第3章 CSS

3.1 CSS 概述

CSS 指的是 Cascading Style Sheets，即层叠样式表，它是一种设计网页样式及布局的技术。所谓“层叠”，实际上指的是将显示样式与显示内容分离。在 CSS 出现之前，网页设计采用 HTML 标记来定义页面文档，通过指定相应标签元素的属性值为元素设置显示样式。这种方式实现起来不仅烦琐、维护困难，而且很难做到整个网站的风格统一。为了解决设计样式和风格的问题，1997 年，W3C 在颁布 HTML4 标准的同时发布了样式表的第一个标准——CSS 1.0。CSS 1.0 较为全面地规定了文档的显示样式，包括选择器、样式属性、伪类、对象等几个部分。此后，W3C 又相继发布了 CSS 2.0、CSS 2.1。从 2010 年开始，W3C 已经开始了对 CSS 3 的研发，大部分的浏览器均支持 CSS 3，现在，下一版的 CSS 4 仍处在开发之中。

使用 CSS 可以将网页的内容和显示相分离，网页的内容仍然使用 HTML 实现，而网页的显示样式使用 CSS 来实现。使用 CSS 有许多优点，主要表现在以下方面：

- 可以方便地控制页面布局；
- 整个网站可以统一风格，只要整个网站使用统一的 CSS 文件即可；
- 网站的风格维护起来简单，只需要更改相应的 CSS 文件；
- 由于 HTML 文件基本上只包含内容，不包括样式，因此结构简化、体积更小、下载更快、更加灵活、可读性增强；
- 浏览器的界面更友好。

3.2 CSS 的基本语法

CSS 的定义是由三部分组成的，包括选择符(selector)、属性(properties)、属性的取值(value)。其语法如下：

```
selector{
  property1:value;
  property2:value;
  …
  propertyN:value
}
```

其中，selector 是选择符，最常见的选择符是 HTML 标签；property 是选择符的属性，value 为选择符的属性值。多个选择符属性之间使用分号隔开。

用户可以使用单个选择符，也可以使用多个选择符，选择符之间用逗号隔开，即将一组属性值应用于多个选择符，例如：

```
body{
    background-color:yellow
}
h1,h2,p{
    background-color:#00FF00;
    color:red
}
```

上述样式表定义了网页的背景色为黄色，h1、h2 以及 p 的背景色为绿色，文字颜色为红色。用户可以将上述样式表定义用一对＜style＞标签加入到 HTML 文件中，样式表对当前文件有效，此种用法称为内部样式表，相关示例见 css1.html。

css1.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv = "Content-Type" content = "text/html; charset = UTF-8">
<title>CSS 使用示例 - HTML 选择符</title>
<style type = "text/css">
body {
    background-color: yellow
}
h1,h2,p {
    background-color: #00FF00;
    color: red
}
</style>
</head>
<body>
  这是 body 内的文字<br/>
  <h1>这是标题 1 文字</h1>
  <h2>这是标题 2 文字</h2>
  <p>这是段落文字</p>
</body>
</html>
```

css1.html 的显示结果如图 3-1 所示。

3.2.1 CSS 选择符

选择符指的是要修改的元素。CSS 中常用的选择符有 HTML 选择符、类选择符、id 选择符。

1. HTML 选择符

所有的 HTML 标签都可以作为 CSS 的选择符，HTML 选择符之后是对应元素的属性及属性值，在指定了 HTML 选择符的样式后，当前页面中的相应元素会自动套用定义的样式。

如果属性的值是由多个单词组成的，则属性值必须加上引号才可以被识别，例如：

```
p{
  font-family: "Courier New"
}
```

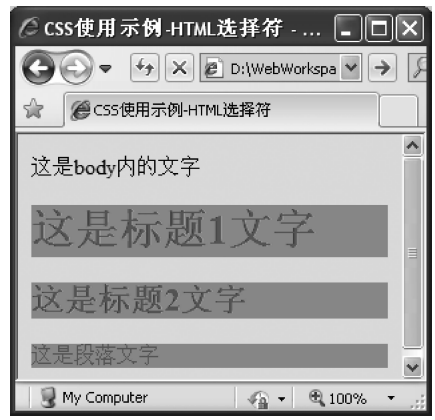

图 3-1 css1.html 的显示结果

以上代码是将段落的字体设置成 Courier New。

HTML 选择符后的属性必须是对应的标签元素的有效属性，否则属性和属性值无效，例如：

```
p{
  text-align:center;
  color:red
}
```

以上代码是将段落的对齐方式设置成居中对齐，并将文字颜色设置为红色。

2. 类选择符

如果要对页面中的元素定义不同的格式，仅使用标签选择符是不够的，还需要使用类选择符。对于类选择符，在选择符之前需要加一个实心的圆点，表示选择符的类型是类选择符。其格式如下：

```
selector.classname{
  property1:value;
  property2:value;
  …
}
```

例如：

```
p. left{
  text-align:left
}
p. center
{
  text-align:center
}
```

如果要使用类选择符定义的样式，需要在标签中利用 class 属性指定，例如：

```
<p class = "left">这是居左显示的段落</p>
<p class = "center">这是居中显示的段落</p>
```

用户在使用类选择符时，也可以不指定具体的选择符，直接使用“.”加类名，或者在“.”前添加“*”，这样可以使不同的选择符共享样式，从而提高代码的重用性。如果要对页面中的多种元素分别定义不同的格式，可以使用这种方式。其语法如下：

```
.classname{
  property1:value;
  …
  propertyN:value;
}
```

对于类选择符的使用示例，用户可参照 css2.html。

css2.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 类选择符</title>
<style type="text/css">
.one {
    color: red;
    font-size: 12pt;
}
.two {
    color: green;
    font-size: 14pt;
}
.three {
    color: #800080;
    font-size: 16pt;
}
</style>
</head>
<body>
<h1 class="one">这是引用 one 类样式的标题 1</h1>
<p class="one">这是引用 one 类样式的段落</p>
<p class="two">这是引用 two 类样式的段落</p>
<p class="three">这是引用 three 类样式的段落</p>
</body>
</html>
```

css2.html 的显示结果如图 3-2 所示。

3. id 选择符

当需要为某个元素单独设计样式时，可以使用 id 选择符。使用 id 选择符可以为每个元

素的具体对象定义不同的样式，使用 id 选择符要先为设计样式的对象定义一个 id 属性。id 选择符是唯一的，不同元素的 id 值是不能重复的。例如：

```
<div id = "top"></div>
```

