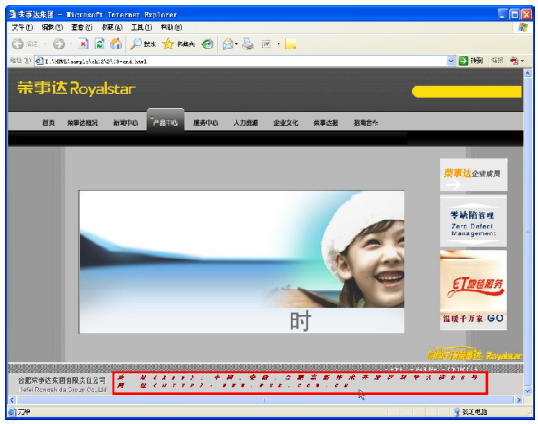
03 声明字母间距为1em。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Word-spacing属性

使用Word-spacing属性可在一个标签内的字之间添加空格。可以指定一个长度值，或用关键字normal来恢复到正常字间距。它的属性值及说明如下表所示。

Word-spacing 属性值及说明	Word-spacing 属性值	说明
	Normal	正常值
	<length>	长度单位

原始文件	Sample\Ch12\2\14.html
最终文件	Sample\Ch12\2\14-end.htmll
学习要点	使用word-spacing属性

1. 范例代码

```
<html>
<head>
-
<style  type="text/css">
<!--
td {
-
word-spacing:2em;}
-
-->
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

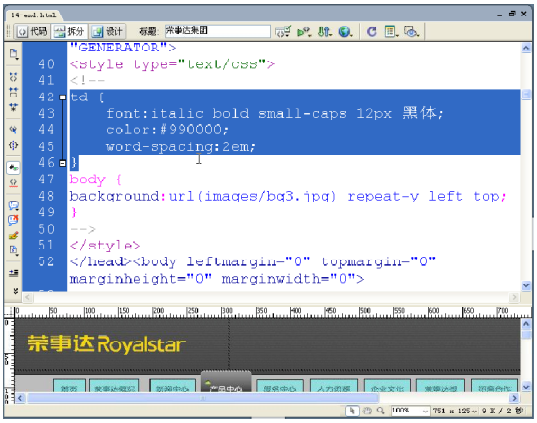
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

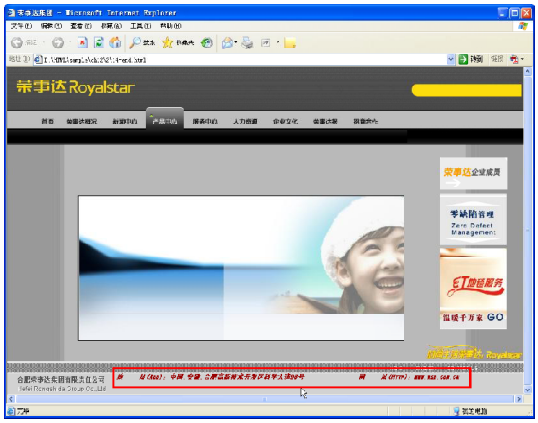
03 设置单词间距为2em。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Text-decoration属性

Text-decoration（文字修饰）属性可以产生文本修饰，其中有些还可以用于最早的物理样式标签。它的属性值是下列关键字中的一个或多个：underline、overline、line-through和blink，none是默认值，具体属性值如下表所示：

	Text-decoration属性值	说明
Text-decoration 属性值及说明	Underline	文字加下划线
	Overline	文字加上划线
	Line-through	文字加删除线
	Blink	闪烁文字，只适用于Netscape浏览器
	None	默认值

原始文件	Sample\Ch12\2\15.html
最终文件	Sample\Ch12\2\15-end.html
学习要点	使用text-decoration属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
…
text-decoration:underline;}
…
-->
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

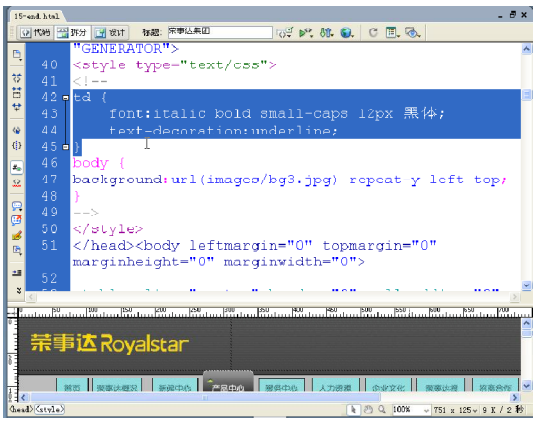
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

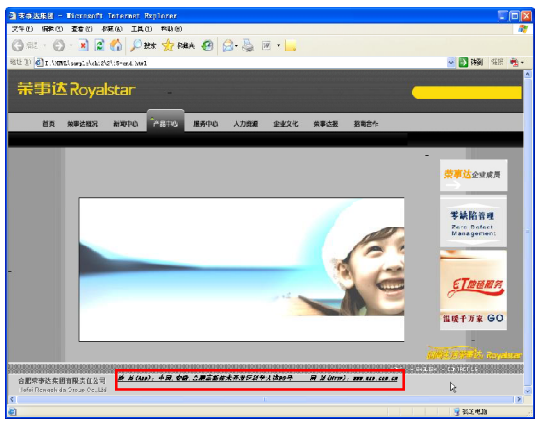
03 设置文字带有下划线。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Text-align属性

文本相对页边的调整几乎是所有文字处理器都具备的基本功能，Text-align属性可以使HTML的任何模块级标签都具有了这个功能。该属性共有4个值：left、right、center或justify，具体如下表所示。

Text-align属性值及说明	Text-align属性值	说明
	Left	居左对齐
	Right	居右对齐
	Center	居中对齐
	Justify	两端对齐

原始文件	Sample\Ch12\2\16.html
最终文件	Sample\Ch12\2\16-end.htmll
学习要点	使用text-align属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
td {
-
text-align:center;}
-
-->
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

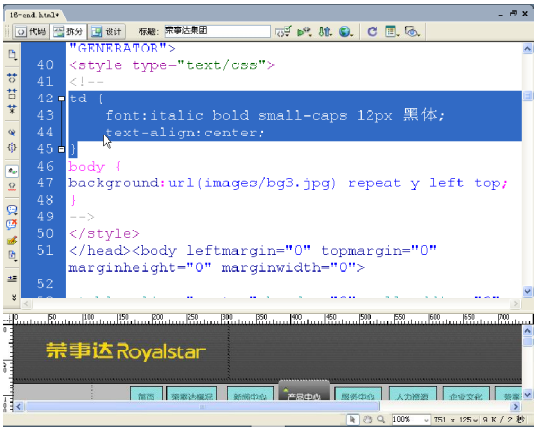
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

03 设置文字居中对齐。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Text-indent属性

使用Text-indent属性可以设定页面文字的首行缩进。

原始文件	Sample\Ch12\2\17.html
最终文件	Sample\Ch12\2\17-end.html
学习要点	使用Text-indent属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
…
text-indent:1em;}
…
-->
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

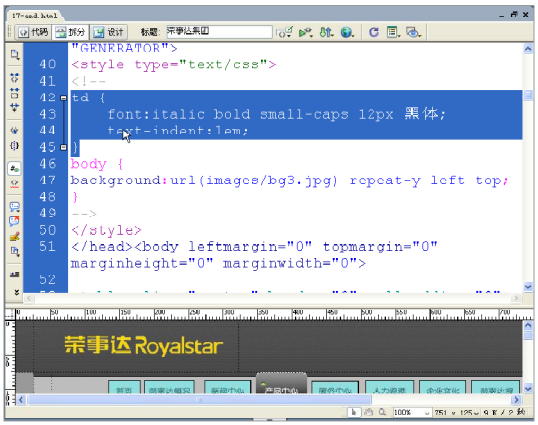
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明td单元格标签样式。

03 设置文字缩进为1em。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果

Line-height属性

使用Line-height（行间距）属性可定义标签文本内容间的最小行间距。通常情况下，浏览器会用单行距离来显示文本行（下一行的上端到上一行的下端只有几磅的间隔），但通过增加行高，便可以增加行间距。

原始文件	Sample\Ch12\2\18.html
最终文件	Sample\Ch12\2\18-end.html
学习要点	使用line-height属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
td {
-
line-height:20px;}
-
-->
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

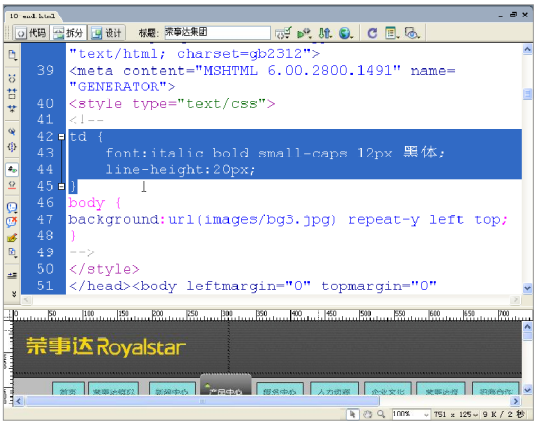
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

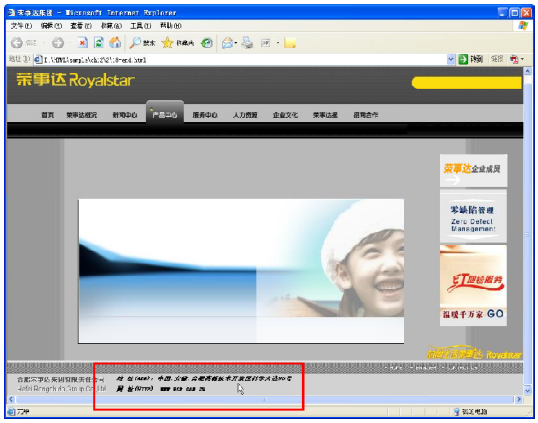
03 设置行高为 20 像素。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Text-transform属性

使用 Text-transform 属性可以自动将文档的部分（或全部）文本转换成大写（或小写）字母。该属性的值为capitalize、uppercase、lowercase或none. capitalize将文本中每个单词的第一个字母都显示为大写字母（即使源文档的文本是小写的）；uppercase 和 lowercase 值分别以相应的状态显示所有文本；none 值会取消任何转换，具体如下表所示。

Text-transform 属性值及说明	Text-transform属性值	说明
	Capitalize	将每个英文单词的首字大写
	Uppercase	将每个英文字母均转换为大写
	Lowercase	将每个英文字母均转换为小写
	None	默认值

原始文件	Sample\Ch12\2\19.html
最终文件	Sample\Ch12\2\19-end.html
学习要点	使用text-transform属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
td {
…
text-transform:uppercase;}
…
-->
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

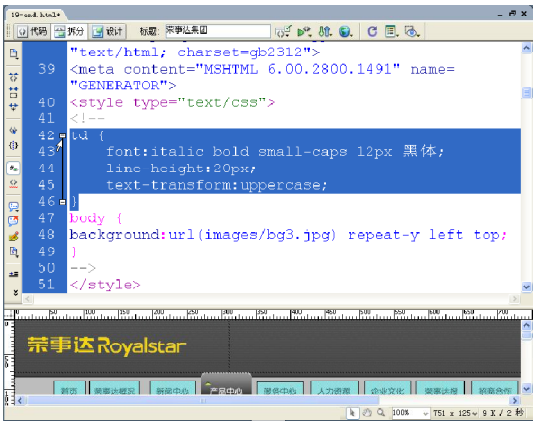
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 td 单元格标签样式。

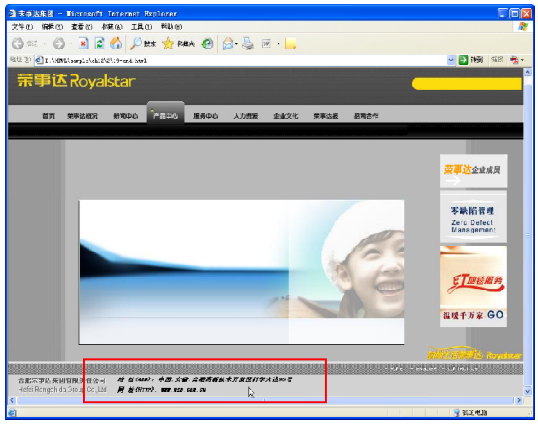
03 设置将每个英文字母均转换为大写。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



边框属性

边框属性是一个用于设置一个元素边框的宽度、样式和颜色的略写。边框属性只能设置4种边框；并且只能给出一组边框的宽度和式样。为了给出一个元素的4种边框的不同的值，网页制作者必需用一个或更多的属性，如：上边框、右边框、下边框、左边框、边框颜色、边框宽度、边框样式、上边框宽度、右边框宽度、下边框宽度或左边框宽度等，具体如下表所示。

	边框属性	说明
边框属性及说明	Border-color	边框颜色
	Border-style	边框样式
	Border-width	边框宽度
	Border-top-color	上边框颜色
	Border-left-color	左边框颜色
	Border-right-color	右边框颜色
	Border-bottom-color	下边框颜色
	Border-top-style	上边框样式
	Border-left-style	左边框样式
	Border-right-style	右边框样式
	Border-bottom-style	下边框样式
	Border-top-width	上边框宽度
	Border-left-width	左边框宽度
	Border-right-width	右边框宽度
	Border-bottom-width	下边框宽度
	Border	组合设置边框属性
	Border-top	组合设置上边框属性
	Border-left	组合设置左边框属性
	Border-right	组合设置右边框属性
	Border-bottom	组合设置下边框属性

Border-width属性

Border-width属性用来设置对象的边框宽度。如果只有一个属性值，边框的所有四面都将设置为指定宽度；如果有两个值则设上下边框为第1个值，左右边框为第2个值；如果有3个值，则第1个值是上边框，第2个值是左右边框，第3个值是下边框；如果有4个值则指定了每一面的宽度，并且按上、右、下、左边框的顺时针顺序。

原始文件	Sample\Ch12\2\20.html
最终文件	Sample\Ch12\2\20-end.html
学习要点	使用border-width属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!--
input  {
border-width:4px;}
-->
</style>
…
</head>
<body>
…
</body>
</html>
```

2. 范例解释

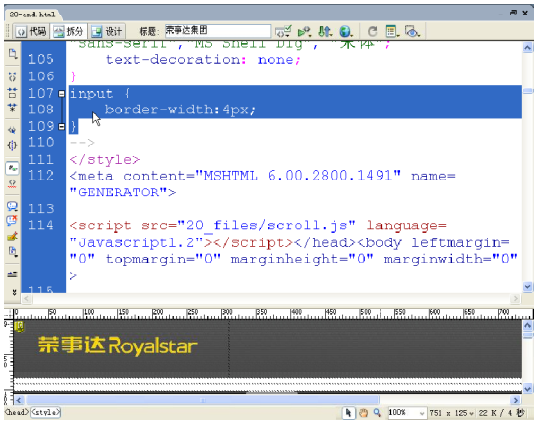
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明input输入标签样式。

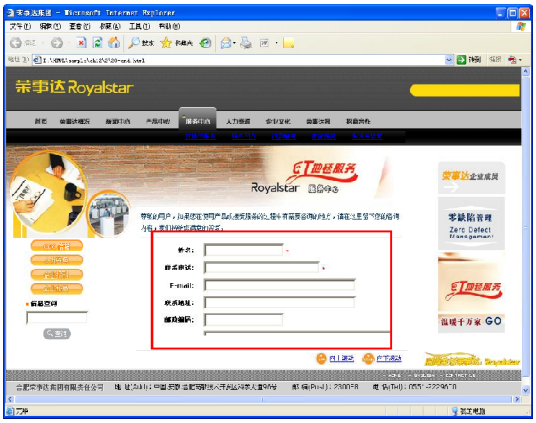
03 设置边框宽度为4像素。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Border-color属性

Border-color属性可以用来设置对象的边框颜色，和Border-width属性一样，它也有4个值，并按照类似的方式应用到不同的边框上。

原始文件	Sample\Ch12\2\21.html
最终文件	Sample\Ch12\2\21-end.html
学习要点	使用border-color属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

```
<!--
input {
border-width:1.px;
border-color:#0099FF;}
-->
</style>
-
</head>
<body>
-
</body>
</html>
```

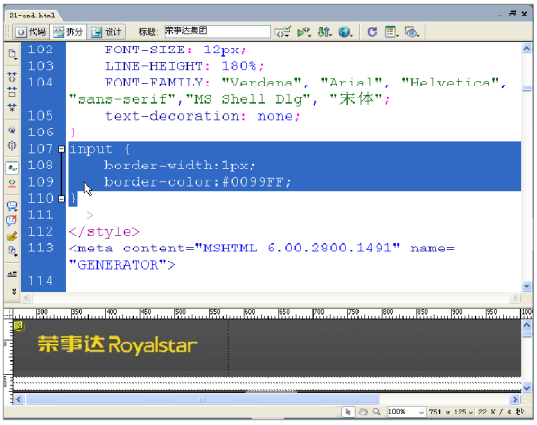
02 声明input输入标签样式。

03 设置边框宽度为1像素。

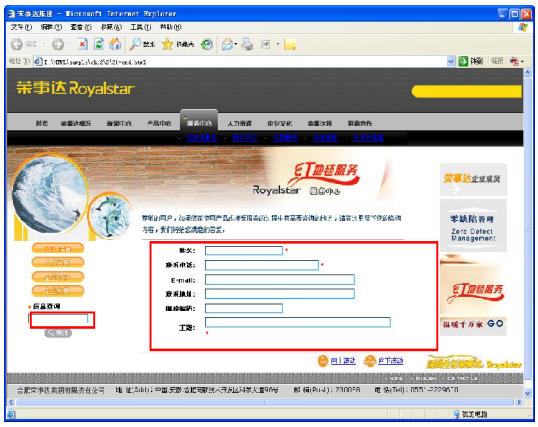
04 设置边框颜色为#0099FF的蓝色。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Border-style属性

根据CSS2模型，可以为HTML元素边框应用很多修饰。Border-style属性值包括none（默认）、dotted、dashed、solid、double、groove、ridge、inset和outset，具体如下表所示。

	Border-style属性值	说明
Border-style 属性值及说明	None	无边框
	Dotted	边框由点组成
	Dash	边框由短线组成
	Solid	边框是实线
	Double	边框是双实线
	Groove	边框带有立体感的沟槽
	Ridge	边框成脊形
	Inset	边框内嵌一个立体边框
	Outset	边框外嵌一个立体边框

原始文件	Sample\Ch12\2\22.html
最终文件	Sample\Ch12\2\22-end.html
学习要点	使用border-style属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!--
input  {
border-width:3px;
border-color:#0099FF;
border-style:double;}
-->
</style>
…
</head>
<body>
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明input输入标签样式。