然后使用以下方式定义 top 的样式：

```
#top{
  property1:value;
  property2:value;
  …
}
```

对于 id 选择符的使用示例，用户可参照 css3.html。

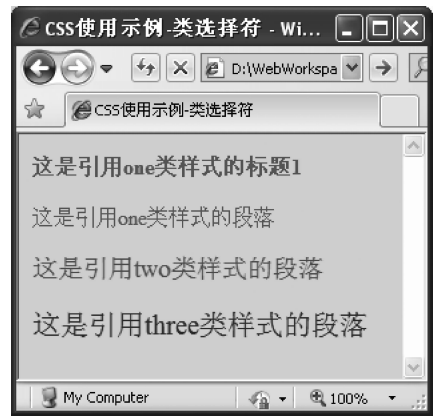

图 3-2 css2.html 的显示结果

css3.html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - id 选择符</title>
<style type = "text/css">
 #top {
     color: blue;
     font - size:18pt;
     font - family: 黑体;
     background - color: #FFB6C1;
 }
</style>
</head>
<body>
<div id = "top">
   白日依山尽<br/>
   黄河入海流<br/>
   欲穷千里目<br/>
   更上一层楼</div>
</body>
</html>
```

css3.html 的显示结果如图 3-3 所示。

在实际开发中，要尽量避免使用 id 选择符，因为 id 选择符要占用元素的 id 属性，而 id 属性一般还有其他的用途，特别是在使用 JavaScript 脚本进行验证时，需要使用 id 属性来唯一地标识标签对象。

4. 其他选择符

图 3-3 css3.html 的显示结果

除了以上几种常用的选择符之外，在 CSS 中还会用到组合选择符、包含选择符等。组合选择符是使用逗号将各个选择符隔开，使多类元素共享样式。包含选择符是使用空格分隔各个选择符，只有选择符之间是包含关系时才能使用样式。例如：

```
p em{color:red}
```

则只有<p>元素内部的元素才会应用颜色为红色的样式，其他的元素不使用此样式。在包含选择符中，规则左边的选择器一端包括两个或多个使用空格分隔的选择符。选择符之间的空格可以理解成一种运算符，解释为“在…找到…”或者“…作为…的一部分”。例如，“p em {color:red}”可以理解成作为 p 元素后代的任何 em 元素。包含选择符的使用示例可参照 css4.html。

css4.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 包含选择符</title>
<style type="text/css">
p em {
    color: red;
    font-size:16pt;
}
</style>
</head>
<body>
<p><em>p 元素中的 em 元素,红色,16pt</em></p>
<em>非 p 元素中的 em 元素,黑色,默认大小</em>
</body>
</html>
```

css4.html 的显示结果如图 3-4 所示。

3.2.2 CSS 的继承

CSS 的继承指的是当标签具有嵌套关系时，内部标签自动拥有外部标签的不冲突的样式的性质。利用 CSS 的继承特性，开发者可以先对整个网页的样式进行预设，然后在需要

特别指定样式的地方进行修改，即可达到较好的效果。继承是一种机制，它允许样式不仅可以应用于某个特定的元素，还可以应用于它的子元素。但是当子元素与父元素的属性样式出现冲突时，子元素的样式具有较高的优先级。

在 CSS 中，有些属性不允许继承，例如 border 属性没有继承性，如果 border 属性可以继承，则某些元素，例如文字的显示样式会比较奇怪。多数边框类的属性都没有继承性，例如 padding、margin、background 等。CCS 继承的使用示例可参照 css5. html。

图 3-4 css4. html 的显示结果

css5. html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4. 01 Transitional//EN" " http://www. w3. org/TR/
html4/loose. dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - CSS 的继承</title>
<style type = "text/css">
div {
    color: red;
    font - size: 10pt;
    font - weight: bold;
    font - family: 黑体;
    border: 1px solid #000;
}
p {
    color: blue;
    font - size : 12pt;
    font - style: italic;
}
em{
    color:green;
}
</style>
</head>
<body>
  <p>这是蓝色,12pt,斜体,默认宋体</p>
  <div><p>这是蓝色,12pt,斜体,加粗,黑体
  </p></div><br/>
  <div>这是红色,10pt,加粗,黑体,有边框</div><br/>
  <div>这是红色,10pt,黑体<br/>
  <em>我是 em 元素的文字,绿色文字周围无边框</em></div>
</body>
</html>
```

css5. html 的显示结果如图 3-5 所示。

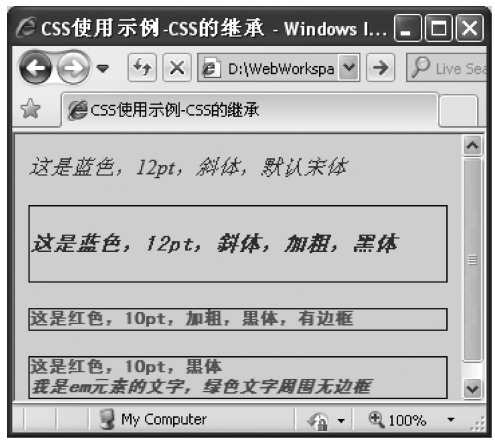

图 3-5　css5.html 的显示结果

3.3　CSS 的使用方式

在 HTML 页面中使用 CSS 主要有 4 种方法，即内嵌方式、内部样式表、使用<link>标签链接外部样式表、使用 CSS 的@import 标记导入样式表。

1. 内嵌方式

内嵌方式指的是将 CSS 规则混合在 HTML 标签中使用的方式，CSS 规则作为 HTML 标签的 style 属性值。例如：

```
<a style="font-family:黑体;font-style:italic;font-size:16pt;color:red">
    这是使用样式的超级链接
</a>
```

内嵌样式表只对其所在的标签起作用，其他的同类标签不受影响。内嵌样式表的使用示例可参照 css6.html。

css6.html：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 内嵌样式表</title>
</head>
<body>
  <a style="font-family:黑体;font-style:italic;font-size:16pt;color:red">使用内嵌样
式的超级链接</a>
  <br/>
  <br/>
  <a>普通的超级链接</a>
```

```
</body>
</html>
```

css6.html 的显示结果如图 3-6 所示。

内嵌样式表使用起来比较烦琐，虽然可以单独为每个标签设置不同的样式，但是不能方便地为同类标签设置相同的样式。

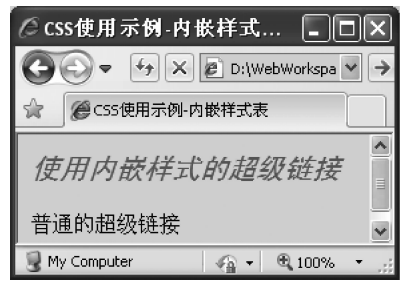

图 3-6 css6.html 的显示结果

2. 内部样式表

内部样式表指的是在 HTML 文档的<head>标签内嵌入样式表，格式如下：

```
<style type="text/css">
  selector{
    property1:value;
    property2:value;
    …
  }
  …
</style>
```

内部样式表的使用示例可参照 css7.html。

css7.html：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例-内部样式表</title>
<style type="text/css">
  body{
    font-family:楷体;
    font-size:16pt;
    color:green;
  }
  a{
    font-family:黑体;
    font-size:14pt;
    color:#FF9600;
  }
</style>
</head>
<body>
  这是正文中的字体<br/>
<p>这是段落中的文字，会继承正文字体的样式</p>
```

```
<a>这是超级链接 1</a><br/>
<a>这是超级链接 2</a>
</body>
</html>
```

css7.html 的显示结果如图 3-7 所示。

图 3-7 css7.html 的显示结果

内部样式表的作用范围是当前的 HTML 页面，样式表中定义的选择符的格式会应用到当前文件中所有的匹配选择符上。在本章示例中，css1.html 至 css5.html 都属于内部样式表的用法。

有些低版本的浏览器可能不支持＜style＞标签，此时，浏览器会忽略＜style＞标签，＜style＞标签内的内容会以文本的形式显示到页面上。为了避免此种情况的发生，可以在＜style＞标签之后添加“＜!--”，在＜/style＞标签之前添加“--＞”，例如：

```
<style type = "text/css">
<!--
  body{
    font-family:楷体;
    font-size:16pt;
    color:green;
  }
 -->
</style>
```

3. 使用＜link＞标签链接外部样式表

虽然内嵌样式表和内部样式表都可以设计 HTML 页面的样式，但是当页面较多时，其实现和维护都比较困难，而且难以将多个页面的样式统一，此时，最好使用外部样式表。外部样式表是使用一个单独的文件保存样式规则，其扩展名为“.css”，需要使用样式表的 HTML 文件链接样式表文件。链接样式表使用＜link＞标签，此标签作为＜head＞的子标签使用，指明当前 HTML 页面和链接的样式表之间的关系，其格式如下：

```
<link href = "…" rel = "stylesheet" type = "text/css"/>
```

其中：

- href 是外部样式表的路径，一般使用相对路径。
- rel 指的是被链接文件的类型，stylesheet 表示被链接的是 CSS 文件。
- type 指的是被链接文件的内容类型。

外部样式表的使用示例可参照 style.css 和 css8.html。

style.css：

```
@CHARSET "UTF-8";
body{
    font-family: 楷体;
    font-size: 16pt;
    color: green;
}
a{
    font-family: 黑体;
    font-size: 14pt;
    color: #FF9600;
}
div{
    color: red;
    font-size: 10pt;
    font-weight: bold;
    font-family: 黑体;
    border: 1px solid #000;
}
p{
    color: blue;
    font-size: 12pt;
    font-style: italic;
}
```

css8.html：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 应用示例-外部样式表</title>
<link href="css/style.css" rel="stylesheet" type="text/css"/>
</head>
<body>
  <a>我是超级链接</a>
  <div>我是 DIV</div>
  <p>我是段落</p>
</body>
</html>
```

css8.html 的显示结果如图 3-8 所示。

在使用样式表的各种方法中，外部样式表的使用最为常见，使用外部样式表有以下优点：

- 多个样式可以重复利用，一个外部 CSS 文件可以被多个网页使用；
- 修改、维护简单，当需要修改样式时，只需要修改 CSS 文件即可，不需要修改页面源

代码；

- 可以有效地减少页面的代码量，提高网页的加载速度，并且，CSS 可以驻留在缓存中，当再次使用时不需要加载；
- 整个网站的风格很容易统一，只要网站中的文件都链接同样的 CSS 文件即可。

图 3-8 css8.html 的显示结果

4. 使用 CCS 的@import 标记导入样式表

除了可以使用＜link＞标签链接外部样式表之外，还可以使用 CSS 提供的@import 标记导入样式表，其格式如下：

```
<style type="text/css">
  @import url("…");
</style>
```

使用 CSS 的@import 标记导入样式表的示例可参照 css9.html。

css9.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 使用@import 标记导入样式表</title>
  <style>
  @import url("css/style.css");
  </style>
</head>
<body>
  <a>我是超级链接</a>
  <div>我是 DIV</div>
  <p>我是段落</p>
</body>
</html>
```

css9.html 的显示结果和 css8.html 的显示结果相同，如图 3-8 所示。

使用＜link＞链接样式表和使用@import 标记导入样式表的方法类似，但是二者仍有区别，主要体现在以下几点：

- ＜link＞属于 XHTML 标签，@import 属于 CSS 标记。＜link＞标签除了可以链接 CSS 文件之外，还可以链接其他的外部资源，而@import 标记只能导入 CSS 文件。
- 当页面加载时，＜link＞链接的外部资源会被同时加载；而@import 标记导入的 CSS 文件会等到页面全部下载完成后再加载，所以使用@import 标记导入 CSS 的页面可能刚开始显示时并没有样式。

- @import 标记只有 IE5 以上的浏览器才支持，而<link>标签无此限制。

5. 各种样式表的优先级

以上各种样式表的优先级顺序如下：

内嵌样式表 > 内部样式表 > 外部样式表 > 浏览器的默认设置

内嵌样式表的实现相对来说比较复杂，基本上相当于使用标签的属性来定义样式，但是不会因为网速的问题导致样式无法应用，所以对于一些布局页面建议使用内嵌样式。对于一些布局之外的其他样式，可以使用外部样式表来完成，即使样式表由于网速问题不能应用，至少不会影响到页面的布局。外部样式表的维护和更新都比较方便。对于一些有特殊应用的样式，可以使用内部样式表或内嵌样式表来补充。

另外，如果使用外部样式表，建议使用<link>标签来链接外部样式表。

3.4 伪类和伪对象

3.4.1 伪类

伪类是一种特殊的类选择符，是能够被支持 CSS 的浏览器自动识别的特殊的选择符。伪类的最大用途就是为超级链接定义不同状态下的样式效果。

伪类的语法是在原有的选择符后加一个伪类，例如：