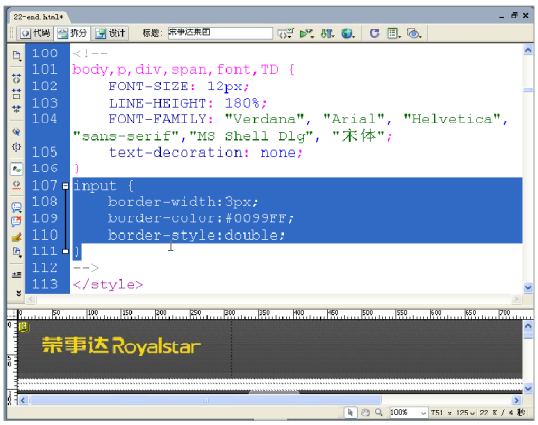
03 设置边框宽度为3像素。

04 设置边框颜色为#0099FF的蓝色。

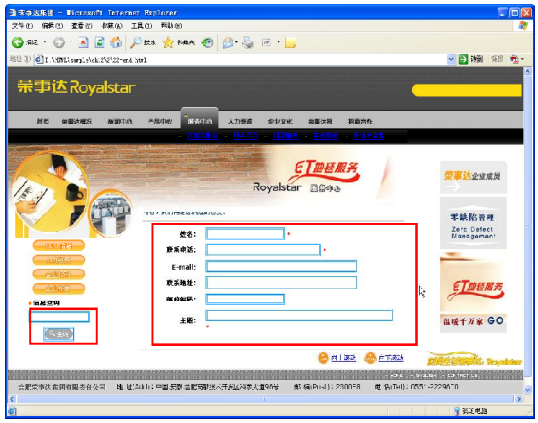
05 设置边框样式为双线。

06 声明文档样式表结束。

3. 代码和设计



4. 显示结果



Border属性

由于指定一个复杂的边框可能很繁琐，因此CSS2标准提供5种缩略属性，它们可以接受一个或所有边框的宽度、颜色和样式值中的一个或多个值；Border-top、Border-bottom、Border-left、Border-right属性只影响对应的边框侧面，而Border属性则可以同时控制边框的4个侧面。

原始文件	Sample\Ch12\2\23.html
最终文件	Sample\Ch12\2\23-end.html
学习要点	使用border属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
input {
border:dashed #3399FF 1px;}
-->
</style>
-
</head>
<body>
-
</body>
</html>
```

2. 范例解释

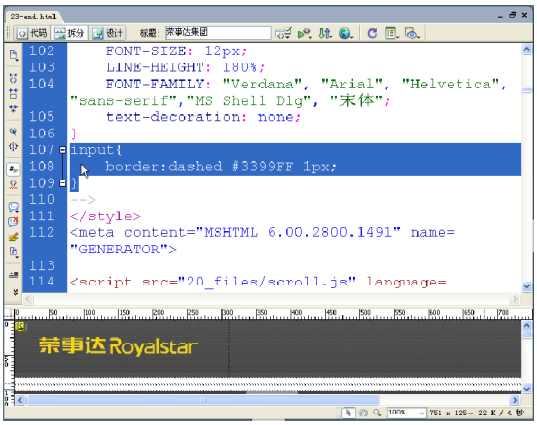
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明input输入标签样式。

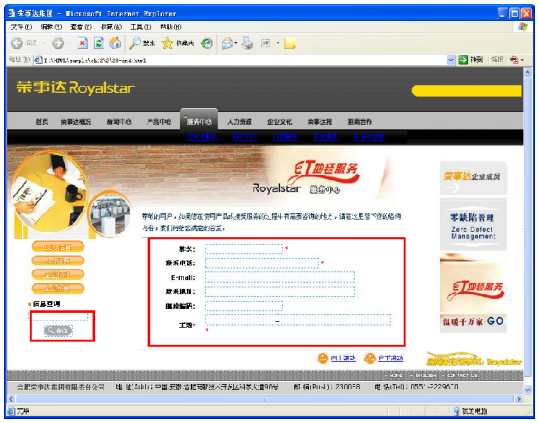
03 设置边框为虚线样式、#3399FF的蓝色，1像素宽度。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



方框属性

文档中的每个元素都可以放置在一个矩形框内，通过CSS可以控制包含文档元素的框的大小、外观和位置。这些方框属性如下表所示。

	方框属性	说明
方框属性及说明	Float	让文字环绕在一个元素的四周
	Clear	指定在某一个元素的某一边是否允许有环绕的文字或对象
	Clip	限定只显示裁切出来的区域
	Width	设定对象的宽度
	Height	设定对象的高度
	Padding	决定了究竟在边框与内容之间应该插入多少空间距离
	Margin	设置一个元素在四个方向上与浏览器窗口边界（或上一级元素的边界）的距离
	Overflow	当层内的内容超出层所能容纳的范围时的处理方式
	Position	设置对象的位置
	Z-index	决定层的先后顺序和覆盖关系
	Visibility	针对嵌套层的可视性设置

Position属性

如果Position属性设置为static，会应用常规的HTML布局和定位规则，并由浏览器决定元素框的左边缘和上边缘；要使元素相对于其包含的流移动，可以将Position属性设置为relative。在这种情况下，Top、Bottom、Left和Right属性都用来计算框相对于其在流中正常位置所处的位置，随后的元素都不会受到这种位置改变的影响，而且放在流中的方式就仿佛没有移动过该元素一样。将Position属性设置为absolute，这样就可以从文本流中去除元素，而且随后的元素可以相应地向前移动，然后使用Top、Bottom、Left和Right属性，相对于包含块计算出元素的位置：这种定位允许将元素放在其包含元素的固定位置，但会随着包含元素的移动而移动。将Position属性设置为fixed，这样会把元素相对于其显示的页面或窗口进行定位：像absolute定位一样，从包含流中去除元素时，其他的元素也会相应移动。

原始文件	Sample\Ch12\2\24.html
最终文件	Sample\Ch12\2\24-end.html
学习要点	使用position属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
.pos {
    position:absolute;
    left:260px;
    top:240px;
    width:500px;
    height:60px;}
-->
</style>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明名为pos的类样式。

03 声明位置为绝对位置。

04 声明居左260像素。

05 声明居顶240像素。

06 声明宽度为500像素。

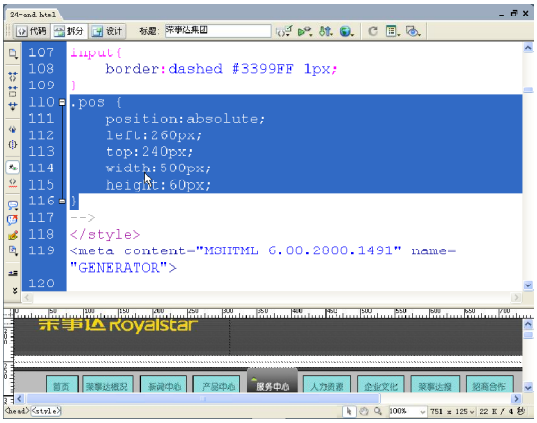
07 声明高度为60像素。

08 声明文档样式表结束。

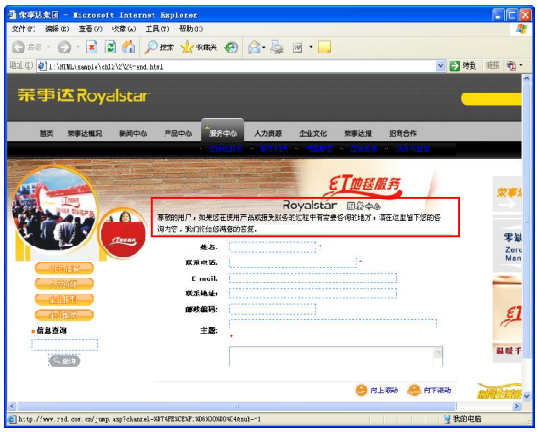
```
-
</head>
<body>
-
<div class=pos>尊敬的用户，如果您在使用产品或接受服务的过程中有需要咨询的地方，请在这里留下您的咨询内容，我们将给您满意的答复。</div>
-
</body>
</html>
```

09 为分块文字应用 pos 类样式。

3. 代码和设计



4. 显示结果



Float属性

Float 属性将标签的显示空间指定为一个浮动元素，并使文本按一定方式环绕它排列。总的来说，它与图像和表格的 Align 属性类似，但可以应用到任何元素上。Float 属性接受以下 3 个值之一：left、right 和 none（默认值），具体如下表所示。

Float属性值及说明	Float属性值	说明
	None	禁用float属性
	Left	居左对齐
	Right	居右对齐

原始文件	Sample\Ch12\2\25.html
最终文件	Sample\Ch12\2\25-end.html
学习要点	使用float属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
.imgfloat {
    float:right}
-->
</style>
…
</head>
<body>
…
<img src="25_files/
200682104410613.gif" width="249"
height="179" border="0"
class="imgfloat" title="点击查看
原图">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式 。

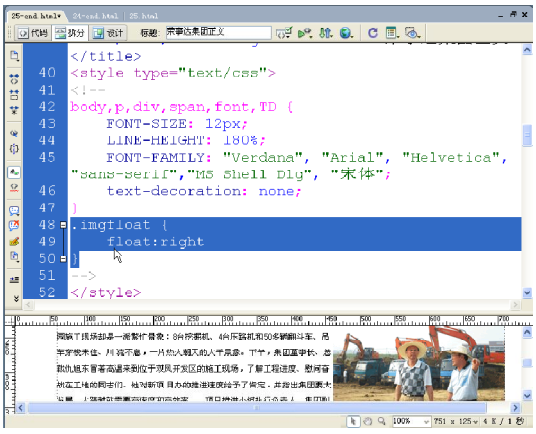
02 声明类样式，名称为imgfloat。

03 声明对象居右对齐。

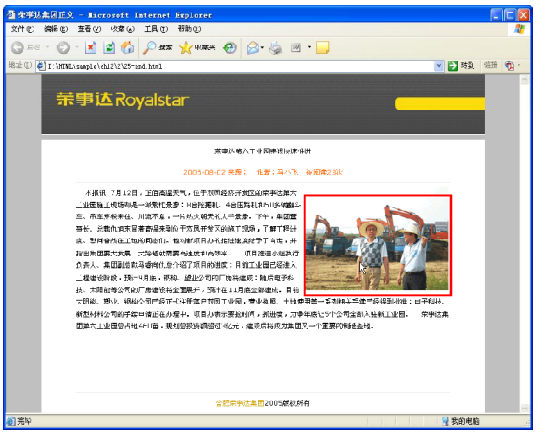
04 声明文档样式表结束。

05 对图像应用imgfloat类样式。

3. 代码和设计



4. 显示结果



Margin属性

不同的Margin属性允许控制元素四周的空白区，也就是它的边框以外的部分，该属性可以指定边界的大小，但它没有颜色或显示样式。Margin-left、Margin-right、Margin-top和Margin-bottom属性都接受长度或百分比值，并指定元素周围要保留多少空白。

原始文件	Sample\Ch12\2\26.html
最终文件	Sample\Ch12\2\26-end.html
学习要点	使用margin属性

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
.imgfloat {
    float:right;
margin:30px;}
-->
</style>
</head>
<body>
-
<img src="25_files/
200682104410613.gif" width="249"
height="179" border="0"
class="imgfloat" title=" 点击查看原图 ">
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明类样式，名称为 imgfloat。

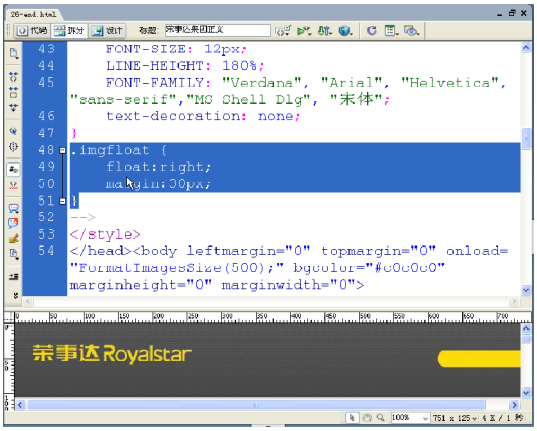
03 声明对象居右对齐。

04 声明对象的周围空白为 30 像素。

05 声明文档样式表结束。

06 对图像引用 imgfloat 类样式。

3. 代码和设计



4. 显示结果

Padding属性

与边界属性一样，不同的 Padding 属性可以控制元素周围的补白区域（元素内容区和边框之间的区域）。Padding-left、Padding-right、Padding-top和Padding-bottom属性都接受长度或百分比值，并指定元素周围要保留多少空白。

原始文件	Sample\Ch12\2\27.html
最终文件	Sample\Ch12\2\27-end.html
学习要点	使用padding属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
.textfloat {
    border:dashed 1px #0066FF;
    padding:30px;
}
-->
</style>
</head>
<body>
…
<p class="textfloat">
    本报讯 7月12日， 正值高温天气……
</p>
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式，名称为textfloat。

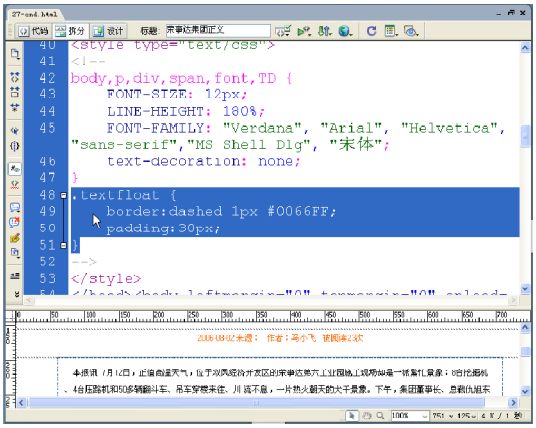
03 设置边框为虚线样式、#0066FF的蓝色，1像素宽度。

04 声明对象的内部空白为30像素。

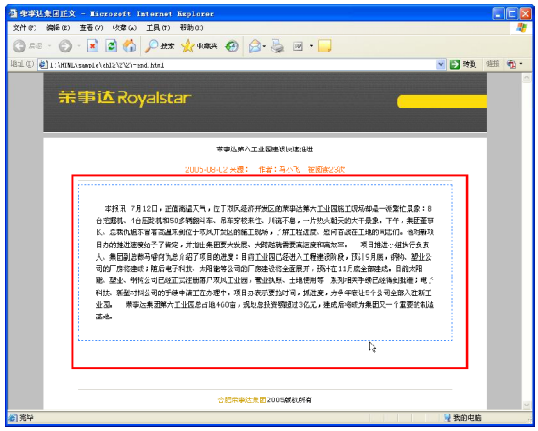
05 声明文档样式表结束。

06 对段落文字引用textfloat类样式。

3. 代码和设计



4. 显示结果



Overflow属性

Overflow属性告诉浏览器该如何处理溢出元素显示区域的内容。该属性的默认值是visible，它告诉浏览器要显示所有内容，即使内容超出了元素显示区域也要使其可见。其属性值如下表所示。

	Overflow属性值	说明
Overflow属性值及说明	Visible	无论层的大小，内容都会显示出来
	Hidden	会隐藏超出层的大小的内容
	Scroll	不管内容是否超出层的范围，选中此项都会为层添加滚动条
	Auto	只在内容超出层的范围时才显示滚动条

原始文件	Sample\Ch12\2\28.html
最终文件	Sample\Ch12\2\28-end.html
学习要点	使用overflow属性

1. 范例代码

```
<html>
<head>
-
<style  type="text/css">
<!--
-
.textfloat  {
    position:absolute;
    left:100px;
    top:220px;
    width:690px;
    height:180px;
    overflow:auto;}
-->
</style>
</head>
<body>
-
<p  class="textfloat">
     本报讯 7月12日，正值高温天气……
</p>
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式，名称为textfloat。

03 声明位置为绝对位置。

04 声明居左100像素。

05 声明居顶220像素。

06 声明宽度为690像素。

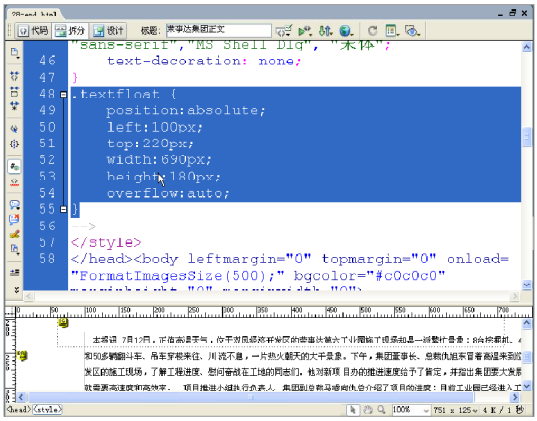
07 声明高度为180像素。

08 声明对象的溢出属性为自动。

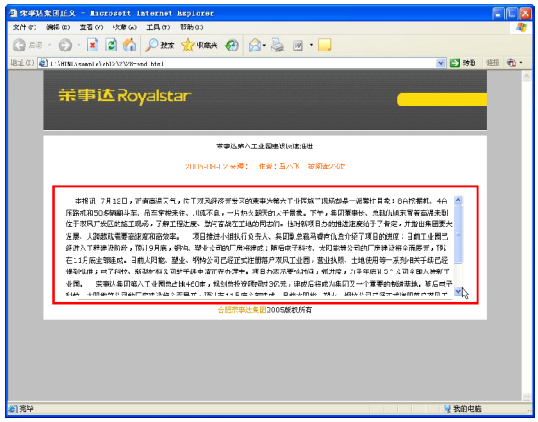
09 声明文档样式表结束。

10 对段落文字引用textfloat类样式。

3. 代码和设计



4. 显示结果



Z-index属性

在浏览器窗口和打印的页面上，除了元素的X和Y位置之外，每个元素还都有一个垂直位置，也就是Z位置。Z位置越高的元素越靠近浏览者，而用Z-index属性可以控制元素的Z位置。

原始文件	Sample\Ch12\2\29.html
最终文件	Sample\Ch12\2\29-end.html
学习要点	使用z-index属性

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
.Layer3 {
    position:absolute;
    left:274px;
    top:324px;
    width:158px;
    height:107px;
    z-index:2;}
.Layer4 {
    position:absolute;
    left:386px;
    top:362px;
    width:175px;
    height:102px;
    z-index:1;}
</style>
…
</head>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式，名称为Layer3。

03 声明对象的z值为2。

04 声明类样式，名称为Layer4。

05 声明对象的z值为1。

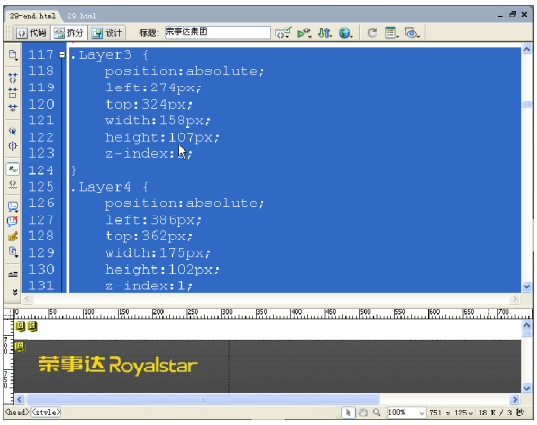
06 声明文档样式表结束。

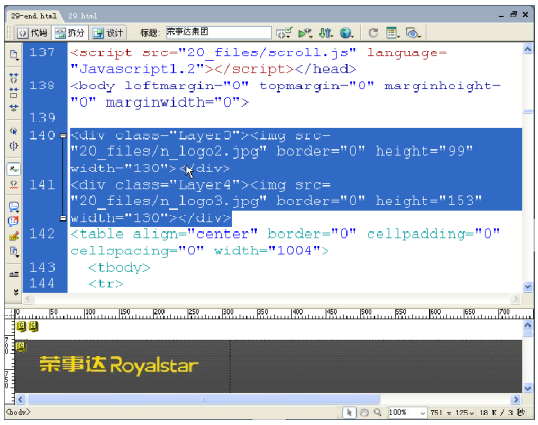
```
<body>
-
<div class="Layer3"><img
src="20_files/n_logo2.jpg" bor-
der="0" height="99"
width="130"></div>
<div class="Layer4"><img
src="20_files/n_logo3.jpg" bor-
der="0" height="153"
width="130"></div>
-
</body>
</html>
```