```
selector:pseudo-class{
  property1:value;
  property2:value;
  …
}
```

伪类是在 CSS 中已经定义好的，不能像类选择符那样使用其他名字，可以解释为对象在某个特殊状态下的样式。常用的伪类如下。

- selector:link {sRules}(sRules 表示样式规则)：设置 a 对象在未被访问前的样式，对于无 href 属性的 a 对象，此伪类不起作用。
- selector:hover {sRules}：设置 a 对象在鼠标悬停在其上时的样式。
- selector:active {sRules}：设置 a 对象被用户激活(在鼠标单击与释放之间发生的事件)时的样式。
- selector:visited {sRules}：设置 a 对象在其链接地址已被访问过时的样式。
- selector:first-child {sRules}：设置对象的第一个子对象的样式。例如，“p a:first-child {color:red}”设置段落中的第一个 a 对象的字体颜色为红色；“table td:first-child {width:100px}”设置表格中的第一个单元格的宽度为 100px。
- selecotr:first {sRules}：设置页面容器第一页使用的样式，仅用于@page 规则。例如，@page:first {margin:3cm}。
- selector:left {sRules}：设置页面容器位于装订线左边的所有页面使用的样式，仅用

于@page 规则。例如，@page:left {margin:4cm}。

- selector:lang {sRules}：设置对象使用特殊语言的内容的样式。
- selector:focus {sRules}：设置对象在成为输入焦点时的样式。

伪类的使用示例可参照 css10.html。

css10.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 伪类</title>
  <style type="text/css">
  a:link{font-size:14pt;text-decoration:underline;color:red}
  a:visited{font-size:12pt;text-decoration:none;color:green}
  a:hover{font-size:16pt;text-decoration:none;color:#FFCC00}
  a:active{font-size:18pt;text-decoration:underline;color:blue}
  </style>
</head>
<body>
  <a href="">我是超级链接 1</a><br/>
  <a href="">我是超级链接 2</a><br/>
  <a href="">我是超级链接 3</a><br/>
  <a href="">我是超级链接 4</a><br/>
</body>
</html>
```

css10.html 的显示结果如图 3-9 所示。

其中，超级连接 1 是还没有访问过的状态，超级链接 2 和超级链接 3 是已访问过的状态，超级链接 4 是将鼠标悬浮在其上时的状态。另外，将 a 对象与类选择符结合使用，可以在同一个页面上做出多组不同的超级链接效果。

图 3-9 css10.html 的显示结果

3.4.2 伪对象

与伪类相似，在 CSS 中也定义一些伪对象来设置一些特殊的文字格式。伪对象通过对插入到文档中的虚构元素进行触发来实现特定的样式。常用的伪对象如下。

- selector:after,before {sRules}：用来和 content 属性一起使用，设置在对象后、对象前（依据对象树的逻辑结构）发生的内容。
- selector:first-letter {sRules}：设置对象内的第一个字符的样式。此伪对象适用于块元素，例如<p>和 。块元素用来生成一个元素框，它会填充其父级元素

的内容，即在元素框之前和之后都存在分隔符。如果内联元素（也称为行内元素）要使用该属性，必须先设定对象的 height 或者 width 属性，或者设定 position 属性为 absolute，或者设定 display 属性为 block（显示为块）。

- selector:first-line {sRules}：设置对象内的第一行字符的样式。此伪对象适用于块元素。如果内联元素要使用该属性，其设置方法与首字符伪对象相同。伪对象的使用示例可参照 css11.html。

css11.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 伪对象的使用</title>
<style type="text/css">
div:first-line {
    font-weight: bold;
    color: red;
    font-size: 14pt;
}
p:first-letter {
    color: green;
    font-size: 18pt;
}
</style>
</head>
<body>
<div>我是 div 的第一行<br />
我是 div 的第二行</div>
<p>我是段落 p</p>
</body>
</html>
```

css11.html 的显示结果如图 3-10 所示。

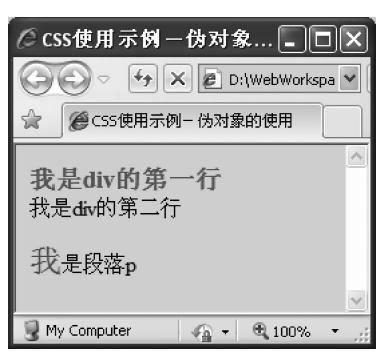

图 3-10 css11.html 的显示结果

3.5 CSS 样式属性

用户可以使用丰富的样式规则为网页中的元素设计显示样式，CSS 样式属性大致分为文字、文本、背景、定位、边框、布局、列表、光标、滤镜等属性。

1. 文字属性

在 CSS 中可以通过文字属性指定网页中文字的显示效果，主要包括文字的字体、字号、样式等，常用的文字属性及其解释如表 3-1 所示。

表 3-1 CSS 文字属性表

文字属性	功能	取值
font-family	设置字体	例如隶书、Times New Roman 等，当指定多种字体时，用逗号分隔，当字体由多个单词组成时，由双引号括起来
font-size	设置字号	absolute-size：根据对象的字体调节；relative-size：根据父对象的字体调节；length：相对于父对象字体的尺寸，不可以为负值
font-style	设置样式	normal：正常；italic：斜体；oblique：倾斜
font-weight	设置加粗	normal：正常；lighter：细体；bold：粗体；bolder：特粗体
color	设置颜色	取英文单词，或形如“#rrggbb”、“#rgb”

文字属性也可以使用 font 属性按照 font-style、font-weight、font-size、font-family 的顺序进行总体设定。例如：

```
p{font :italic bold 12pt 隶书 ;}
```

文字属性的使用示例可参照 css12.html。

css12.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv = "Content-Type" content = "text/html; charset = UTF-8">
<title>CSS 使用示例 - 文字属性的使用</title>
<style type = "text/css">
p {
    font-family: 楷体;
    font-size: 18pt;
    font-style: italic;
    font-weight: bold;
    color: blue;
}
h1{
    font-family:楷体;
    font-size:20pt;
```

```
        color:red;
    }
    </style>
    </head>
    <body>
    <h1>黄鹤楼送孟浩然之广陵</h1><br /><p align="center">
    故人西辞黄鹤楼<br />
    烟花三月下扬州<br />
    孤帆远影碧空尽<br />
    惟见长江天际流<br />
    </p>
    </body>
    </html>
```