07 声明20_files文件夹下n_logo2.jpg所在的图像分块引用Layer3类样式。

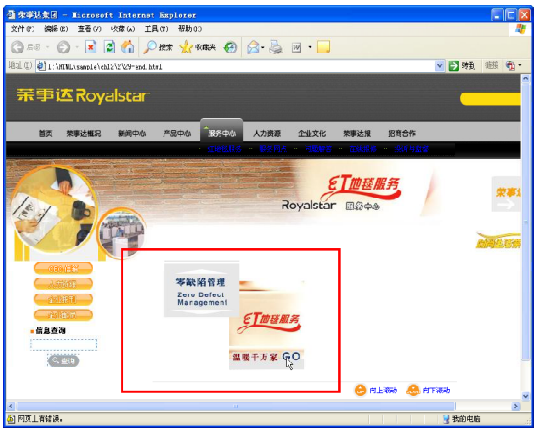
08 声明20_files文件夹下n_logo3.jpg所在的图像分块引用Layer4类样式。

3. 代码和设计





4. 显示结果



列表属性

CSS 中有关列表的属性丰富了列表的外观，这些属性如下表所示。

列表属性及说明	列表属性	说明
	List-style-type	设定引导列表项目的符号类型
	List-style-image	选择图像作为项目的引导符号
	List-style-position	决定列表项目所缩进的程度

List-style-type属性

List-style-type属性决定了有序和无序列表项如何显示在能识别样式的浏览器上。它的属性值如下表所示。

List-style-type 属性值及说明	List-style-type属性值	说明
	Disc	在文本行前面加“●”实心圆
	Circle	在文本行前面加“○”空心圆
	Square	在文本行前面加“■”实心方块
	Decimal	在文本行前面加普通的阿拉伯数字
	Lower-roman	在文本行前面加小写罗马数字
	Upper-roman	在文本行前面加大写罗马数字
	Lower-alpha	在文本行前面加小写英文字母
	Upper-alpha	在文本行前面加大写英文字母
	None	不显示任何项目符号或编号

原始文件	Sample\Ch12\2\30.html
最终文件	Sample\Ch12\2\30-end.html
学习要点	使用list-style-type属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!--
…
ul{
    list-style-type:disc;
}
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

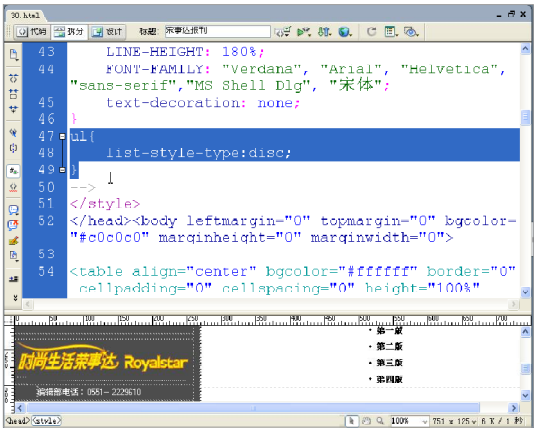
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明ul无序列表标签样式。

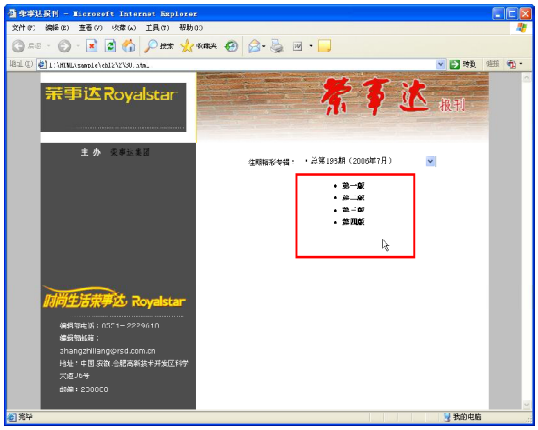
03 声明列表样式为实心圆。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



List-style-image属性

List-style-image属性定义了浏览器用来标记一个列表项的图像。它的属性值是一个图像文件的URL或者关键字none，其默认值是none。

原始文件	Sample\Ch12\2\31.html
最终文件	Sample\Ch12\2\31-end.html
学习要点	使用list-style-image属性

1. 范例代码

```
<html>
<head>
-
<style  type="text/css">
<!--
-
ul{
    list-style-image:url
(30_files/point2_r.gif);
}
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

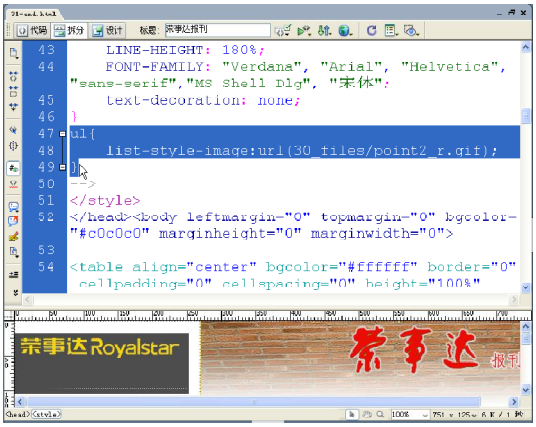
01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明ul无序列表标签样式。

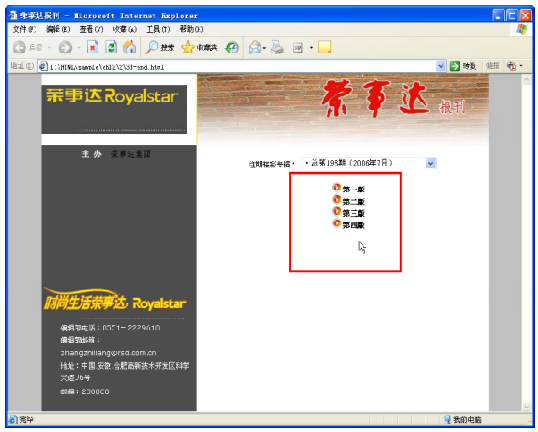
03 声明列表项图像为30_files文件夹下的point2_r.gif文件。

04 声明文档样式表结束。

3. 代码和设计



4. 显示结果



List-style-position属性

有两种方法可用来定位与一个列表项有关的记号：在与项目有关的块外面或里面。List-styleposition属性接受以下两个值之一：inside或outside，具体如下表所示。

List-style-position属性及说明	List-style-position属性值	说明
	Outside	列表贴近左侧边框
	Inside	列表缩进

原始文件	Sample\Ch12\2\32.html
最终文件	Sample\Ch12\2\32-end.html
学习要点	使用list-style-position属性

1. 范例代码

```
<html>
<head>
…
<style  type="text/css">
<!--
…
ul{
    list-style-image:url
(30_files/point2_r.gif);
list-style-position:outside;
}
</style>
</head>
<body>
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

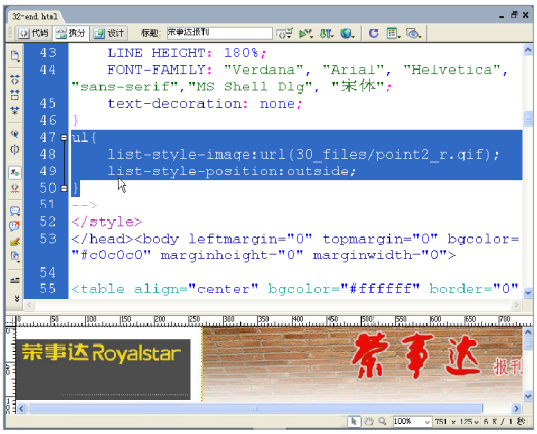
02 声明ul无序列表标签样式。

03 声明列表项图像为30_files文件夹下的point2_r.gif文件。

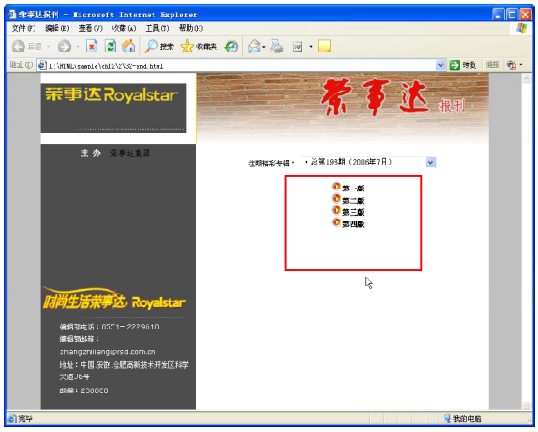
04 声明列表贴在表格中线的左侧。

05 声明文档样式表结束。

3. 代码和设计



4. 显示结果



滤镜属性

使用滤镜属性可以把可视化的滤镜和转换效果添加到一个标准的HTML 元素上，例如图片、文本，以及其他一些对象，滤镜的基本语法如下。

```
Filter：滤镜（参数）
```

可以使用的滤镜如下表所示。

滤镜	说明
Alpha	透明的层次效果
Blur	快速移动的模糊效果
Chroma	特定颜色的透明效果
Dropshadow	阴影效果
FlipH	水平翻转效果
FlipV	垂直翻转效果
Glow	边缘光晕效果
Gray	灰度效果
Invert	将颜色的饱和度及亮度值完全反转
Mask	遮罩效果
Shadow	渐变阴影效果
Wave	波浪变形效果
Xray	X 射线效果

Alpha滤镜

Alpha 滤镜可以产生颜色透明及渐变的效果，它的可用属性如下表所示。

	Alpha滤镜属性值	说明
Alpha滤镜属性值及说明	Opacity	代表透明度水准。默认的范围是从0 到 100，其实是百分比的形式。也就是说，0代表完全透明，100代表完全不透明
	Finishopacity	是一个可选参数，如果想要设置渐变的透明效果，就可以使用它们来指定结束时的透明度。范围也是0 到 100
	Style	参数指定了透明区域的形状特征。其中0代表统一形状，1代表线形，2代表放射状，3代表长方形
	Startx	代表渐变透明效果的开始X坐标
	Starty	代表渐变透明效果的开始Y坐标
	Finishx	代表渐变透明效果结束X 的坐标
	Finishy	代表渐变透明效果结束Y 的坐标

原始文件	Sample\Ch12\2\33.html
最终文件	Sample\Ch12\2\33-end.html
学习要点	使用alpha滤镜

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
.alphaimg{
    filter:alpha(opacity=50,
style=2);}
-->
</style>
</head>
<body>
…
<img src="33_files/and.jpg"
width="150" height="150"
class="alphaimg">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML 的CSS 样式。

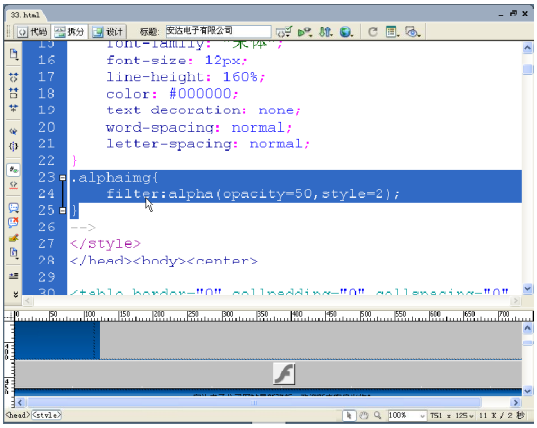
02 声明类样式，名称为alphaimg。

03 声明alpha 滤镜，不透明度为50，样式为放射状。

04 声明文档样式表结束。

05 为33_files 文件夹下的and.jpg 图像应用alphaimg样式。

3. 代码和设计



4. 显示结果

Blur滤镜

Blur 滤镜可以产生快速移动的动态模糊效果，它的可用属性值如下表所示。

	Blur滤镜属性值	说明
Blur滤镜属性值及说明	Add	是一个布尔判断“TRUE（默认）”或者“FALSE”。它指定图片是否被改变成印象派的模糊效果
	Direction	用来设置模糊的方向。其中0°代表垂直向上，然后每45°为一个单位。它的默认值是向左的270°
	Strength	只能使用整数来指定，它代表有多少像素的宽度将受到模糊影响，默认值是5个

原始文件	Sample\Ch12\2\34.html
最终文件	Sample\Ch12\2\34-end.html
学习要点	使用blur滤镜

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
-
.blurimg{
    filter:blur(strength=20);}
-->
</style>
-
</head>
<body>
-
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明类样式，名称为 blurimg。

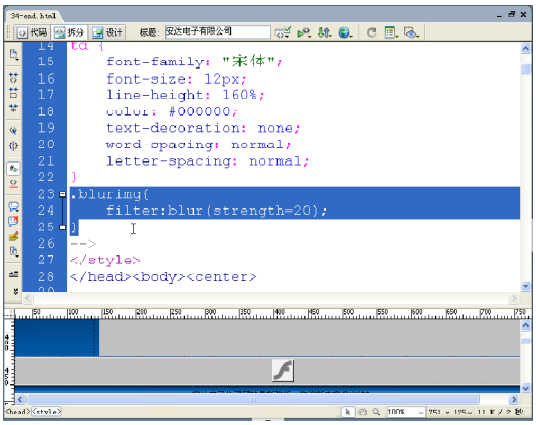
03 声明 blur 滤镜，模糊程度为 20。

04 声明文档样式表结束。

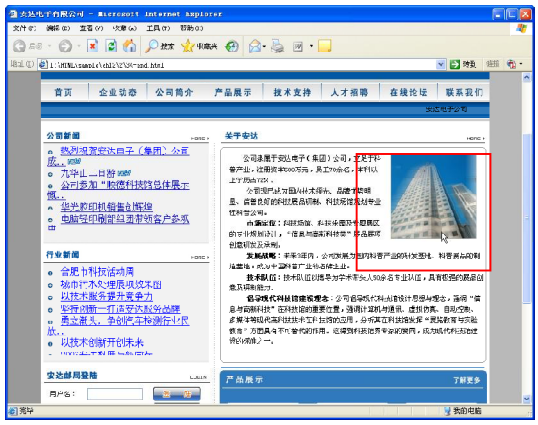
```
<img src="33_files/and.jpg"
width="150" height="150"
class="blurimg">
…
</body>
</html>
```

05 为33_files文件夹下的and.jpg图像应用blurimg样式。

3. 代码和设计



4. 显示结果



Chroma滤镜

Chroma 滤镜可以指定对象中的某个颜色变为透明效果。

原始文件	Sample\Ch12\2\35.html
最终文件	Sample\Ch12\2\35-end.html
学习要点	使用chroma滤镜

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
.chromaimg{
    filter:chroma
(color=#f2f2f2);}
-->
</style>
</head>
<body>
…
<img src="33_files/id_15.jpg"
alt="" width="296" height="230"
border="0" class="chromaimg">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

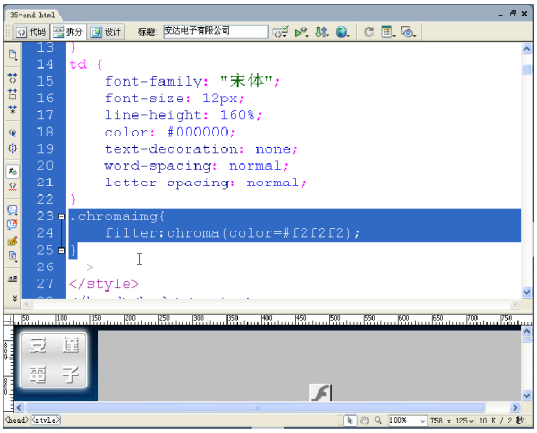
02 声明类样式，名称为 Chromaimg。

03 声明 Chroma 滤镜，其颜色为 #F2F2F2。

04 声明文档样式表结束。

05 为 33_files 文件夹下的 id_15.jpg 图像应用 chromaimg 样式。

3. 代码和设计



4. 显示结果

Dropshadow滤镜

Dropshadow 滤镜可以设置对象的阴影效果，它的属性如下表所示。

	Dropshadow 滤镜属性值	说明
Dropshadow 滤镜属性值及说明	Color	设置阴影颜色
	Offx	阴影相对于原始对象的水平位置
	Offy	阴影相对于原始对象的垂直位置
	Positive	设置阴影的透明，0 为透明，1 为不透明

原始文件	Sample\Ch12\2\36.html
最终文件	Sample\Ch12\2\36-end.html
学习要点	使用dropshadow滤镜

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
-
.dropshadowimg{
    filter:dropshadow
(color=black,offx=5,offy=5,
positive=1);}
-->
</style>
</head>
-
<img src="33_files/t.gif"
width="88" height="31"
class="dropshadowimg">
-
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

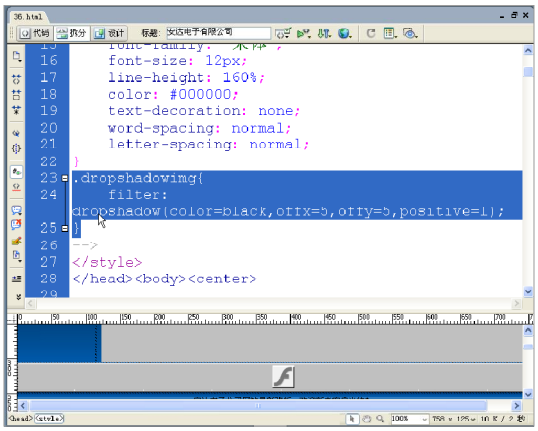
02 声明类样式，名称为 dropshadowimg。

03 声明 dropshadow 滤镜，颜色为黑色，阴影位于右下 5 像素的位置，不透明。

04 声明文档样式表结束。

05 为 33_files 文件夹下的 t.gif 图像应用 dropshadowimg 样式。

3. 代码和设计



4. 显示结果

FlipH与FlipV滤镜

FlipH滤镜可以设置对象的水平翻转，FlipV滤镜可以设置对象的垂直翻转。

原始文件	Sample\Ch12\2\37.html
最终文件	Sample\Ch12\2\37-end.html
学习要点	使用flipH滤镜

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
.fliphimg{
    filter:FlipH;}
-->
</style>
</head>
<body>
…
<img src="33_files/and.jpg"
width="150" height="150"
class="fliphimg">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

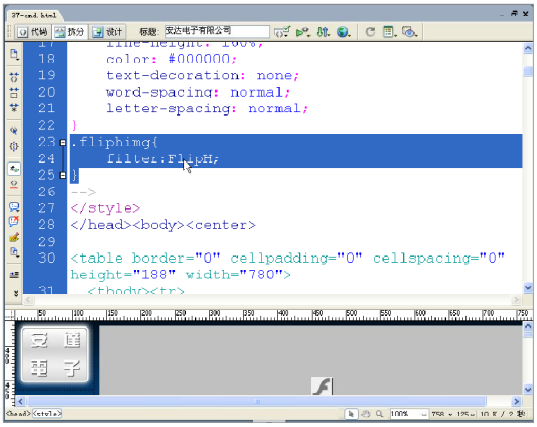
02 声明类样式，名称为fliphimg。

03 声明FlipH滤镜。

04 声明文档样式表结束。

05 为33_files文件夹下的and.jpg图像应用filphimg样式。

3. 代码和设计



4. 显示结果

Glow滤镜

Glow 滤镜设置对象产生边缘光晕的模糊效果。它的属性值如下表所示。

Glow滤镜属性值及说明	Glow 滤镜属性值	说明
	Color	设定边缘光晕的颜色
	Strength	设定边缘光晕的强度大小，取值为1~255

原始文件	Sample\Ch12\2\38.html
最终文件	Sample\Ch12\2\38-end.html
学习要点	使用glow滤镜

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
.glowimg{
    filter:glow(color=yellow,
strength=5);}
-->
</style>
</head>
<body>
-
<img src="33_files/t.gif"
width="88" height="31"
class="glowimg">
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

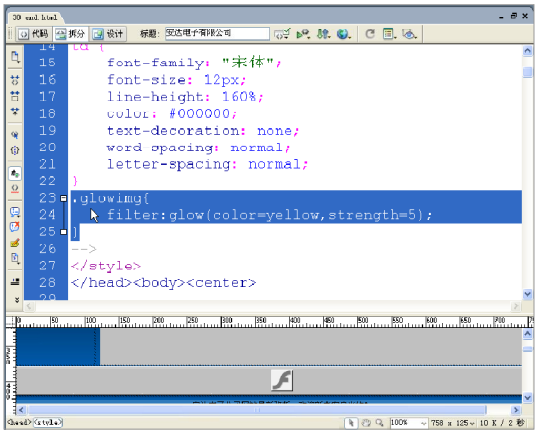
02 声明类样式，名称为glowimg。

03 声明glow滤镜，其颜色为黄色，强度为5。

04 声明文档样式表结束。

05 为33_files文件夹下的t.gif图像应用glowimg样式。

3. 代码和设计



4. 显示结果

Gray滤镜

Gray 滤镜可以将对象中的颜色除去，以变成黑白的效果。

原始文件	Sample\Ch12\2\39.html
最终文件	Sample\Ch12\2\39-end.html
学习要点	使用gray滤镜

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
.grayimg{
    filter:gray;}
-->
</style>
</head>
<body>
…
<img src="33_files/and.jpg"
 width="150" height="150"
class="grayimg">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

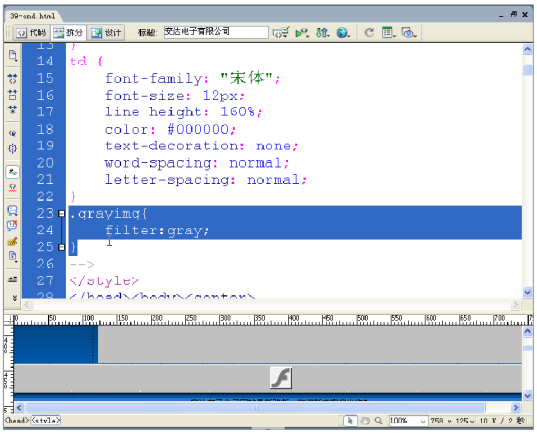
02 声明类样式，名称为grayimg。

03 声明Gray滤镜。

04 声明文档样式表结束。

05 为33_files文件夹下的and.jpg图像应用grayimg样式。

3. 代码和设计



4. 显示结果

Invert滤镜

Invert 滤镜可以将颜色的饱和度以及亮度值完全反转，类似底片效果。

原始文件	Sample\Ch12\2\40.html
最终文件	Sample\Ch12\2\40-end.html
学习要点	使用invert滤镜

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
.invertimg{
    filter:invert;}
-->
</style>
</head>
<body>
-
<img src="33_files/and.jpg"
width="150" height="150"
class="invertimg">
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

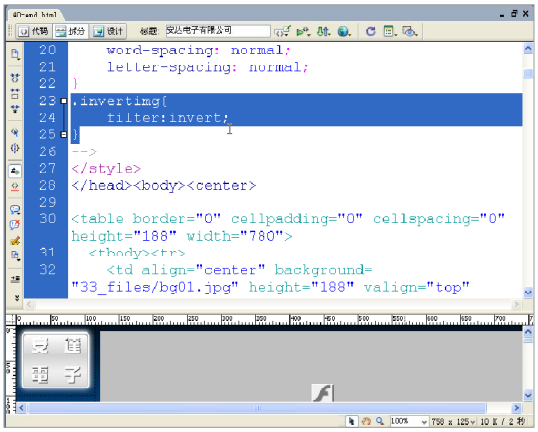
02 声明类样式，名称为 invertimg。

03 声明Invert滤镜。

04 声明文档样式表结束。

05 为 33_files 文件夹下的 and.jpg 图像应用 invertimg样式。

3. 代码和设计



4. 显示结果

Mask滤镜

Mask 滤镜设置对象的屏蔽效果，就好像印章印出的一样。

原始文件	Sample\Ch12\2\41.html
最终文件	Sample\Ch12\2\41-end.html
学习要点	使用mask滤镜

1. 范例代码

```
<html>
<head>
…
<style type="text/css">
<!--
…
    .maskimg{
    filter:mask(color=#CC0000);}
-->
</style>
</head>
<body>
…
<img src="33_files/t.gif"
 width="88" height="31"
class="maskimg">
…
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

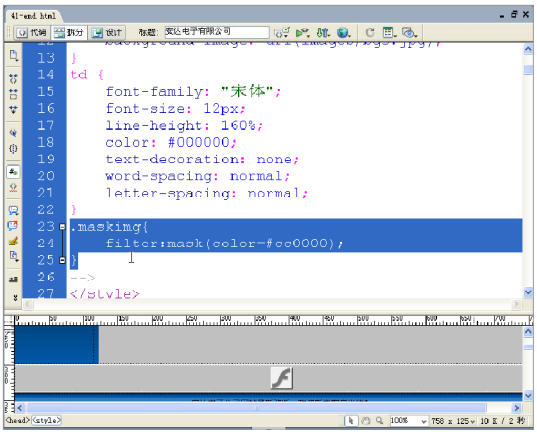
02 声明类样式，名称为 maskimg。

03 声明 mask 滤镜，颜色为 #CC0000。

04 声明文档样式表结束。

05 为33_files文件夹下的t.gif图像应用maskimg样式。

3. 代码和设计



4. 显示结果

Shadow滤镜

Shadow 滤镜除了具备 Dropshadow 滤镜的效果外，还具有渐变阴影的效果。属性值如下表所示 。

Shadow滤镜属性值及说明	Shadow 滤镜属性值	说明
	Color	设定渐变阴影的颜色
	DirectionDirection	设定阴影的方向

原始文件	Sample\Ch12\2\42.html
最终文件	Sample\Ch12\2\42-end.html
学习要点	使用shadow滤镜

1. 范例代码

```
<html>
<head>
 -
<style  type="text/css">
<!--
    .shadowimg{
    filter:shadow(color=#336699,
direction=135);}
 -->
</style>
</head>
<body>
 -
<img  src="33_files/t.gif"
width="88"  height="31"
class="shadowimg">
 -
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

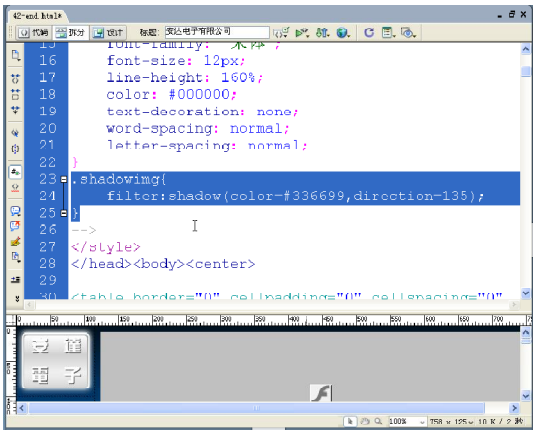
02 声明类样式，名称为 shadowimg。

03 声明 shadow 滤镜，其颜色为 #336699，方向为 135° 。

04 声明文档样式表结束。

05 为33_files文件夹下的t.gif图像应用shadowimg样式。

3. 代码和设计



4. 显示结果

Wave滤镜

Wave 滤镜设置对象产生垂直的波浪效果。它的属性值如下表所示。

Wave滤镜属性值及说明	Wave 滤镜属性值	说明
	Wave	把对象按垂直的波形样式打乱。默认是“TRUE（非0）
	Add	表示是否要把对象按照波形样式打乱
	Freq	是波纹的频率，也就是指定在对象上一共需要产生多少个完整的波纹
	Lightstrength	可以对于波纹增强光影的效果，范围0~100
	Phase	用来设置正弦波的偏移量
	Strength	代表振幅大小

原始文件	Sample\Ch12\2\43.html
最终文件	Sample\Ch12\2\43-end.html
学习要点	使用wave滤镜

1. 范例代码

```
<html>
<head>
<style type="text/css">
<!--
    .waveimg{
    filter:wave(add=add,freq=2,
lightstrength=50,phase=45,
strength=10);}
-->
</style>
</head>
<body>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

02 声明类样式，名称为waveimg。

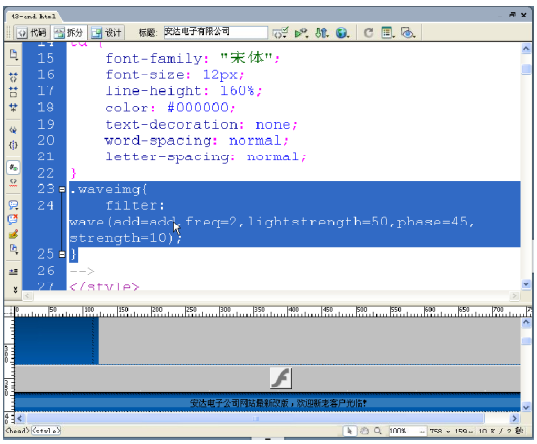
03 声明wave滤镜，把波纹对象打乱，频率为2，光影强度为50，正弦波的偏移量为45，振幅为10。

04 声明文档样式表结束。

```
<img src="33_files/and.jpg"
width="150" height="150"
class="waveimg">
-
</body>
</html>
```

05 为33_files文件夹下的and.jpg图像应用waveimg样式。

3. 代码和设计



4. 显示结果

Xray滤镜

Xray滤镜让对象显示轮廓加亮，有点类似X光片的效果。

原始文件	Sample\Ch12\2\44.html
最终文件	Sample\Ch12\2\44-end.html
学习要点	使用xray滤镜

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
    .xrayimg{
    filter:xray}
-->
</style>
</head>
<body>
-
<img src="33_files/and.jpg"
width="150" height="150"
class="xrayimg">
-
</body>
</html>
```

2. 范例解释

01 声明文档样式表开始，类型为HTML的CSS样式。

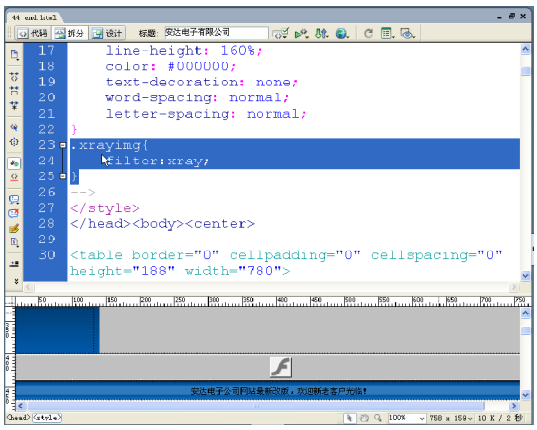
02 声明类样式，名称为xrayimg。

03 声明Xray滤镜。

04 声明文档样式表结束。

05 为33_files文件夹下的and.jpg图像应用xrayimg样式鼠标滤镜。

3. 代码和设计



4. 显示结果

鼠标滤镜

在网页中默认的鼠标指针有两种：一种是最普通的箭头，另一种是当移动到链接上时出现的“小手”。但是现在越来越多的网页都使用了CSS 鼠标指针技术，当将光标移动到链接上时，可以看到多种不同的效果。使用鼠标滤镜的基本语法如下：

```
cursor：value
```

它的属性值如下表所示。

	Cursor属性值	说明
Cursor属性值及说明	Hand	手
	Crosshair	交叉十字
	Text	文本选择符号
	Wait	Windows的沙漏形状
	Default	默认的鼠标形状
	Help	带问号的鼠标
	E-resize	向东的箭头
	Ne-resizey	指向东北方的箭头
	N-resize	向北的箭头
	Nw-resize	指向西北的箭头
	W-resize	向西的箭头
	Sw-resize	向西南的箭头
	S-resize	向南的箭头
	Se-resize	向东南的箭头

原始文件	Sample\Ch12\2\45.html
最终文件	Sample\Ch12\2\45-end.html
学习要点	使用cursor

1. 范例代码

```
<html>
<head>
-
<style type="text/css">
<!--
    body{
-
cursor:wait}
-->
</style>
</head>
<body>
-
</body>
</html>
```

2. 范例解释

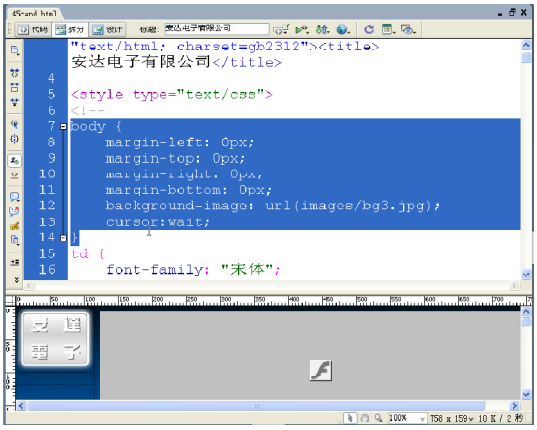
01 声明文档样式表开始，类型为 HTML 的 CSS 样式。

02 声明 body 主体标签样式。

03 声明鼠标指针为沙漏形状。

04 声明文档样式表结束。

3. 代码和设计

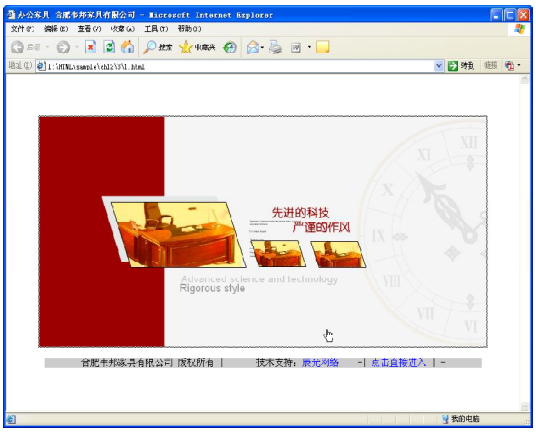


4. 显示结果

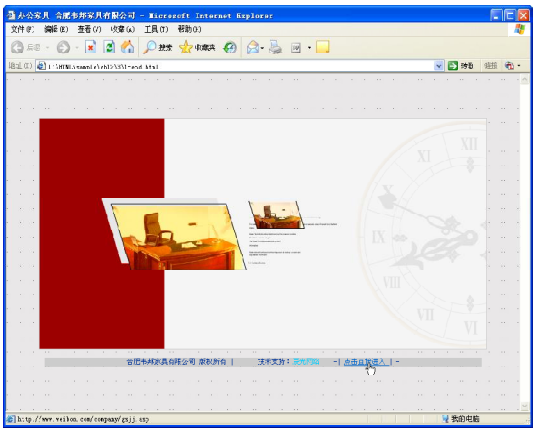
4 实例制作

下面通过一个实例练习页面中样式表的创建以及使用样式表美化页面的方法。

原始文件	Sample\Ch12\3\1.html
最终文件	Sample\Ch12\3\1-end.html
学习要点	使用CSS样式表及相关属性

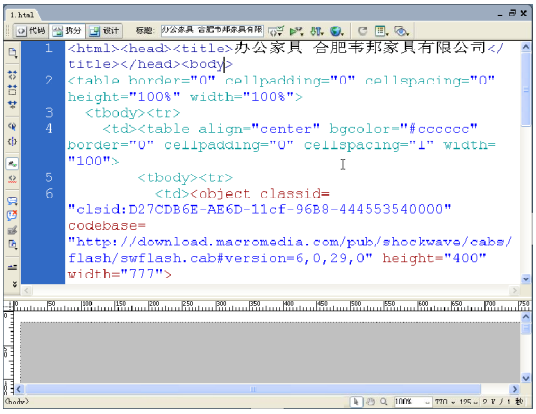


原始效果



最终效果

1 在Dreamweaver中打开提供的页面，单击“拆分”按钮，切换到“代码视图和设计视图”环境。



2 将光标定位到<head>标签内部，在代码视图中插入以下代码：

```
<style type="text/css">
<!--
td {
    font-family: "宋体";
    font-size: 12px;
    line-height: 15px;
    color: #003399;
}
body {
    background-image: url
(1_files/bj.jpg);
    margin-left: 0px;
    margin-top: 0px;
    margin-right: 0px;
    margin-bottom: 0px;
}
a {
    font-family: 宋体;
    font-size: 12px;
```

01 声明文档样式表开始。

02 声明td单元格标签样式。

03 声明字体为宋体。

04 声明字号为12像素。

05 声明行高为15像素。

06 声明颜色为#003399。

07 声明body主体标签样式。

08 声明背景图像为1_files文件夹中的bj.jpg。

09 声明页面4个方向的边距为0。

10 声明a整体链接标签样式。

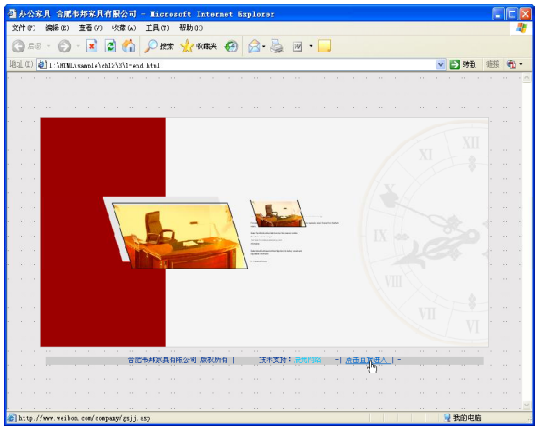
11 声明字体为宋体。

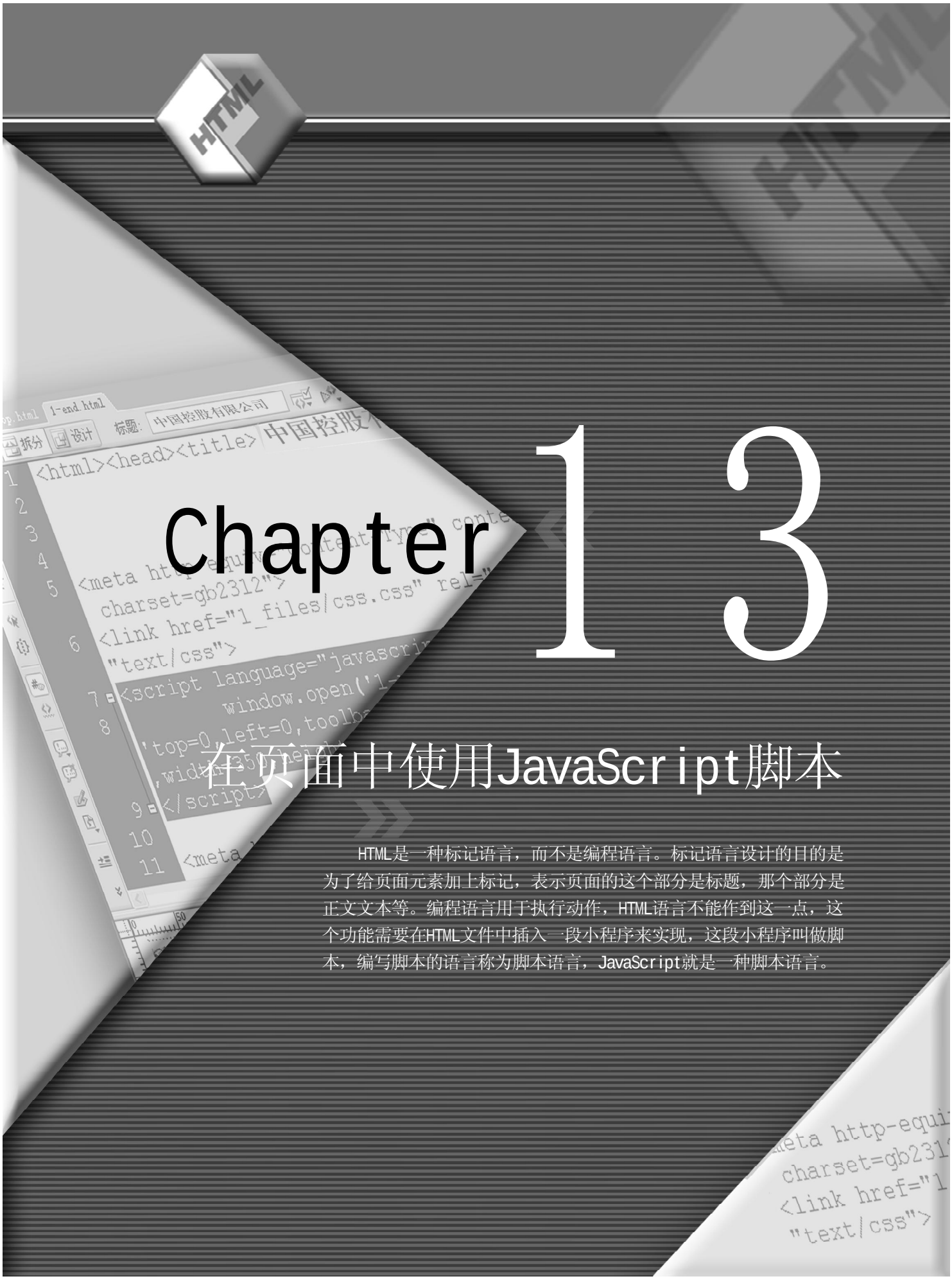
12 声明字号为12像素。

```
    color: #00CCFF;
}
a:link {
    text-decoration: none;
}
a:visited {
    text-decoration: none;
    color: #0099FF;
}
a:hover {
    text-decoration: underline;
    color: #0066CC;
}
a:active {
    text-decoration: none;
    color: #0000CC;
}
-->
</style>
```