css12.html 的显示结果如图 3-11 所示。

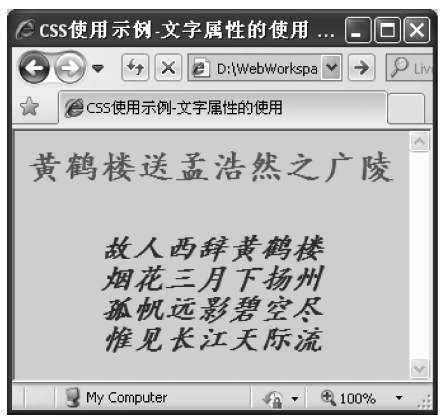

图 3-11 css12.html 的显示结果

2. 文本属性

文本属性主要用来设置块元素内的文字的显示样式，主要包括缩进、对齐方式、行高、文字间隔、文本转换和文本修饰等。常用的文本属性及其解释如表 3-2 所示。

表 3-2 CSS 文本属性表

文本属性	功能	取值
word-spacing	设置英文单词之间的间隔	默认值为 0，可取正值(间隔增大)或负值(间隔减小)
letter-spacing	设置字符或汉字之间的间隔	同上
text-indent	设置缩进	可以使用绝对单位(cm、mm、in、pt、pc)，也可以使用相对单位(em、ex、px)或者百分比
text-align	设置水平对齐方式	left：左对齐；center：居中对齐；right：右对齐；justify：两端对齐
vertical-align	设置垂直对齐方式	top：顶端对齐；bottom：底部对齐；baseline：基线对齐；middle：居中对齐

续表

文本属性	功　能	取　值
line-height	设置行高	参照 text-indent 属性
text-transform	设置文字的大小写	uppercase：所有文字大写显示；lowercase：所有文字小写显示；capitalize：每个单词的首字母大写；none：默认，无转换
text-decoration	设置文字的修饰	none：默认；underline：下划线；overline：上划线；line-through：删除线；blink：闪烁；inherit：继承父元素

文字属性的应用示例可参照 css13.html。

css13.html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - 文本属性</title>
  <style type = "text/css">
    div{
      text - indent:30px;
      text - align:left;
      line - height:25px;
      letter - spacing:5px;
      text - decoration:underline;
    }
  </style>
</head>
<body>
  <div>
  天将降大任于斯人也,必先苦其心志,劳其筋骨,饿其体肤,空乏其身,行拂乱其所为,所以动心忍
性,曾益其所不能。
  </div>
</body>
</html>
```

css13.html 的显示结果如图 3-12 所示。

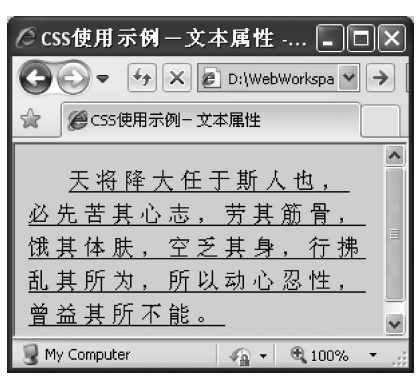

图 3-12　css13.html 的显示结果

3. 背景属性

CSS 中的背景属性用来设置 body、table 等的背景，主要包括背景色、背景图片、背景平铺方式等。常用的背景属性及其解释如表 3-3 所示。

表 3-3 CSS 背景属性表

背景属性	功能	取值
background-color	设置背景颜色	取英文单词，或形如“#rrggbb”、“#rgb”
background-image	设置背景图片	绝对路径或相对路径表示的 URL
background-repeat	设置背景图片的平铺方式	repeat-x：横向平铺；repeat-y：纵向平铺；norepeat：不平铺
background-attachment	设置背景是否随内容滚动	scroll：背景图像随内容滚动；fixed：背景图像不随内容滚动
background-position	设置背景的水平和垂直位置	left、right、top、bottom 或精确的值

用户也可以使用复合属性 background 来定义背景，例如，“background:url”等价于“background-image:url。”

另外，还可以按照 background-color、background-image、background-repeat、background-attachment、background-position 的顺序定义，例如：

```
background: #FF9600 none repeat scroll 0 % 0 %
```

关于背景属性的使用示例可参照 css14.html。

css14.html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - 背景属性</title>
  <style type = "text/css">
    body{
      color: #FF9600;
      font - family:黑体;
      font - size:18pt;
      background - image:url(image/back.jpg);
      background - repeat:repeat - y;
      background - attachment:fixed;
    }
  </style>
</head>
<body>
范仲淹： 苏幕遮<br/>
碧云天，黄叶地，秋色连波，波上寒烟翠。山映斜阳天接水，芳草无情，更在斜阳外。<br/>
黯乡魂，追旅思。夜夜除非，好梦留人睡。明月楼高休独倚，酒入愁肠，化作相思泪。<br/>
```

```
<br/>
范仲淹：渔家傲<br/>
塞下秋来风景异，衡阳雁去无留意。四面边声连角起。千嶂里，长烟落日孤城闭。<br/>
浊酒一杯家万里，燕然未勒归无计。羌管悠悠霜满地。人不寐，将军白发征夫泪。<br/>
</body>
</html>
```

css14.html 的显示结果如图 3-13 所示。

图 3-13　css14.html 的显示结果

在该页面中拖动垂直滚动条，背景图片并不随文字滚动。

4. 定位属性

利用 CSS 定位属性可以定义元素上、下、左、右的位置，也可以定义元素的叠放次序，以及元素内容超出显示范围的处理方法等。CSS 定位属性及其解释如表 3-4 所示。

表 3-4　CSS 定位属性表

定位属性	功　能	取　值
position	设置是否定位及定位方式	static：无特殊定位；relative：相对定位，对象不可层叠；absolute：绝对定位，对象可以层叠
top、right、bottom、left	设置与父对象的上、右、下、左的距离	auto：无特殊定位；自定义数值：%或长度，只有 position 取值为 absolute 或 relative，此值才有效
clip	设置对象的可视区域	auto：自动；shape：按照形状定义显示
overflow	设置当内容超过对象大小时如何显示内容	auto：根据内容确定是否需要显示滚动条；visible：默认值，内容显示在对象边框之外；hidden：超出边框的内容不显示；scroll：显示滚动条
vertical-align	设置对象的垂直对齐方式	top、middle、bottom 等
z-index	设置对象的层叠顺序	auto：遵循父对象的定位；自定义的数值：无单位的整数值，可为负数，较大值的对象会覆盖较小值的对象