13 声明颜色为#00CCFF。
14 声明默认链接样式。
15 声明没有下划线。
16 声明访问过后链接样式。
17 声明没有下划线。
18 声明颜色为#0099FF。
19 声明光标上滚链接样式。
20 声明出现下划线。
21 声明颜色为#0066CC。
22 声明激活状态链接样式。
23 声明没有下划线。
24 声明颜色为#0000CC。

3 按下F12快捷键预览页面，可以看到CSS样式表美化页面的效果。





Chapter 13

在页面中使用JavaScript脚本

HTML是一种标记语言，而不是编程语言。标记语言设计的目的是为了给页面元素加上标记，表示页面的这个部分是标题，那个部分是正文文本等。编程语言用于执行动作，HTML语言不能作到这一点，这个功能需要在HTML文件中插入一段小程序来实现，这段小程序叫做脚本，编写脚本的语言称为脚本语言，JavaScript就是一种脚本语言。

1 什么是JavaScript脚本

JavaScript 是一种脚本编程语言，支持 Web 应用程序的客户机和服务器方构件的开发。在客户机中，它可用于编写 Web 浏览器在 Web 页面上下文中执行的程序；在服务器中，它可用于编写处理 Web 浏览器提交的信息并相应地更新浏览器显示的 Web 服务器程序。

JavaScript 是一种脚本语言，它采用小程序段的方式实现编程，像其他脚本语言一样，JavaScript 同样也是一种解释性语言，它提供了一个易于开发的过程。它的基本结构形式与 C、C++、VB、Delphi 十分类似；但它不像这些语言一样，需要先编译，而是在程序运行过程中被逐行地解释。它与 HTML 标识结合在一起，从而方便用户的使用操作。JavaScript 是一种基于对象的语言，同时也可以看作一种面向对象的语言。这意味着它能运用自己已经创建的对象。因此，许多功能可以来自于脚本环境中对象的方法与脚本的相互作用。JavaScript的简单性主要体现在：首先它是一种基于Java基本语句和控制流之上的简单而紧凑的设计，对于学习 Java 是一种非常好的过渡；其次它的变量类型是弱类型，并未使用严格的数据类型。JavaScript 是一种安全性语言：它不允许访问本地的硬盘，并不能将数据存入到服务器上，也不允许对网络文档进行修改和删除，只能通过浏览器实现信息浏览或动态交互，从而有效地防止数据的丢失。JavaScript 是动态的：它可以直接对用户或客户输入做出响应，而无须经过 Web 服务程序。它对用户的响应，是采用以事件驱动的方式进行的。所谓事件驱动，就是指在主页（Home Page）中执行了某种操作所产生的动作，这就称为“事件”。例如按下鼠标、移动窗口、选择菜单等都可以视为事件。当事件发生后，可能会引起相应的事件响应。JavaScript 依赖于浏览器本身，与操作环境无关，只要浏览器支持JavaScript就可正确执行。

2 在页面中加入JavaScript脚本

将JavaScript代码插入到文档中的一个方法是通过HTML标准的<script>标签。在<script>和</script>之间的任何东西都被浏览器当作可执行的JavaScript语句和数据处理；不能将HTML放在这个标签内部，它们将会被浏览器当作错误标记。

Script标签

基本语法	<script>	
	包含脚本内容	
	</script>	
功能	定义脚本	
属性及说明	属性	说明
	Charset	编码脚本程序的字符集
	Language	指定脚本语言
	Src	包含脚本程序的URL
	Type	指定脚本类型

设计者可以在一个文档中包含不止一个<script>标签，位于<head>或者<body>之内均可；支持JavaScript的浏览器会顺序执行这些语句。

原始文件	Sample\Ch13\1\1.html
最终文件	Sample\Ch13\1\1-end.html
学习要点	使用<script>标签

1. 范例代码

```
<html>
<head>
-
<script language="javascript">
window.open('1-pop.html','',
'top=0,left=0,toolbar=no,
scrollbars=no,resizable=no,
width=350,height=250');
</script>
-
</head>
<body>
-
</body>
</html>
```

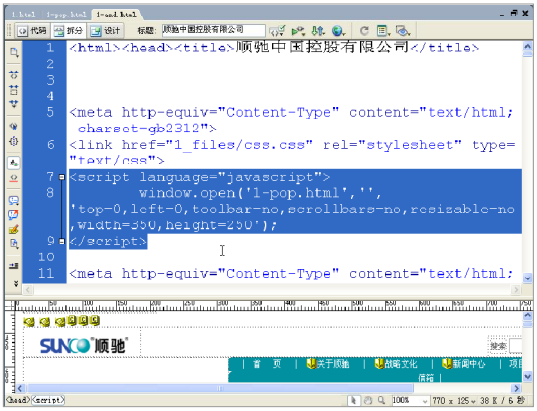
2. 范例解释

01 声明 JavaScript 脚本开始。

02 为 JavaScript 脚本内容（调用 Window 窗口对象的 open 方法，打开 1-pop.html，宽度为 350 像素，高度为 250 像素，距左为 0，距顶为 0，没有工具条和滚动条，不能改变窗口大小）。

03 结束 JavaScript 脚本。

3. 代码和设计



4. 显示结果

对特别长的JavaScript程序或者经常重复使用的程序来说，可能希望将这些代码存放在一个单独的文件中。在这种情况下，可以让浏览器通过 Src 属性来载入那个单独的文件。Src 的值是包含这个 JavaScript 程序的文件的 URL，文件名的后缀为 .js。

原始文件	Sample\Ch13\1\2.html
最终文件	Sample\Ch13\1\2-end.html,2.js
学习要点	使用<script>标签调用外部js文件

1. 2-end.html范例代码

```
<html>
<head>
…
<script language="javascript"
src="2.js">
</script>
…
</head>
<body>
…
</body>
</html>
```

2. 范例解释

01 声明JavaScript脚本，调用2.js文件。

02 结束脚本。

3. 2.js范例代码

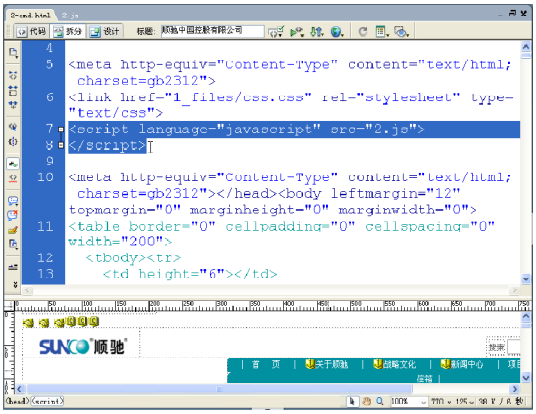
```
// JavaScript Document
window.open('1-pop.html','',
'top=0,left=0,toolbar=no,
scrollbars=no,resizable=no,
width=350,height=250');
```

4. 范例解释

01 注释当前文件为JavaScript文件。

02 为JavaScript脚本内容（调用Window窗口对象的open方法，打开1-pop.html，宽度为350像素，高度为250像素，距左为0，距顶为0，没有工具条和滚动条，不能改变窗口大小）。

5. 代码和设计



6. 显示结果

使用内在事件调用

JavaScript是基于对象的语言，而基于对象的基本特征，就是采用事件驱动。它可以在图形界面的环境下，使得一切输入简单化。通常鼠标或热键的动作称之为事件，而由鼠标或热键引发的一连串程序的动作，则称之为事件驱动。

JavaScript提供的重要的特性之一就是检测和响应文档下载、显示用户浏览的过程中发生的事

件。处理这些事件的JavaScript代码可以放置在<script>标签中，但是更普遍的做法是通过一个或者多个特殊的标签属性与一个特殊的标签联合使用。

原始文件	Sample\Ch13\1\3.html
最终文件	Sample\Ch13\1\3-end.html
学习要点	使用内在事件

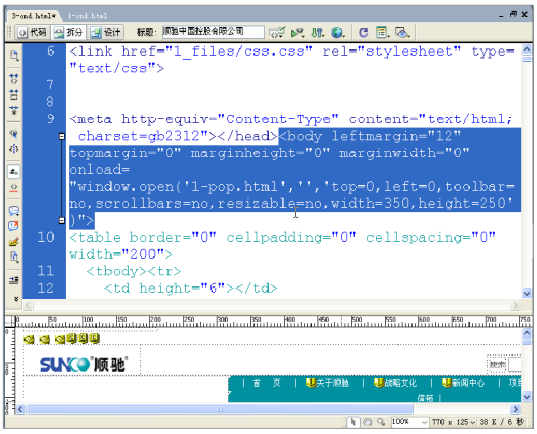
1. 范例代码

```
<html>
<head>
…
</head>
<body leftmargin="12"
topmargin="0" marginheight="0"
marginwidth="0"
onload="window.open('1-pop.
html','','top=0,left=0,toolb-
ar=no,
scrollbars=no,resizable=no,
width=350,height=250')">
…
</body>
</html>
```

2. 范例解释

使用onload事件调用脚本，表示当页面载入时执行（调用Window窗口对象的open方法，打开1-pop.html，宽度为350像素，高度为250像素，距左为0，距顶为0，没有工具条和滚动条，不能改变窗口大小）。

3. 代码和设计



4. 显示结果

脚本链接

可以用一条或者多条JavaScript语句来取代在一个文档中的任何传统的URL引用。这样当浏览器引用这个URL时，浏览器就会执行JavaScript代码。

原始文件	Sample\Ch13\1\4.html
最终文件	Sample\Ch13\1\4-end.html
学习要点	使用脚本链接

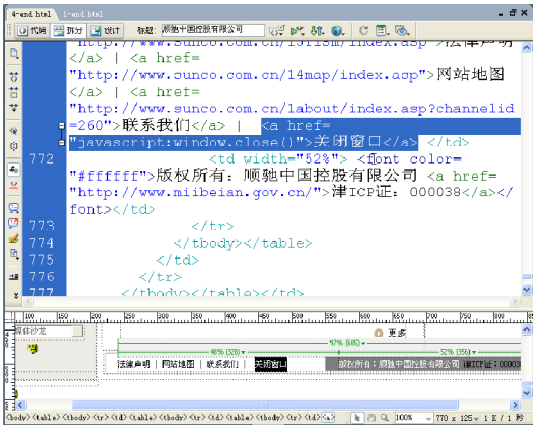
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<a href="javascript:window.close()">关闭窗口</a>
…
</body>
</html>
```

2. 范例解释

声明对文字“关闭窗口”链接JavaScript脚本，执行window.close()语句，使窗口被关闭。

3. 代码和设计



4. 显示结果

3 JavaScript内在事件

事件是浏览器响应用户操作的机制，JavaScript的事件处理功能可以改变浏览器响应这些操作的标准方式，这样就可以开发更具交互性、响应和更易使用的Web页面。常用的内在事件如下表所示。

事件	描述
onBlur	失去当前输入焦点时
onChange	因修改而失去当前输入焦点时
onClick	单击时
onFocus	得到当前输入焦点时
onLoad	载入时

（续表）

事件	描述
onMouseOut	鼠标移出时
onMouseOver	鼠标经过时
onReset	复位表单时
onSelect	文本域中的文本被选中时
onSubmit	提交表单时
onUnload	退出载入时

onBlur 事件

onBlur 事件就是当光标离开文本框时发生的事件。

原始文件	Sample\Ch13\2\1.html
最终文件	Sample\Ch13\2\1-end.html
学习要点	使用onBlur事件

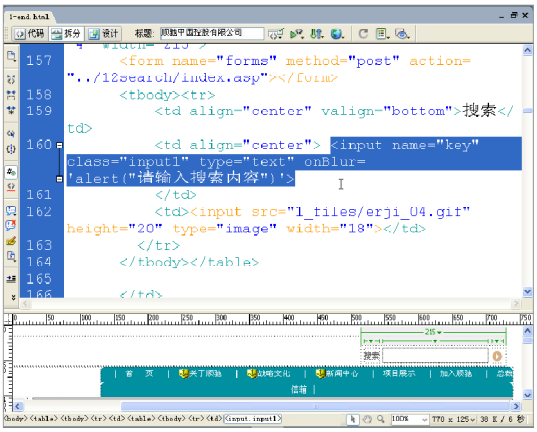
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<input name="key" class="input1" type="text" onBlur='alert("请输入搜索内容")'>
-
</body>
</html>
```

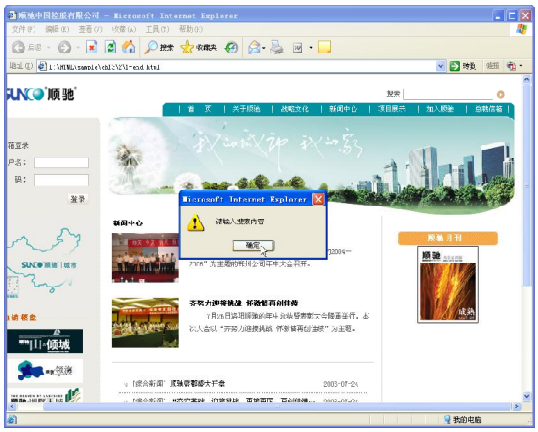
2. 范例解释

对名称为 key 的单行文本框，使用 input1 样式。调用 onBlur 事件，当光标离开“搜索”文本框后，弹出“请输入搜索内容”的提示框。

3. 代码和设计



4. 显示结果



onChange事件

onChange 事件就是当文本框的内容改变时发生的事件。

原始文件	Sample\Ch13\2\2.html
最终文件	Sample\Ch13\2\2-end.html
学习要点	使用onChange事件

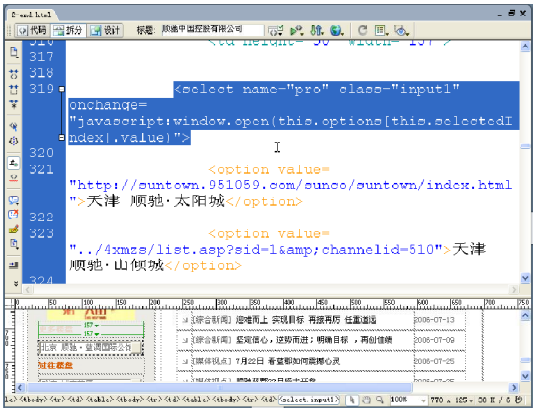
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<select name="pro" class="input1"
onchange="javascript:window.open
(this.options[this.
selectedIndex].value)">
…
</body>
</html>
```

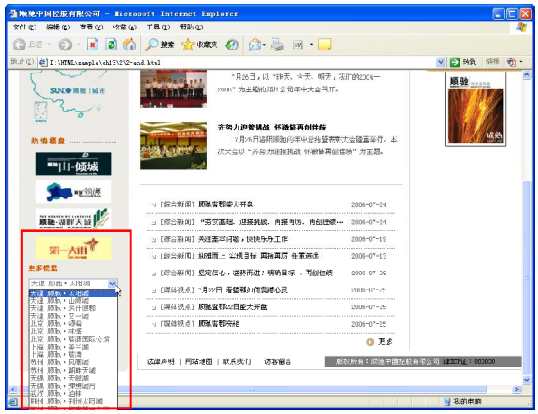
2. 范例解释

对名称为pro的单行文本框，使用input1样式。声明当改变下拉菜单内容后，调用JavaScript脚本，打开窗口显示所选项的页面。

3. 代码和设计



4. 显示结果



onClick事件

当用户单击鼠标按钮时，就会产生onClick事件。

原始文件	Sample\Ch13\2\3.html
最终文件	Sample\Ch13\2\3-end.html
学习要点	使用onClick事件

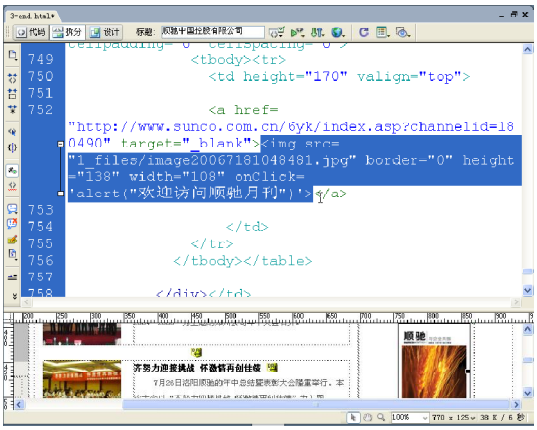
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<img src="1_files/
image20067181048481.jpg" bor
der="0" height="138" width="108"
onClick='alert("欢迎访问顺驰月刊")
'>
-
</body>
</html>
```

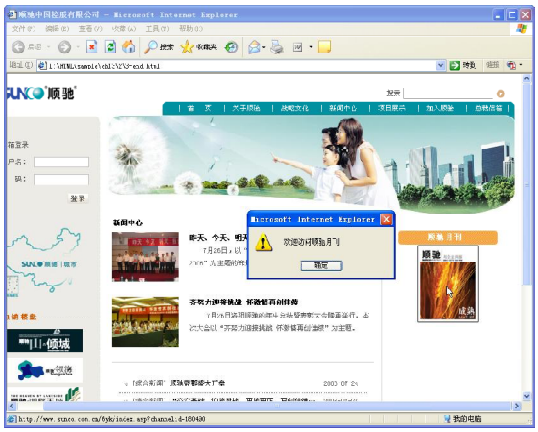
2. 范例解释

对1_files文件夹下的image20067181048481.jpg图片，设置高度为138像素，宽度为108像素，边框为0。声明当鼠标单击此图片后，弹出“欢迎访问顺驰月刊”的提示框。

3. 代码和设计



4. 显示结果



onFocus 事件

onFocus 事件就是当光标落在文本框中时发生的事件。

原始文件	Sample\Ch13\2\4.html
最终文件	Sample\Ch13\2\4-end.html
学习要点	使用onFocus事件

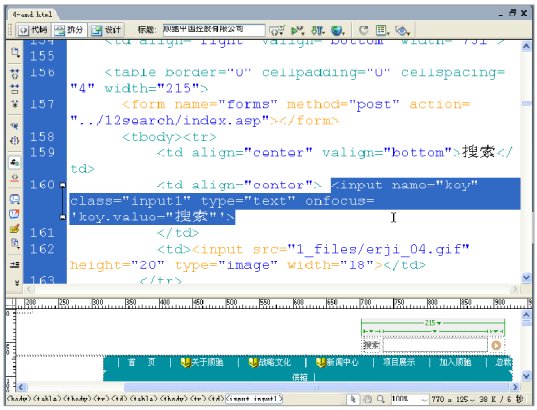
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<input name="key" class="input1"
type="text" onfocus='key.value="
搜索"'>
…
</body>
</html>
```

2. 范例解释

对名称为key的单行文本框，使用input1样式。当光标落在文本框中时，设置当前文本框的值为“搜索”。

3. 代码和设计



4. 显示结果

onLoad事件

onLoad事件是显示当前的网页时发生的事件。

原始文件	Sample\Ch13\2\5.html
最终文件	Sample\Ch13\2\5-end.html
学习要点	使用onLoad事件

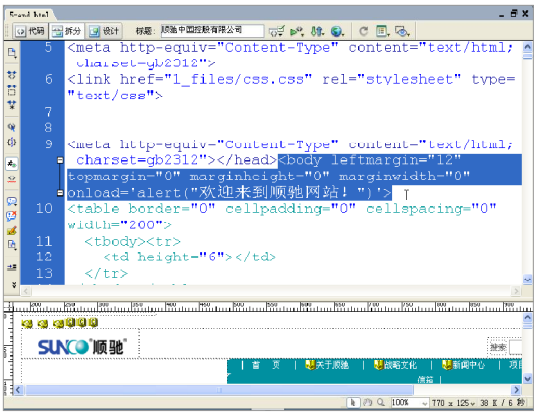
1. 范例代码

```
<html>
<head>
…
</head>
<body leftmargin="12"
topmargin="0" marginheight="0"
marginwidth="0" onload='alert("
欢迎来到顺驰网站！")'>
…
</body>
</html>
```

2. 范例解释

声明页面左边距为12像素，顶边距为0，边缘宽度和边缘高度都为0。当页面载入时，弹出“欢迎来到顺驰网站！”的提示框。

3. 代码和设计



4. 显示结果

onMouseOver 事件

onMouseOver 事件是指当光标移动到页面元素上方时发生的事件。

在滚动字幕中 onMouseOver 事件用于设置光标移动到滚动字幕时的动作，常设置为停止滚动，例如在 onMouseOver 后面输入“this.stop()”。动作上是 onMouseOver 事件触发后，再触发 onMouseOut 事件。

原始文件	Sample\Ch13\2\6.html
最终文件	Sample\Ch13\2\6-end.html
学习要点	使用onMouseOver事件

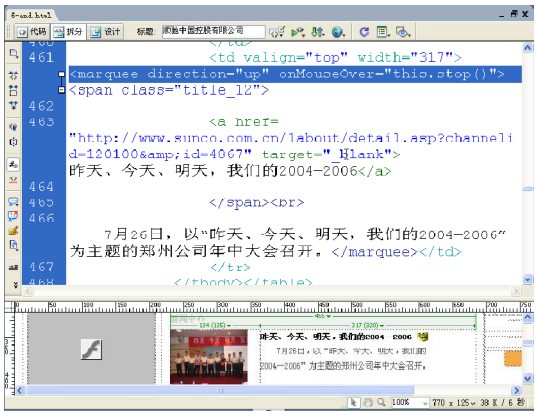
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<marquee direction="up"
onMouseOver="this.stop()">
-
</marquee>
-
</body>
</html>
```

2. 范例解释

声明方向向上的滚动文字。当光标上移到文字上面时，停止移动。

3. 代码和设计



4. 显示结果

onMouseOut事件

onMouseOut 事件是指当光标离开页面元素上方时发生的事件。

在滚动字幕中 onMouseOut 事件设置光标离开滚动字幕时的动作，常设置为开始滚动。例如在 onMouseOut 后面输入“this.start()”。它常和 onMouseOver 一起使用，但是当光标从页面元素上移动时，并不是 100% 能触发它们。当光标移动很快时，可能不会触发这两个事件。

原始文件	Sample\Ch13\2\7.html
最终文件	Sample\Ch13\2\7-end.html
学习要点	使用onMouseOut事件

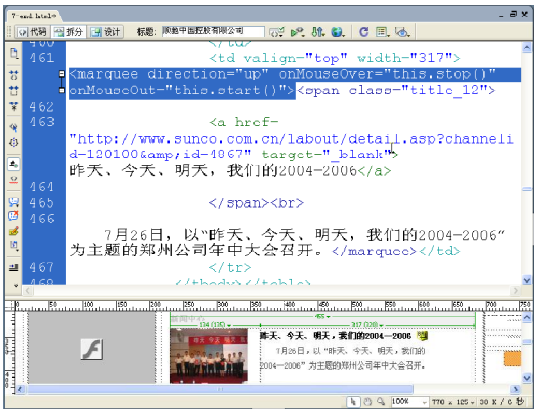
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<marquee direction="up"
onMouseOver="this.stop()"
onMouseOut="this.start()">
…
</marquee>
…
</body>
</html>
```

2. 范例解释

声明方向向上的滚动文字。当光标上移到文字上方时，停止移动；当光标离开文字时，继续开始滚动。

3. 代码和设计



4. 显示结果

onReset 事件

onReset 事件是指当重置表单时发生的事件。

onReset 出现于一个表单被重填时，只用于 Form 元素，它将清除表单中的所有输入，并将各输入域的值设为原来的缺省值。通过在事件处理程序中返回 false 值（Return False）可以阻止表单重置。

原始文件	Sample\Ch13\2\8.html
最终文件	Sample\Ch13\2\8-end.html
学习要点	使用onReset事件

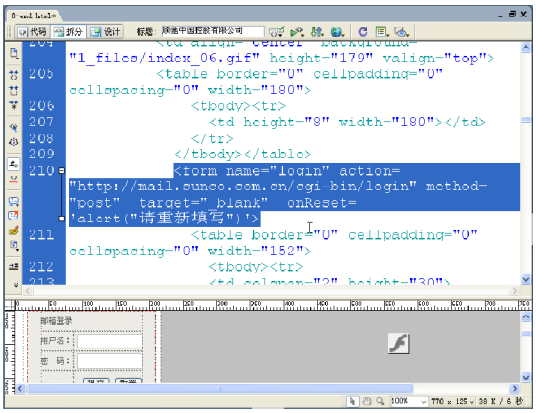
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<form name="login" action="http:/
/mail.sunco.com.cn/cgi-bin/ login"
method="post"
target="_blank"   onReset='alert
(" 请重新填写")'>
-
</form>
</body>
</html>
```

2. 范例解释

声明名称为login的表单，提交地址为http://mail.sunco.com.cn/cgi-bin/login，方法为post。当重置表单时，弹出“请重新填写”的提示框。

3. 代码和设计



4. 显示结果

onSelect事件

onSelect事件就是当文本框的内容被选中时发生的事件。

由于onSelect事件出现在用户在文字域选中文本时，因此它可以和Input和Textarea元素一起使用。

原始文件	Sample\Ch13\2\9.html
最终文件	Sample\Ch13\2\9-end.html
学习要点	使用onSelect事件

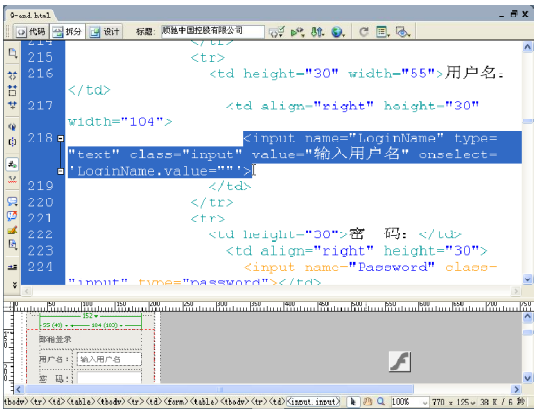
1. 范例代码

```
<html>
<head>
…
</head>
<body>
…
<input name="LoginName"
type="text" class="input" value="
输入用户名" onselect='LoginName.
value=""'>
…
</body>
</html>
```

2. 范例解释

声明名称为LoginName，使用input样式，初始值为“输入用户名”的单行文本框。当选中文本域中内容时，清空文本域文字。

3. 代码和设计



4. 显示结果

onSubmit 事件

onSubmit 事件就是当提交表单时发生的事件。

用户可以使用onSubmit事件来验证表单的有效性，通过在事件处理程序中返回false值（Return False）可以阻止表单提交。

原始文件	Sample\Ch13\2\10.html
最终文件	Sample\Ch13\2\10-end.html
学习要点	使用onSubmit事件

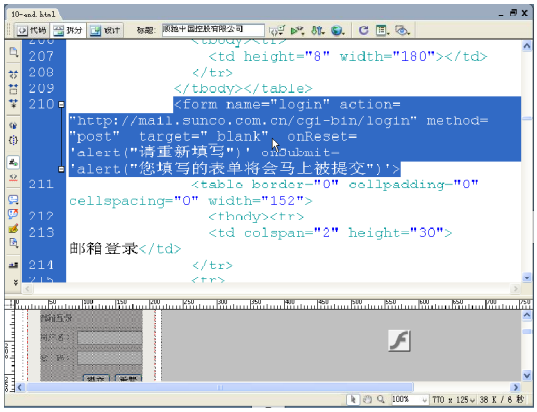
1. 范例代码

```
<html>
<head>
-
</head>
<body>
-
<form name="login" action="http://mail.sunco.com.cn/cgi-bin/login" method="post" target="_blank" onReset='alert("请重新填写")' onSubmit='alert("您填写的表单将会马上被提交")'>
-
</body>
</html>
```

2. 范例解释

声明名称为login的表单，提交地址为http://mail.sunco.com.cn/cgi-bin/login，方法为post。当提交表单后，弹出“您填写的表单将会马上被提交”的提示框。

3. 代码和设计



4. 显示结果

onUnload事件

onUnLoad 事件是指当前网页被关闭时发生的事件。

原始文件	Sample\Ch13\2\11.html
最终文件	Sample\Ch13\2\11-end.html
学习要点	使用onUnload事件

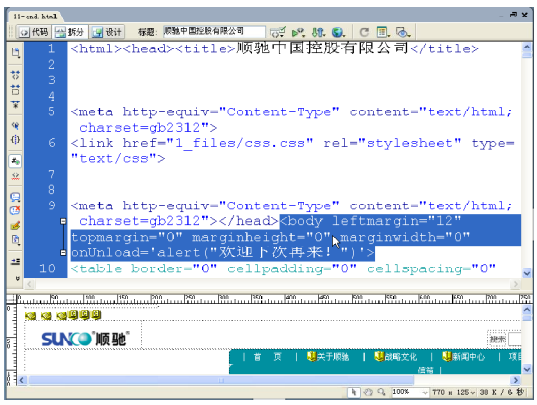
1. 范例代码

```
<html>
<head>
…
</head>
<body leftmargin="12"
topmargin="0" marginheight="0"
marginwidth="0" onUnload='alert
("欢迎下次再来！")'>
…
</body>
</html>
```

2. 范例解释

声明页面左边距为12像素，顶边距为0，边缘宽度和边缘高度都为0。当离开当前页面后，弹出“欢迎下次再来！”的提示框。

3. 代码和设计



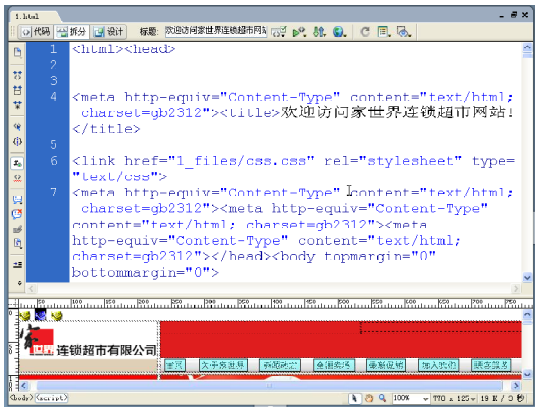
4. 显示结果

4 实例制作

下面通过一个实例练习页面中JavaScript脚本的调用以及使用JavaScript事件调用的方法。读者通过编写该实例将体会到具有交互性、响应性和易用性Web页面会使很多操作更加简单。

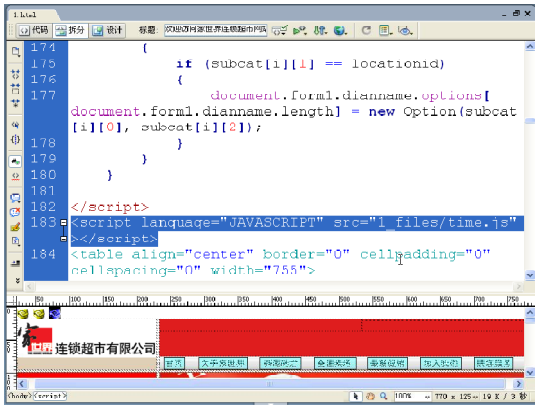
原始文件	Sample\Ch13\3\1.html
最终文件	Sample\Ch13\3\1-end.html
学习要点	使用<script>标签及JavaScript事件

1 在Dreamweaver中打开提供的1.html文件页面，单击“拆分”按钮，切换到“代码视图和设计视图”环境。



2 将光标定位到如下代码的前面：

```
<table align="center" border="0"
cellpadding="0" cellspacing="0"
width="755">
```



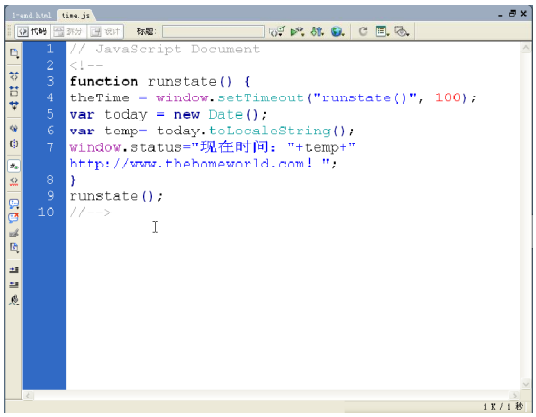
然后调用time.js脚本文件，可在代码视图中插入以下代码：

```
<script language="JAVASCRIPT"
src="1_files/time.js">
</script>
```

❶ 声明调用位于1_files文件夹中的time.js脚本文件。

3 打开time.js文件，输入如下JavaScript代码：

```
// JavaScript Document
<!--
function runstate() {
theTime = window.setTimeout
("runstate()", 100);
var today = new Date();
var temp= today.toLocaleString();
window.status="现在时间："+temp+"
http://www.thehomeworld.com！ ";
}
runstate();
//-->
```



❶ 这段代码将产生在状态栏出现当前时间和网址文字的效果。

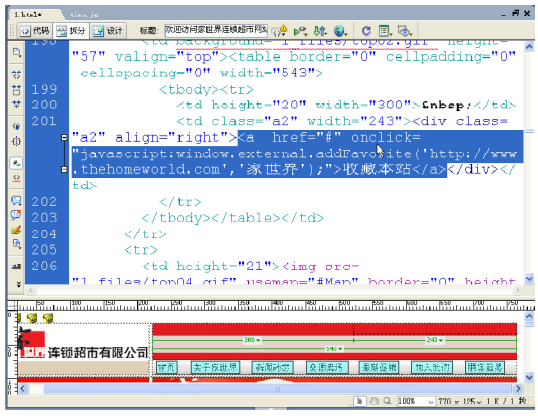
4 按下F12快捷键预览页面，可以看到状态栏中出现了显示现在的精确日期、时间和http://www.thehomeworld.com网站的动态提示信息效果了。

5 将光标定位到右上角空白单元格中如下代码的中间：

```
<div class="a2" align="right">
</div>
```

在代码视图中插入以下代码：

```
<a href="#" onClick="javascript:
window.external.addFavorite
('http://www.thehomeworld.com','
家世界');">收藏本站</a>
```



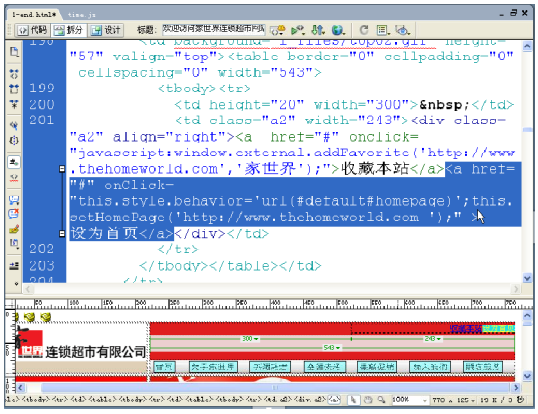
❶ 声明当单击“收藏本站”文字链接后，调用添加到收藏夹的JavaScript脚本。

6 按下F12快捷键预览页面，当单击页面“收藏本站”文字链接后，将弹出“添加到收藏夹”对话框。

7 将光标定位到刚才插入的代码的后面，在代码视图中插入以下代码：

```
<a href="#" onClick="this.style.
behavior='url(#default#homepage)
';this.setHomePage('http://www.
thehomeworld.com ');" >设为首页
</a>
```