如果两个对象具有相同的 z-index 值，那么将根据在文档中声明的顺序来决定覆盖顺序。定位属性的使用示例可参照 css15.html。

css15.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 定位属性</title>
</head>
<body>
<div style="position:absolute;background-color:#33CCFF;width:250px;height:100px;
display:inline">
<div style="position:relative;top:10px;left:10px;width:100px;height:100px;background-
color:#FFCC00;">春晓<br />
春眠不觉晓<br />
处处闻啼鸟<br />
夜来风雨声<br />
花落知多少</div>
<div style="position:relative;top:-40px;left:120px;width:100px;height:100px;background
-color:#FF9966">静夜思<br/>
床前明月光<br />
疑是地上霜<br />
举头望明月<br />
低头思故乡</div>
</div>
</body>
</html>
```

css15.html 的显示结果如图 3-14 所示。

5. 边框属性

利用 CSS 边框属性可以设置对象边框的颜色、样式以及宽度。用户在使用对象的边框属性之前，必须先设定对象的高度和宽度，或者将对象的 position 属性设置成 absolute。

1）边框颜色

定义边框颜色使用 border-color 属性，对象有上方、右方、下方、左方 4 个边框，对边框颜色赋值有以下几种方式。

- 4 个参数：按上方、右方、下方、左方的顺序赋值，例如“border-color:red green blue black;”。
- 一个参数：颜色作用于 4 个边框，例如“border-color:red;”。

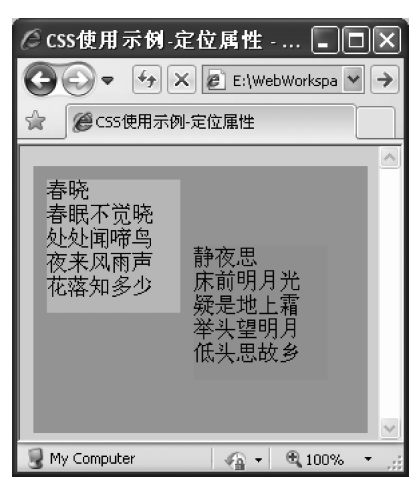

图 3-14 css15.html 的显示结果

- 两个参数：按照上下、左右的顺序赋值，例如“border-color:red green;”表示上、下边框为红色，左、右边框为绿色。
- 3 个参数：按照上、左右、下的顺序赋值，例如“border-color:red green blue;”表示上边框为红色，左、右边框为绿色，下边框为蓝色。

2）边框样式

定义边框样式使用 border-style 属性，边框样式的参数的个数及赋值方式与边框颜色类似，边框样式的取值及其解释如表 3-5 所示。

表 3-5 边框样式的取值

边框样式	取 值	边框样式	取 值
none	无边框，无论边框宽度设置为多大	double	双线边框
hidden	隐藏边框	groove	3D 凹槽边框
dotted	点线边框	ridge	菱形边框
dashed	虚线边框	inset	3D 内嵌边框
solid	实线边框	outset	3D 凸边框

3）边框宽度

边框宽度使用 border-width 来定义，宽度的取值可以是关键字或自定义的数值，宽度的参数的个数及赋值方式与边框颜色类似。用户可使用的关键字有 medium、thin、thick，其中，medium 指默认宽度；thin 指小于默认宽度；thick 指大于默认宽度。若使边框宽度只对某个边框有效，只需要在 border-width 中加入边框位置，例如“border-left-width:thin;”。边框的属性也可以使用 border 复合属性按照宽度、样式、颜色的顺序定义，例如：“border:2px solid #FF9600;”。边框属性的使用示例可参照 css16.html。

css16.html：

```
<!DOCTYPE html PUBLIC " -//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - 边框属性</title>
<style type="text/css">
div.b1{
  border:2px dashed #FF9600;
}
div.b2{
  border:4px double red;
}
</style>
</head>
<body>
<div class="b1" style="position:absolute;background-color:#33CCFF;width:250px;
height:100px;display:inline">
<div class="b2" style="position:relative;top:10px;left:10px;width:100px;height:100px;
background-color:#FFCC00;">春晓<br />
```

```
春眠不觉晓< br />
处处闻啼鸟< br />
夜来风雨声< br />
花落知多少</div>
</div>
</body>
</html>
```

css16.html 的显示结果如图 3-15 所示。

图 3-15 css16.html 的显示结果

6. 布局属性

在 HTML 中，元素的位置是依次排列的，而在 CSS 中可以利用布局属性来定义元素的排列位置。常用的布局属性有 display、visibility、margin、float 等。

1）display 属性

display 属性用来确定页面元素是否显示以及显示方式，display 属性不可继承。display 属性常用的值如下。

- block：定义元素为块对象；
- inline：定义元素为内联对象；
- list-item：定义元素为列表项目；
- none：隐藏对象，同时清除元素所占的空间。

2）visibility 属性

visibility 属性用来确定元素是否显示，该属性也不可继承，其可取值如下。

- visible：元素可见；
- hidden：元素不可见，元素所占的空间依然存在；
- collapse：在表格元素中使用时，会隐藏表格中的行和列，在非表格元素中使用时，元素不可见。

3）margin 属性

margin 属性用来定义元素的边界属性，边界属性也是不可继承的属性。其语法结构如下：

```
margin:auto|长度值|百分比值;
```

当取百分比值时，该值是相对于元素的宽度。

margin 属性有 4 个单侧边框属性，分别为 margin-top、margin-right、margin-bottom、margin-left。

4）float 属性

float 属性用来定义元素是否浮动以及浮动的方式，其可取值如下。

- none：元素不浮动；
- left：元素的浮动在左侧；
- right：元素的浮动在右侧。

对于布局属性的使用示例，用户可参照本书的后续章节。

7. 列表属性

在CSS中，列表属性用来专门控制列表的表现形式，常用的列表属性有list-style-type、list-style-image、list-style-position以及list-style属性。

1）list-style-type属性

list-style-type属性用来定义<li>列表项的显示符号，该属性可以继承，其可取值如下。

- disc：实心圆；
- circle：空心圆；
- square：实心方块；
- decimal：阿拉伯数字；
- lower-roman：小写罗马数字；
- upper-roman：大写罗马数字；
- lower-alpha：小写英文字母；
- upper-alpha：大写英文字母；
- none：不使用项目符号。

2）list-style-position属性

list-style-position属性用来定义项目符号在列表中显示的位置，该属性可以继承，其可取值如下。

- outside：项目符号放置在列表项文本以外；
- inside：项目符号放置在列表项文本以内。

3）list-style-image属性

list-style-image属性用来定义代替列表项符号的图像。其语法结构如下：