❶ 声明当单击“设为首页”文字链接后，调用设置为浏览器首页的JavaScript脚本。



8 按下F12快捷键预览页面，当单击“设为首页”文字链接后，将弹出询问是否将当前网页设为首页的提示框。


HTML
Appendix
附 录

附录1　十六进制RGB值与命名对照表

十六进制RGB值	颜色命名	十六进制RGB值	颜色命名
#F0F8FF	aliceblue	#A00000	antiquewith
#7FFFD4	aquamarine	#F0FFFF	azure
#F5F5DC	beige	#FFE4C4	bisque
#000000	black	#FFEBCD	blanchedalmond
#0000FF	blue	#8A2BE2	blueviolet
#A52A2A	brown	#DEB887	burlywood
#5F9EA0	cadetblue	#7FFF00	chartreuse
#D2691E	chocolate	#FF7F50	coral
#C0F000	cornfloewrblue	#FFF8DC	cornsilk
#00FFFF	cyan	#00008B	darkblue
#008B8B	darkcyan	#B8860B	darkgoldenrod
#A9A9A9	darkgray	#006400	darkgreen
#DA0000	darkhaki	#8B008B	darkmagenta
#556B2F	darkolivegreen	#DA000E	darkorenge
#9932CC	darkorchid	#8B0000	darkred
#E9967A	darksalmon	#8FBC8F	darkseagreen
#483D8B	darkslateblue	#2F4F4F	darkslategray
#00CED1	darkturquoise	#9400D3	darkviolet
#FF1493	deeppink	#00BFFF	deepskyblue
#696969	dimgray	#1E90FF	dodgerblue
#B22222	firebrick	#FFFAF0	floralwhite
#228B22	forestgreen	#DCDCDC	gainsboro
#00000E	gostwhite	#FFD700	gold
#00E00D	golenrod	#808080	gray
#008000	green	#ADFF2F	greenyellow
#F0FFF0	honeydew	#FF69B4	hotpink
#CD5C5C	indianred	#FFFFF0	ivory
#F0E68C	khaki	#E6E6FA	lavender
#FFF0F5	lavenderblush	#7CFC00	lawngreen
#FFFACD	lemonchiffon	#ADD8E6	lightblue
#F08080	lightcoral	#E0FFFF	lightcyan
#0000E0	lightgodenrod	#0000E0	lightgodenrodyellow
#0000A0	lightgray	#90EE90	lightgreen
#FFB6C1	lightpink	#FFA07A	lightsalmon
#20B2AA	lightseagreen	#87CEFA	lightskyblue
#0000EB	lightslateblue	#778899	lightslategray
#B0C4DE	lightsteelblue	#FFFFE0	lightyellow
#32CD32	limegreen	#FAF0E6	linen
#FF00FF	magenta	#800000	maroon
#66CDAA	mediumaquamarine	#0000CD	mediumblue
#BA55D3	mediumorchid	#ED0000	mediumpurpul

（续表）

十六进制RGB值	颜色命名	十六进制RGB值	颜色命名
#3CB371	mediumseagreen	#7B68EE	mediumslateblue
#00FA9A	mediumspringgreen	#48D1CC	mediumturquoise
#C71585	mediumvioletred	#191970	midnightblue
#F5FFFA	mintcream	#FFE4E1	mistyrose
#FFE4B5	moccasin	#FFDEAD	navajowhite
#000080	navy	#A0B0E0	navyblue
#FDF5E6	oldlace	#6B8E23	olivedrab
#FFA500	orange	#0E0EED	orengered
#DA70D6	orchid	#A00D00	palegodenrod
#98FB98	palegreen	#AFEEEE	paleturquoise
#DB7093	palevioletred	#FFEFD5	papayawhip
#FFDAB9	peachpuff	#CD853F	peru
#FFC0CB	pink	#DDA0DD	plum
#B0E0E6	powderblue	#800080	purple
#FF0000	red	#BC8F8F	rosybrown
#4169E1	royalblue	#8B4513	saddlebrown
#FA8072	salmon	#F4A460	sandybrown
#2E8B57	seagreen	#FFF5EE	seashell
#A0522D	sienna	#87CEEB	skyblue
#6A5ACD	slateblue	#708090	slategray
#FFFAFA	snow	#00FF7F	springgreen
#4682B4	steelblue	#D2B48C	tan
#D8BFD8	thistle	#FF6347	tomato
#40E0D0	turquoise	#EE82EE	violet
#00E0ED	violetred	#F5DEB3	wheat
#000E00	hite	#F5F5F5	whitesmoke
#FFFF00	yellow	#9ACD32	yellowgreen

附录2　特殊字符的编码

ISO Latin-1 字符集

下面的表格包含了完整的 ISO Latin-1 字符集，其中还包括了Internet Explorer 4.0 及以后版本中预留的前 256 项 Unicode 字符。表格中提供了每个字符，其十进制代码，其 HTML 命名项目引用，以及简短的描述。

字符	十进制代码	命名项目	描述
---	&#00;	---	未用
---	&#01;	---	未用
---	&#02;	---	未用
---	&#03;	---	未用
---	&#04;	---	未用
---	&#05;	---	未用

（续表）

字符	十进制代码	命名项目	描述
- - -	&#06;	- - -	未用
- - -	&#07;	- - -	未用
- - -	&#08;	- - -	未用
- - -		- - -	水平跳格
- - -
- - -	换行
- - -	&#11;	- - -	未用
- - -		- - -	未用
- - -
- - -	回车
- - -	&#14;	- - -	未用
- - -	&#15;	- - -	未用
- - -	&#16;	- - -	未用
- - -	&#17;	- - -	未用
- - -	&#18;	- - -	未用
- - -	&#19;	- - -	未用
- - -	&#20;	- - -	未用
- - -	&#21;	- - -	未用
- - -	&#22;	- - -	未用
- - -	&#23;	- - -	未用
- - -	&#24;	- - -	未用
- - -	&#25;	- - -	未用
- - -	&#26;	- - -	未用
- - -	&#27;	- - -	未用
- - -	&#28;	- - -	未用
- - -	&#29;	- - -	未用
- - -	&#30;	- - -	未用
- - -	&#31;	- - -	未用
		- - -	空格
!	!	- - -	叹号
"	"	"	双引号
#	#	- - -	数值符号
$	$	- - -	美元符号
%	%	- - -	百分号
&	&	&	and 简写
'	'	- - -	单引号
(	(	- - -	左小括号
)	)	- - -	右小括号
*	*	- - -	星号
+	+	- - -	加号
,	,	- - -	逗号
-	-	- - -	连字符
。	.	- - -	句号
/	/	- - -	斜线
0	0	- - -	数字 0
1	1	- - -	数字 1

（续表）

字符	十进制代码	命名项目	描述
2	&#50;	- - -	数字 2
3	&#51;	- - -	数字 3
4	&#52;	- - -	数字 4
5	&#53;	- - -	数字 5
6	&#54;	- - -	数字 6
7	&#55;	- - -	数字 7
8	&#56;	- - -	数字 8
9	&#57;	- - -	数字 9
:	&#58;	- - -	冒号
;	&#59;	- - -	分号
<	&#60;	&lt;	小于
=	&#61;	- - -	等于
>	&#62;	&gt;	大于
?	&#63;	- - -	问号
@	&#64;	- - -	贸易 at
A	&#65;	- - -	大写 A
B	&#66;	- - -	大写 B
C	&#67;	- - -	大写 C
D	&#68;	- - -	大写 D
E	&#69;	- - -	大写 E
F	&#70;	- - -	大写 F
G	&#71;	- - -	大写 G
H	&#72;	- - -	大写 H
I	&#73;	- - -	大写 I
J	&#74;	- - -	大写 J
K	&#75;	- - -	大写 K
L	&#76;	- - -	大写 L
M	&#77;	- - -	大写 M
N	&#78;	- - -	大写 N
O	&#79;	- - -	大写 O
P	&#80;	- - -	大写 P
Q	&#81;	- - -	大写 Q
R	&#82;	- - -	大写 R
S	&#83;	- - -	大写 S
T	&#84;	- - -	大写 T
U	&#85;	- - -	大写 U
V	&#86;	- - -	大写 V
W	&#87;	- - -	大写 W
X	&#88;	- - -	大写 X
Y	&#89;	- - -	大写 Y
Z	&#90;	- - -	大写 Z
[	&#91;	- - -	左中括号
\	&#92;	- - -	反斜线
]	&#93;	- - -	右方括号

（续表）

字符	十进制代码	命名项目	描述
^	&#94;	- - -	脱字符号
_	&#95;	- - -	下划线
`	&#96;	- - -	重音符号
a	&#97;	- - -	小写 a
b	&#98;	- - -	小写 b
c	&#99;	- - -	小写 c
d	&#100;	- - -	小写 d
e	&#101;	- - -	小写 e
f	&#102;	- - -	小写 f
g	&#103;	- - -	小写 g
h	&#104;	- - -	小写 h
i	&#105;	- - -	小写 i
j	&#106;	- - -	小写 j
k	&#107;	- - -	小写 k
l	&#108;	- - -	小写 l
m	&#109;	- - -	小写 m
n	&#110;	- - -	小写 n
o	&#111;	- - -	小写 o
p	&#112;	- - -	小写 p
q	&#113;	- - -	小写 q
r	&#114;	- - -	小写 r
s	&#115;	- - -	小写 s
t	&#116;	- - -	小写 t
u	&#117;	- - -	小写 u
v	&#118;	- - - -	小写 v
w	&#119;	- - -	小写 w
x	&#120;	- - -	小写 x
y	&#121;	- - -	小写 y
z	&#122;	- - -	小写 z
{	&#123;	- - -	左大括号
|	&#124;	- - -	竖线
}	&#125;	- - -	右大括号
~	&#126;	- - -	波浪线
- - -	&#127;	- - -	未用
°			不间断空格
¡	&#161;	&iexcl;	翻转的叹号
¢	&#162;	&cent;	美分符号
£	&#163;	&pound;	英镑符号
¤	&#164;	&curren;	通用货币符号
¥	&#165;	&yen;	元符号
¦	&#166;	¦或&brkbar;	中断竖线
§	&#167;	&sect;	节符号
¨	&#168;	¨或¨	分音符号/变音符号

（续表）

字符	十进制代码	命名项目	描述
©	©	©	版权
a	ª	ª	阴性
«	«	«	左书名号
¬	¬	&not	非符号
&			软连字符
®	®	®	注册商标
¯	¯	¯ 或 &hibar;	长重音
°	°	°	度符号
±	±	±	加减号
²	²	²	上标 2
³	³	³	上标 3
´	´	´	重音符号
µ	µ	µ	微符号
¶	¶	¶	段落符号
·	·	·	中间点
¸	¸	¸	变音符号
¹	¹	¹	上标 1
º	º	º	阳性
»	»	»	右书名号
¼	¼	¼	四分之一
½	½	½	二分之一
¾	¾	¾	四分之三
¿	¿	¿	翻转的问号
À	À	À	大写 A，沉音符
Á	Á	Á	大写 A，尖音符
Â	Â	Â	大写 A，抑扬符
Ã	Ã	Ã	大写 A，颚化符
Ä	Ä	Ä	大写 A，分音符号 / 变音符号
Å	Å	Å	大写 A，带圈
Æ	Æ	Æ	大写 AE 组合
Ç	Ç	Ç	大写 C，下加符
È	È	È	大写 E，沉音符
É	É	É	大写 E，尖音符
Ê	Ê	Ê	大写 E，抑扬符
Ë	Ë	Ë	大写 E，分音符号 / 变音符号
Ì	Ì	Ì	大写 I，沉音符
Í	Í	Í	大写 I，尖音符
Î	Î	Î	大写 I，抑扬符

（续表）

字符	十进制代码	命名项目	描述
Ï	Ï	Ï	大写 I，分音符号 / 变音符号
Ð	Ð	Ð	大写 Eth，冰岛语
Ñ	Ñ	Ñ	大写 N，颚化符
Ò	Ò	Ò	大写 O，沉音符
Ó	Ó	Ó	大写 O，尖音符
Ô	Ô	Ô	大写 O，抑扬符
Õ	Õ	Õ	大写 O，颚化符
Ö	Ö	Ö	大写 O，分音符号 / 变音符号
×	×	×	乘号
Ø	Ø	Ø	大写 O，斜线
ù	Ù	Ù	大写 U，沉音符
ú	Ú	Ú	大写 U，尖音符
Û	Û	Û	大写 U，抑扬符
ü	Ü	Ü	大写 U，分音符号 / 变音符号
Ý	Ý	Ý	大写 Y，尖音符
Þ	Þ	Þ	大写 Thorn，冰岛语
ß	ß	ß	小写辅音 s，德语 sz
à	à	à	小写 a，沉音符
á	á	á	小写 a，尖音符
â	â	â	小写 a，抑扬符
ã	ã	ã	小写 a，颚化符
ä	ä	ä	小写 a，分音符号 / 变音符号
å	å	å	小写 a，带圈
æ	æ	æ	小写 ae 组合
ç	ç	ç	小写 c，下加符
è	è	è	小写 e，沉音符
é	é	é	小写 e，尖音符
ê	ê	ê	小写 e，抑扬符
ë	ë	ë	小写 e，分音符号 / 变音符号
ì	ì	ì	小写 i，沉音符
í	í	í	小写 i，尖音符
î	î	î	小写 i，抑扬符
ï	ï	ï	小写 i，分音符号 / 变音符号
ð	ð	ð	小写 eth，冰岛语

（续表）

字符	十进制代码	命名项目	描述
ñ	ñ	ñ	小写 n，颚化符
ò	ò	ò	小写 o，沉音符
ó	ó	ó	小写 o，尖音符
ô	ô	ô	小写 o，抑扬符
õ	õ	õ	小写 o，颚化符
ö	ö	ö	小写 o，分音号/变音符号
÷	÷	÷	除号
ø	ø	ø	小写 o，斜线
ù	ù	ù	小写 u，沉音符
ú	ú	ú	小写 u，尖音符
û	û	û	小写 u，抑扬符
ü	ü	ü	小写u,分音符号/变音符号
ý	ý	ý	小写 y，尖音符
Þ	þ	þ	小写 thorn，冰岛语
ÿ	ÿ	ÿ	小写 y，分音符号/变音符号

HTML 额外的命名项目

下面的表格包含了额外的命名项目、其数字字符引用及简短描述。除了左右尖括号(〈 和 〉),此页面上的项目都是使用 Lucida sans Unicode 字体渲染的。

字符	命名项目	数字字符引用	描述
拉丁语扩展-B			
f	ƒ	ƒ	拉丁小写 f 带钩子，=函数，=弗罗林，U0192 ISOtec
希腊语			
Α	Α	Α	希腊语大写字母 alpha，U0391
Β	Β	Β	希腊语大写字母 beta，U0392
Γ	Γ	Γ	希腊语大写字母 gamma，U0393 ISOgrk3
Δ	Δ	Δ	希腊语大写字母 delta，U0394 ISOgrk3
Ε	Ε	Ε	希腊语大写字母 epsilon，U0395
Ζ	Ζ	Ζ	希腊语大写字母 zeta，U0396

（续表）

字符	十进制代码	数字字符引用	描述
Η	Η	Η	希腊语大写字母 eta, U0397
Θ	Θ	Θ	希腊语大写字母 theta, U0398 ISOgrk3
Ι	Ι	Ι	希腊语大写字母 iota, U0399
Κ	Κ	Κ	希腊语大写字母 kappa, U039A
Λ	Λ	Λ	希腊语大写字母 lambda, U039B I SOgrk3
Μ	Μ	Μ	希腊语大写字母 mu, U039C
Ν	Ν	Ν	希腊语大写字母 nu, U039D
Ξ	Ξ	Ξ	希腊语大写字母 xi, U039E ISOgrk3
Ο	Ο	Ο	希腊语大写字母 omicron, U039F
Π	Π	Π	希腊语大写字母 pi, U03A0 ISOgrk3
Ρ	Ρ	Ρ	希腊语大写字母 rho, U03A1
Σ	Σ	Σ	希腊语大写字母 sigma, U03A3 ISOgrk3
Τ	Τ	Τ	希腊语大写字母 tau, U03A4
Υ	Υ	Υ	希腊语大写字母 upsilon, U03A5 ISOgrk3
Φ	Φ	Φ	希腊语大写字母 phi, U03A6 ISOgrk3
Χ	Χ	Χ	希腊语大写字母 chi, U03A7
Ψ	Ψ	Ψ	希腊语大写字母 psi, U03A8 ISOgrk3
Ω	Ω	Ω	希腊语大写字母 omega, U03A9 ISOgrk3
α	α	α	希腊语小写字母alpha, U03B1 ISOgrk3
β	β	β	希腊语小写字母 beta, U03B2 ISOgrk3
γ	γ	γ	希腊语小写字母 gamma, U03B3 ISOgrk3
δ	δ	δ	希腊语小写字母 delta, U03B4 ISOgrk3
ε	ε	ε	希腊语小写字母 epsilon, U03B5 ISOgrk3
ζ	ζ	ζ	希腊语小写字母 zeta, U03B6 ISOgrk3
η	η	η	希腊语小写字母 eta, U03B7 ISOgrk3

（续表）

字符	十进制代码	数字字符引用	描述
θ	θ	θ	希腊语小写字母 theta, U03B8 ISOgrk3
ι	ι	ι	希腊语小写字母 iota, U03B9 ISOgrk3
κ	κ	κ	希腊语小写字母 kappa, U03BA ISOgrk3
λ	λ	λ	希腊语小写字母 lambda, U03BB ISOgrk3
μ	μ	μ	希腊语小写字母 mu, U03BC ISOgrk3
ν	ν	ν	希腊语小写字母 nu, U03BD ISOgrk3
ξ	ξ	ξ	希腊语小写字母 xi, U03BE ISOgrk3
ο	ο	ο	希腊语小写字母 omicron, U03BF (新)
π	π	π	希腊语小写字母 pi, U03C0 ISOgrk3
ρ	ρ	ρ	希腊语小写字母 rho, U03C1 ISOgrk3
ς	ς	ς	希腊语小写字母 final sigma, U03C2 ISOgrk3
σ	σ	σ	希腊语小写字母 sigma, U03C3 ISOgrk3
τ	τ	τ	希腊语小写字母 tau, U03C4 ISOgrk3
υ	υ	υ	希腊语小写字母 upsilon, U03C5 ISOgrk3
φ	φ	φ	希腊语小写字母 phi, U03C6 ISOgrk3
χ	χ	χ	希腊语小写字母 chi, U03C7 ISOgrk3
ψ	ψ	ψ	希腊语小写字母 psi, U03C8 ISOgrk3
ω	ω	ω	希腊语小写字母 omega, U03C9 ISOgrk3
ϑ	ϑ	ϑ	希腊语小写字母 theta 符号, U03D1 (新)
ϒ	ϒ	ϒ	希腊语 upsilon 带钩子符号, U03D2 (新)
ϖ	ϖ	ϖ	希腊语 omega 符号, U03D6 ISOgrk3

（续表）

字符	十进制代码	数字字符引用	描述
通用符号			
•	•	•	子弹，=小黑点，U2022 ISOpub
…	…	…	水平省略号，=三点前导，U2026 ISOpub
′	′	′	撇号，=分，=英尺，U2032 ISOtech
″	″	″	双撇号，=秒，=英寸，U2033 ISOtech
‾	‾	‾	上划线，=顶上划线，U203E(新)
⁄	⁄	⁄	除法斜线，U2044 (新)
字母形状的符号			
℘	℘	℘	手写体大写 P，=幂集，=韦氏 p，U2118 ISOamso
ℑ	ℑ	ℑ	黑体大写 I，=虚部，U2111 ISOamso
ℜ	ℜ	ℜ	黑体大写 R，=实部符号，U211C ISOamso
™	™	™	商标符号，U2122 ISOnum
ℵ	ℵ	ℵ	alef 符号，=第一个无穷基数，U2135 (新)
箭头			
←	←	←	向左箭头，U2190 ISOnum
↑	↑	↑	向上箭头，U2191 ISOnum
→	→	→	向右箭头，U2192 ISOnum
↓	↓	↓	向下箭头，U2193 ISOnum
↔	↔	↔	左右箭头，U2194 ISOamsa
↵	↵	↵	向下再向左箭头，=回车，U21B5 (新)
⇐	⇐	⇐	向左双箭头，U21D0 ISOtech
⇑	⇑	⇑	向上双箭头，U21D1 ISOamsa
⇒	⇒	⇒	向右双箭头，U21D2 ISOtech
⇓	⇓	⇓	向下双箭头，U21D3 ISOamsa
⇔	⇔	⇔	左右双箭头，U21D4 ISOamsa
数学操作符			
∀	∀	∀	对于所有，U2200 ISOtech
∂	∂	∂	偏微分，U2202 ISOtech
∃	∃	∃	存在，U2203 ISOtech
∅	∅	∅	空集，=直径，U2205 ISOamso