```
list-style-image:none|url;
```

用户除了可以使用以上属性之外，还可以使用list-style属性综合设置列表项的显示样式。

8. 光标属性

在CSS中，可以通过光标属性cursor来设置光标的显示图形，cursor属性的可取值如表3-6所示。

9. 滤镜属性

CSS的滤镜属性filter用于定义元素的显示效果，例如图片的透明显示等。可以应用滤镜属性的HTML标签有body、button、div、img、input、marquee、span、table、td、textarea、th以及tr。编写CSS filter代码的基本语法如下：

```
<p style="filter:滤镜名1(参数值1,参数值2,…);滤镜名2(参数值1,参数值2,…);…">
标签体内容
</p>
```

常用滤镜及其解释如表3-7所示。

表 3-6 CSS 光标属性

光标属性	说明
crosshair	十字准线
pointer|hand	手形
wait	表或沙漏
help	问号或气球
no-drop	无法释放
text	文字或编辑
move	移动
n-resize	向上改变大小
s-resize	向下改变大小
e-resize	向右改变大小
w-resize	向左改变大小
ne-resize	向上右改变大小
nw-resize	向上左改变大小
se-resize	向下右改变大小
sw-resize	向下左改变大小

表 3-7 常用滤镜表

滤镜	说明
Alpha	设置透明度
Blur	设置模糊效果
Chroma	设置某颜色透明
Dropshadow	设置投射阴影
Fliph	设置水平翻转
Flipv	设置垂直翻转
Glow	设置对象的外边界增加光效
Gray	设置灰度对象
Invert	设置底片效果
Light	设置灯光投影
Mask	设置透明膜
Shadow	设置阴影效果
Wave	利用正弦波纹打乱图片
Xray	只显示轮廓

1) Alpha 滤镜

Alpha 滤镜能使对象呈现渐变透明的效果，其格式如下：

```
filter:Alpha(opacity = …, finishOpacity = …, style = …, startx = …, starty = …, finishx = …, finishy = …)
```

其中：

- opacity 指透明度的起始值，取值范围为 0～100(0 为透明，100 为不透明，原图显示)；
- finishOpacity 指渐变的目标值；
- style 指渐变模式，取值范围为 0～3(0 表示均匀渐变，1 表示线性渐变，2 表示放射渐变，3 表示直角渐变)；
- startx 指透明渐变的起始 X 坐标；
- starty 指透明渐变的起始 Y 坐标；
- finishx 指透明渐变的终止 X 坐标；
- finishy 指透明渐变的终止 Y 坐标。

例如，filter:Alpha(opacity=0, finishopacity=50, style=2)

Alpha 滤镜的使用示例可参照 css17.html。

css17.html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/html4/loose.dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - Alpha 滤镜</title>
```

```
</head>
<body>
<div>
  <img src="image/f1.jpg" width="220"
style="filter:alpha(opacity=30,finishopacity=70,style=2)"/>
</div>
</body>
</html>
```

css17.html 的显示结果如图 3-16 所示。

图 3-16 css17.html 的显示结果

2）Blur 滤镜

Blur 滤镜能使对象呈现动感模糊的效果，其格式如下：

```
filter:blur(add=…, direction=…, strength=…)
```

其中：

- add 是布尔值，指图片是否被变成印象派的模糊效果；
- direction 指模糊的方向，它是按顺时针方向以 45°为单位进行累计的；
- strength 指受模糊影响的像素的宽度。

例如，

```
filter:blur(add=true, direction=135, strength=6)
```

Alpha 滤镜的使用示例可参照 css18.html。

css18.html：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - Blur 滤镜</title>
</head>
<body>
<div>
  <img src="image/f1.jpg" width="220" style="filter:blur(add=true,direction=45,
strength=30)"/>
</div>
</body>
</html>
```

css18.html 的显示结果如图 3-17 所示。

3）Chroma 滤镜

Chroma 滤镜可以设置对象中指定的颜色为透明色，其格式如下：

```
filter:chroma(color = …)
```

color 即指定透明的颜色，例如，

```
filter:chroma(color = red)
```

4）Dropshadow 滤镜

Dropshadow 滤镜用于为对象添加投影效果，其格式如下：

```
filter: dropshadow(color = …, offx = …, offy = …,
positive = …)
```

其中：

- color 指的是投影阴影的颜色；
- offx 指的是 X 方向的投影偏移量；
- offy 指的是 Y 方向的投影偏移量；
- positive 为布尔值，指为非透明元素建立可见的投影。

例如，

```
filter:dropshadow(color = blue, offx = 5, offy = 5, positive = 0)
```

图 3-17 css18.html 的显示结果

当对文字加载 Dropshadow 滤镜时，一般先将文字置于表格的单元格内，然后再使用 Dropshadow 滤镜。Dropshadow 滤镜对图片应用投影的效果不太理想。

5）Fliph 滤镜

Fliph 滤镜可以产生一个垂直翻转的对象，它是无参数的滤镜，使用方法简单，效果明显，例如

```
filter:filph
```

6）Flipv 滤镜

Fliph 滤镜可以产生一个水平翻转的对象，它也是无参数的滤镜，例如

```
filter:filpv
```

7）Glow 滤镜

Glow 滤镜可以使对象产生边缘发光的效果，其格式如下：

```
filter:glow(color = …, strength = …)
```

其中：

- color 指的是发光的颜色；
- strength 指的是强度，其取值为 1～255 的整数值。

例如，

```
filter:glow(color = green, strength = 5)
```

8）Gray 滤镜

Gray 滤镜可以使对象产生灰度图，即消除目标对象的所有色彩，将其以灰度级别显示。

Gray 滤镜没有参数，使用简单，例如“filter:gray”。Gray 滤镜的使用示例可参照 css19.html。

css19.html：