（续表）

字符	十进制代码	数字字符引用	描述
∇	∇	∇	微分算符，=微分算子，U2207 ISOtech
∈	∈	∈	属于，U2208 ISOtech
∉	∉	∉	不属于，U2209 ISOtech
∋	∋	∋	有一个元素是，U220B ISOtech
∏	∏	∏	n 次连乘，=乘积符号，U220F ISOamsb
∑	∑	∑	n 次连加，U2211 ISOamsb
-	−	−	减号，U2212 ISOtech
*	∗	∗	星号操作符，U2217 ISOtech
√	√	√	平方根，=根号，U221A ISOtech
∝	∝	∝	成比例于，U221D ISOtech
∞	∞	∞	无穷，U221E ISOtech
∠	∠	∠	角，U2220 ISOamso
∧	∧	∧	逻辑与，=wedge，U2227 ISOtech
∨	∨	∨	逻辑或，=vee，U2228 ISOtech
∩	∩	∩	交集，=上限，U2229 ISOtech
∪	∪	∪	并集，=格结，U222A ISOtech
∫	∫	∫	积分，U222B ISOtech
∴	∴	∴	因此，U2234 ISOtech
∼	∼	∼	否定操作符，=服从于，=相似于，U223C ISOtech
≈	≅	≈	约等于，U2245 ISOtech
≅	≈	≅	几乎等于，=趋于，U2248 ISOamsr
≠	≠	≠	不等于，U2260 ISOtech
≡	≡	≡	等同于，U2261 ISOtech
≤	≤	≤	小于等于，U2264 ISOtech
≥	ge;	≥	大于等于，U2265 ISOtech
⊂	⊂	⊂	包含于，U2282 ISOtech
⊃	⊃	⊃	包含，U2283 ISOtech
⊄	&nsub	⊄	不包含于，U2284 ISOamsn
⊆	⊆	⊆	包含于或等于，U2286 ISOtech
⊇	⊇	⊇	包含或等于，U2287
⊕	⊕	⊕	带圈加号，=直接和，U2295 ISOamsb
⊗	⊗	⊗	带圈乘号，=矢量积，U2297 ISOamsb

（续表）

字符	十进制代码	数字字符引用	描述
⊥	⊥	⊥	垂直于，=正交于，U22A5 ISOtech
⋅	⋅	⋅	点运算符，U22C5 ISOamsb
杂类技术			
⌈	⌈	⌈	左上限，U2308 ISOamsc
⌉	⌉	⌉	右上限，U2309 ISOamsc
⌊	⌊	⌊	左下限，U230A ISOamsc
⌋	⌋	⌋	右下限，U230B ISOamsc
〈	⟨	〈	左尖括号，U2329 ISOtech
〉	⟩	〉	右尖括号，U232A ISOtech
几何形状			
◊	◊	◊	菱形，U25CA ISOpub
杂类符号			
♠	♠	♠	黑桃，U2660 ISOpub
♣	♣	♣	梅花，=三叶草，U2663 ISOpub
♥	♥	♥	红桃，=情人节，U2665 ISOpub
♦	♦	♦	方片，U2666 ISOpub

附录3 HTML元素速查

下面列出了 HTML 定义的元素集，按字母顺序排列。

元素	说明
a	标明超链接的起始或目的位置
acronym	标明缩写词
address	特定信息，如地址、签名、作者、此文档的原创者等
applet	在页面上放置可执行内容
area	定义一个客户端图像映射中一个超级链接区域的形状、坐标和关联 URL
b	指定文本应以粗体渲染
base	指定一个显示 URL 用于解析对于外部源的链接和引用，如图像和样式表
baseFont	设置渲染文本时作为缺省字体的基础字体值
bdo	允许作者为选定文本片断禁用双向法则
bgSound	允许页面带有背景声音或创建音轨
big	指定内含文本要以比当前字体稍大的字体显示
blockQuote	设置文本中的一段引语
body	指定文档主体的开始和结束
br	插入一个换行符
button	指定其中所含的 HTML 要被渲染为一个按钮
caption	指定表格的简要描述
center	将后面的文本和图像居中显示
cite	用斜体显示标明引言
code	指定代码范例

（续表）

元素	说明
col	指定基于列的表格缺省属性
colGroup	指定表格中一列或一组列的缺省属性
comment	标明不可见的注释
custom	代表了一个用户自定义元素
dataTransfer	提供了对于预定义的剪贴板格式的访问，以便在拖曳操作中使用
dd	在定义列表中表明定义。定义通常在定义列表中缩进
del	表明文本已经从文档中删除
dfn	表明术语的定义实例
dir	引起目录列表
div	指定渲染 HTML 的容器
dl	引起定义列表
dt	在定义列表中表明定义术语
em	强调文本，通常以斜体渲染
embed	允许嵌入任何文档
fieldSet	在字段集包含的文本和其它元素外面绘制一个方框
font	指定用于渲染所包含文本的新字体、大小和颜色
form	指定所包含控件在表单中起作用
frame	在 frameset 元素内指定单个框架
frameSet	指定一个框架集，用于组织多个框架和嵌套框架集
head	提供了关于文档的无序信息集合
hn	以标题样式渲染文本
hr	绘制水平线
html	表明文档包含 HTML 元素
HTML 注释	避免任何内含文本或 HTML 源代码被处理并在浏览器窗口中显示
i	若可用的话，指定文本应以斜体渲染
iframe	创建内嵌浮动框架
img	在文档中嵌入图像或视频剪辑
IMPORT	从元素行为中导入标签定义
input	创建各种表单输入控件
input type=button	创建按钮控件
input type=checkbox	创建复选框控件
input type=file	创建文件上载控件，该控件带有一个文本框和一个浏览按钮
input type=hidden	传输关于客户 / 服务器交互的状态信息
input type=image	创建一个图像控件，该控件单击后将导致表单立即被提交
input type=password	创建与 INPUT type=text 控件类似的单行文本输入控件，不过其中并不显示用户输入的内容
input type=radio	创建单选钮控件
input type=reset	创建一个按钮，该按钮单击后将重置表单控件设为其缺省值
input type=submit	创建一个按钮，该按钮单击后将提交表单
input type=text	创建一个单行的文本输入控件
ins	指定被插入到文档中的文本
isIndex	使浏览器显示一个对话框，提示用户输入单行文本
kbd	以固定字体渲染文本
label	为页面上的其它元素指定标签

（续表）

元素	说明
legend	在 fieldSet 对象绘制的方框内插入一个标题
il	引起列表中的一个项目
link	允许当前文档和外部文档之间建立连接
listing	以固定字体渲染文本
map	包含客户端图像映射的坐标数据
marquee	创建一个滚动的文本字幕
menu	创建一个项目的无序列表
meta	向服务器和客户端传达关于文档的隐藏信息
noBR	不换行渲染文本
noFrames	包含对于那些不支持 Frameset 元素的浏览器使用 HTML
noScript	指定要在不支持脚本的浏览器显示的 HTML
object	向 HTML 页面中插入对象
ol	绘制文本的编号列表
optGroup	允许作者对 Select 元素中的选项进行逻辑分组
option	引起 Select元素中的一个选项
p	引起一段
param	设置 Applet、Embed 或 Object 元素的属性初始值
plainText	以固定宽度字体渲染文本，不处理标签
pre	以固定宽度字体渲染文本
q	分离文本中的引语
rt	指明 Ruby 元素的注音文本
ruby	指明要放置在文本串之上或内嵌的注解或发音指南
s	以删除线字体渲染文本
samp	指定代码范例
script	为脚本指定由脚本引擎解释的脚本
select	引起列表框或下拉框
small	指定内含文本要以比当前字体稍小的字体显示
span	指定内嵌文本容器
strike	以删除线字体渲染文本
strong	以粗体渲染文本
style	指定页面的样式表
sub	指定内含文本要以下标的形式显示，通常比当前字体稍小
sup	指定内含文本要以上标的形式显示，通常比当前字体稍小
table	指定所含内容要组织成行列的表格
tBody	指明行作为表格主体
td	指定表格中的单元格
textArea	指定多行文本输入控件
tFoot	指明行作为表尾
th	指定标题列。标题列将在单元格中居中并以粗体显示
tHead	指明行作为表头
title	包含文档的标题
tr	指定表格中的一行
t	以固定宽度字体渲染文本
u	带下划线渲染文本
ul	绘制文本的项目符号列表

（续表）

元素	说明
var	定义编程变量。通常以斜体渲染
wbr	向一块 Nobr 文本中插入软换行
xml	在 HTML 页面上定义一个 XML 数据岛
xmp	以固定宽度字体渲染作为示例的字体

附录4　HTML字符集识别

字符集友好名称	首选字符集标签	别名
阿拉伯 (ASMO 708)	ASMO-708	
阿拉伯 (DOS)	DOS-720	
阿拉伯 (ISO)	iso-8859-6	arabic, csISOLatinArabic, ECMA-114, ISO_8859-6, ISO_8859-6:1987, iso-ir-127
阿拉伯 (Mac)	x-mac-arabic	
阿拉伯 (Windows)	windows-1256	cp1256
波罗的语 (DOS)	ibm775	CP500
波罗的语 (ISO)	iso-8859-4	csISOLatin4, ISO_8859-4, ISO_8859-4:1988, iso-ir-110, l4, latin4
波罗的语 (Windows)	windows-1257	
中欧 (DOS)	ibm852	cp852
中欧 (ISO)	iso-8859-2	csISOLatin2, iso_8859-2, iso_8859-2:1987, iso8859-2, iso-ir-101, l2, latin2
中欧 (Mac)	x-mac-ce	
中欧 (Windows)	windows-1250	x-cp1250
简体中文 (EUC)	EUC-CN	x-euc-cn
简体中文 (GB2312)	gb2312	chinese, CN-GB, csGB2312, csGB231280, csISO58GB231280, GB_2312-80, GB231280, GB2312-80, GBK, iso-ir-58
简体中文 (HZ)	hz-gb-2312	
简体中文 (Mac)	x-mac-chinesesimp	
繁体中文 (Big5)	big5	cn-big5, csbig5, x-x-big5
繁体中文 (CNS)	x-Chinese-CNS	
繁体中文 (Eten)	x-Chinese-Eten	
繁体中文 (Mac)	x-mac-chinesetrad	
西里尔语 (DOS)	cp866	ibm866
西里尔语 (ISO)	iso-8859-5	csISOLatin5, csISOLatinCyrillic, cyrillic, ISO_8859-5, ISO_8859-5:1988, iso-ir-144, l5
西里尔语 (KOI8-R)	koi8-r	csKOI8R, koi, koi8, koi8r
西里尔语 (KOI8-U)	koi8-u	koi8-ru
西里尔语 (Mac)	x-mac-cyrillic	
西里尔语 (Windows)	windows-1251	x-cp1251

（续表）

字符集友好名称	首选字符集标签	别名
欧罗巴	x-Europa	
德语 (IA5)	x-IA5-German	
希腊语 (DOS)	ibm737	
希腊语 (ISO)	iso-8859-7	csISOLatinGreek, ECMA-118, ELOT_928, greek, greek8, ISO_8859-7, ISO_8859-7:1987, iso-ir-126
希腊语 (Mac)	x-mac-greek	
希腊语 (Windows)	windows-1253	
希腊语，现代 (DOS)	ibm869	
希伯莱语 (DOS)	DOS-862	
希伯莱语 (ISO-Logical)	iso-8859-8-i	logical
希伯莱语 (ISO-Visual)	iso-8859-8	csISOLatinHebrew, hebrew, ISO_8859-8, ISO_8859-8:1988, ISO-8859-8, iso-ir-138, visual
希伯莱语 (Mac)	x-mac-hebrew	
希伯莱语 (Windows)	windows-1255	ISO_8859-8-I, ISO-8859-8, visual
IBM EBCDIC (阿拉伯)	x-EBCDIC-Arabic	
IBM EBCDIC (西里尔俄语)	x-EBCDIC-CyrillicRussian	
IBM EBCDIC (西里尔塞尔维亚-保加利亚)	x-EBCDIC-CyrillicSerbian Bulgarian	
IBM EBCDIC (丹麦-挪威)	x-EBCDIC-DenmarkNorway	
IBM EBCDIC (丹麦-挪威-欧洲)	x-ebcdic-denmarknorway -euro	
IBM EBCDIC (芬兰-瑞典)	x-EBCDIC-FinlandSweden	
IBM EBCDIC (芬兰-瑞典-欧洲)	x-ebcdic-finlandsweden -euro	
IBM EBCDIC (芬兰-瑞典-欧洲)	x-ebcdic-finlandsweden -euro	x-EBCDIC-France
IBM EBCDIC (法国-欧洲)	x-ebcdic-france-euro	
IBM EBCDIC (德语)	x-EBCDIC-Germany	
IBM EBCDIC (德语-欧洲)	x-ebcdic-germany-euro	
IBM EBCDIC (希腊语现代)	x-EBCDIC-GreekModern	
IBM EBCDIC (希腊语)	x-EBCDIC-Greek	
IBM EBCDIC (希伯莱语)	x-EBCDIC-Hebrew	
IBM EBCDIC (冰岛)	x-EBCDIC-Icelandic	
IBM EBCDIC (冰岛-欧洲)	x-ebcdic-icelandic-euro	
IBM EBCDIC (国际-欧洲)	x-ebcdic-international-euro	
IBM EBCDIC (意大利)	x-EBCDIC-Italy	
IBM EBCDIC (意大利-欧洲)	x-ebcdic-italy-euro	
IBM EBCDIC (日语和日语片假名)	x-EBCDIC-JapaneseAndKana	
IBM EBCDIC (日语和日本-拉丁)	x-EBCDIC-JapaneseAndJapaneseLatin	
IBM EBCDIC (日语和美国-加拿大)	x-EBCDIC-JapaneseAndUSCanada	
IBM EBCDIC (日语片假名)	x-EBCDIC-JapaneseKatakana	
IBM EBCDIC (朝鲜语和朝鲜语扩展)	x-EBCDIC-KoreanAndKorean Extended	

（续表）

字符集友好名称	首选字符集标签	别名
IBM EBCDIC (朝鲜语扩展)	x-EBCDIC-KoreanExtended	
IBM EBCDIC (多语种拉丁-2)	CP870	
IBM EBCDIC (简体中文)	x-EBCDIC-SimplifiedChinese	
IBM EBCDIC (西班牙)	X-EBCDIC-Spain	
IBM EBCDIC (西班牙-欧洲)	x-ebcdic-spain-euro	
IBM EBCDIC (泰语)	x-EBCDIC-Thai	
IBM EBCDIC (繁体中文)	x-EBCDIC-TraditionalChinese	
IBM EBCDIC (土耳其语拉丁-5)	CP1026	
IBM EBCDIC (土耳其语)	x-EBCDIC-Turkish	
IBM EBCDIC (英国)	x-EBCDIC-UK	
IBM EBCDIC (英国-欧洲)	x-ebcdic-uk-euro	
IBM EBCDIC (美国-加拿大)	ebcdic-cp-us	
IBM EBCDIC (美国-加拿大-欧洲)	x-ebcdic-cp-us-euro	
冰岛语 (DOS)	ibm861	
冰岛语 (Mac)	x-mac-icelandic	
ISCII 阿萨姆语	x-iscii-as	
ISCII 孟加拉语	x-iscii-be	
ISCII 梵文	x-iscii-de	
ISCII 瓜西拉	x-iscii-gu	
ISCII 埃纳德语	x-iscii-ka	
ISCII 马来西亚语	x-iscii-ma	
ISCII 奥里雅	x-iscii-or	
ISCII 旁遮普语	x-iscii-pa	
ISCII 泰米尔语	x-iscii-ta	
ISCII 泰卢固语	x-iscii-te	
日语 (EUC)	euc-jp	csEUCPkdFmtJapanese, xtended_UNIX_Code_Packed_Format_for_Japanese, x-euc, x-euc-jp
日语 (JIS)	iso-2022-jp	
日语 (JIS- 允许 1 字节假名 - SO/SI)	iso-2022-jp	_iso-2022-jp$SIO
日语 (JIS- 允许 1 字节假名)	csISO2022JP	_iso-2022-jp
日语 (Mac)	x-mac-japanese	
日语(Shift-JIS)	shift_jis	csShiftJIS, csWindows31J, ms_Kanji, shift-jis,x-ms-cp932,x-sjis
朝鲜语	ks_c_5601-1987	csKSC56011987, euc-kr, iso-ir-149, korean, ks_c_5601, ks_c_5601_1987, ks_c_5601-1989, KSC_5601, KSC5601
朝鲜语 (EUC)	euc-kr	csEUCKR
朝鲜语 (ISO)	iso-2022-kr	csISO2022KR
朝鲜语 (Johab)	Johab	
朝鲜语 (Mac)	x-mac-korean	
拉丁语 3 (ISO)	iso-8859-3	csISO, Latin3, ISO_8859-3, ISO_8859-3:1988, iso-ir-109, l3, latin3
拉丁语 9 (ISO)	iso-8859-15	csISO, Latin9, ISO_8859-15, l9, latin9

（续表）

字符集友好名称	首选字符集标签	别名
挪威语 (IA5)	x-IA5-Norwegian	
OEM 美国	IBM437	437, cp437, csPC8, CodePage437
瑞典语 (IA5)	x-IA5-Swedish	
泰语 (Windows)	windows-874	DOS-874, iso-8859-11, TIS-620
土耳其语 (DOS)	ibm857	
土耳其语 (ISO)	iso-8859-9	csISO, Latin5, ISO_8859-9, ISO_8859-9:1989, iso-ir-148, l5, latin5
土耳其语 (Mac)	x-mac-turkish	
土耳其语 (Windows)	windows-1254	ISO_8859-9, ISO_8859-9:1989, iso-8859-9,iso-ir-148,latin5
Unicode	unicode	utf-16
Unicode (Big-Endian)	unicodeFFFE	
Unicode (UTF-7)	utf-7	csUnicode11UTF7, unicode-1-1-utf-7, x-unicode-2-0-utf-7
Unicode (UTF-8)	utf-8	unicode-1-1-utf-8,unicode-2-0-utf-8, x-unicode-2-0-utf-8
US-ASCII	us-ascii	ANSI_X3.4-1968, ANSI_X3.4-1986, ascii, cp367, csASCII, IBM367, ISO_646.irv:1991, ISO646-US, iso-ir-6us
越南语 (Windows)	windows-1258	
西欧 (DOS)	ibm850	
西欧 (IA5)	x-IA5	
西欧 (ISO)	iso-8859-1	cp819, csISO, Latin1, ibm819, iso_8859-1, iso_8859-1:1987, iso8859-1, iso-ir-100,l1,latin1
西欧 (Mac)	macintosh	
西欧 (Windows)	Windows-1252	ANSI_X3.4-1968, ANSI_X3.4-1986, ascii, cp367, cp819, csASCII, IBM367, ibm819, ISO_646.irv:1991, iso_8859-1, iso_8859-1:1987, ISO646-US, iso8859-1, iso-8859-1, iso-ir-100, iso-ir-6, latin1,us,us-ascii,x-ansi