```
<!DOCTYPE html PUBLIC " - //W3C//DTD HTML 4.01 Transitional//EN" " http://www.w3.org/TR/
html4/loose.dtd">
<html>
<head>
<meta http - equiv = "Content - Type" content = "text/html; charset = UTF - 8">
<title>CSS 使用示例 - Gray 滤镜</title>
</head>
<body>
<div>
  <img src = "image/f1.jpg" width = "220" style = "filter:gray"/>
</div>
</body>
</html>
```

css19.html 的显示结果如图 3-18 所示。

图 3-18 css19.html 的显示结果

9) Invert 滤镜

Invert 滤镜是将对象反相，即将所有可视化属性反转，包括色彩、饱和度以及亮度值。Invert 滤镜没有参数，使用简单，例如“filter:invert”。

10) Light 滤镜

Light 滤镜是模拟光源的投射效果，可用于特殊场合。Light 滤镜是无参数滤镜，要使用 JavaScript 调用相应的方法来模拟光源的效果。一旦为对象设置了 Light 滤镜，就可以使用 Light 滤镜的方法来设置或者改变对象的属性。Light 滤镜可使用的方法如下。

- addCone(x1, y1, z1, x2, y2, iRed, iGreen, iBlue, strength, spread)：此方法的作用是添加一个锥形光源，向页面投射有方向的光源。其中，“x1,y1,z1”代表光源的位置；“x2,y2”代表光源焦点的位置；“iRed, iGreen, iBlue”代表三色定义的光源的颜色；strength 表示光源的强度；spread 代表光的发散度，即光源位置与物体表面之间的光传播角度，取值范围为 0～90。
- addAmbient(iRed, iGreen, iBlue, iStrength)：此方法的作用是对应用了 Light 滤镜的对象添加环境光源。其中，“iRed, iGreen, iBlue”指三色定义的环境光源的颜色；strength 表示光源的强度。
- addPoint(iX, iY, iZ, iRed, iGreen, iBlue, iStrength)：此方法的作用是设置点光源。其中，“iX, iY, iZ”是点光源的坐标；“iRed, iGreen, iBlue”指的是点光源的颜色；iStrength 指的是点光源的强度。
- changeColor(iLightNumber, iRed, iGreen, iBlue, iAbsolute)：此方法的作用是改变光源的颜色。其中，iLightNumber 指的是光源的标识符，“iRed, iGreen, iBlue”

是更改后的光源的颜色；iAbsolute 的值是布尔值，当为 0 时表示采用相对值替换，当为非 0 时表示采用绝对值替换，用参数指定的颜色代替原来的颜色。

- changeStrength(iLightNumber, iStrength, iAbsolute)：此方法的作用是改变光源的强度，参数解释同上。
- clear()：此方法的作用是清除所有的光源。
- moveLight(iLightNumber, iX, iY, iZ, fAbsolute)：此方法的作用是改变光源的位置。其中，“iX,iY,iZ”为光源改变位置后的坐标值；fAbsolute 为布尔值，其含义和 changeColor 方法中的 iAbsolute 参数相同。

11) Mask 滤镜

Mask 滤镜用于为对象建立一个覆盖于表面的膜。其语法如下：

```
filter:mask(color = …)
```

例如，

```
filter:mask(color = "#FF9600")
```

12) Shadow 滤镜

Shadow 滤镜用于为对象建立特定方向上的投影，其语法如下：

```
filter:shadow(color = …, direction = …)
```

其中，color 用于指定投影的颜色；direction 用于指定建立对象的投影角度，其取值及含义和 dropShadow 滤镜的参数相同。

例如，

```
filter:shadow(color = blue, direction = 45)
```

13) Wave 滤镜

Wave 滤镜用于把对象按照垂直的波纹样式打乱。其语法如下：

```
filter:wave(add = …, freq = …, lightStrength = …, phase = …, strength = …)
```

其中：

- add 为布尔值，表示是否为对象添加滤镜效果；
- freq 指波形产生的频率；
- lightStrength 指施加在波纹上的光的强度；
- phase 指波纹起始位置的偏移量，其取值范围为 0～100(0 代表 0°，100 代表 360°)；
- strength 指波纹振幅的大小。

例如，

```
filter:wave(add = 0, phase = 10, freq = 5, lightStrength = 5, strength = 3)
```

Wave 滤镜的使用示例可参照 css20.html。

css20.html：

```
<!DOCTYPE html PUBLIC "-//W3C//DTD HTML 4.01 Transitional//EN" "http://www.w3.org/TR/
html4/loose.dtd">
```

```
<html>
<head>
<meta http-equiv="Content-Type" content="text/html; charset=UTF-8">
<title>CSS 使用示例 - Wave 滤镜</title>
</head>
<body>
<div>
  <img src="image/f1.jpg" width="220" style="filter:wave(add=1,phase=10,freq=10,
lightStrength=20,strength=5)"/>
</div>
</body>
</html>
```

css20.html 的显示结果如图 3-19 所示。

图 3-19 css20.html 的显示结果

14) Xray 滤镜

Xray 滤镜用于显示对象的轮廓并将其加亮,类似于 X 光片的感觉。Xray 滤镜没有参数,使用简单,例如“filter:xray”。

小结

- CSS 是层叠样式表,它是设计网页的布局和格式的有效手段。
- CSS 的样式规则包括选择符、属性名以及属性值。
- CSS 常用的选择符有 HTML 选择符、类选择符、id 选择符、组合选择符以及包含选择符等。
- 用户可以使用内嵌样式表、内部样式表、外部样式表的方式来使用样式表,其中,外部样式表最为常见。
- 各种样式表的优先级从高到低依次为内嵌样式表、内部样式表、外部样式表、浏览器的默认样式。

- 在 CSS 中还提供了伪类和伪对象为特殊的对象提供不同状态下的样式。
- CSS 的属性包括文本属性、文字属性、背景属性、定位属性、边框属性、列表属性、布局属性、滤镜属性等。

习题

1. 下面说法错误的是(　　)。
   A. CSS 样式表可以将网页的内容和样式分离
   B. CSS 样式表可以控制页面的布局
   C. CSS 样式表可以同时更新许多网页
   D. 使用 CSS 样式表不能制作体积更小、下载更快的网页
2. 下面(　　)不属于 CSS 的使用方法。
   A. 索引式　　B. 内联式　　C. 内嵌式　　D. 外部式
3. 下面的(　　)方式使用 CSS 的优先级最高。
   A. 内联式　　B. 内嵌式
   C. 外部式　　D. 浏览器默认的样式
4. 下列关于选择符的说法正确的是(　　)。
   A. 类选择符只能应用于某一类 HTML 元素
   B. id 选择符可以重复使用
   C. 类选择符的优先级高于标签选择符
   D. 标签选择符用于设置 HTML 元素的默认样式
5. 若要在外部样式表中使用 my.css,以下用法中,(　　)是正确的。
   A. <link href="my.css" type="text/css" rel="stylesheet"/>
   B. <link src="my.css" type="text/css" rel="stylesheet"/>
   C. <link href="my.css" type="text/css"/>
   D. <include hef="my.css" type="text/css" rel="stylesheet"/>
6. CSS 的样式规则包括哪几部分? 常用的 CSS 选择符包括哪些?
7. 使用 CSS 有哪几种方法?
8. CSS 属性分为哪几类?
9. 使用<link>方式和使用@import 方式导入样式表有何区别?