第3章 CSS 基础

本章主要介绍 CSS 的基础知识，包括 CSS 样式表的使用、选择器、语法规则、常用取值与单位、常用样式和页面定位功能。在 CSS 常用样式部分介绍了关于背景、框模型、文本、字体、超链接、列表和表格等样式设置。最后介绍四种在页面上定位 HTML 元素位置的方式，包括绝对定位、相对定位、层叠效果与浮动。

本章学习目标

- 了解 CSS 的基本语法规则；
- 了解 CSS 的常见取值与单位；
- 熟悉 CSS 样式表的层叠优先级；
- 掌握 CSS 样式表的三种使用方式；
- 掌握 CSS 常用选择器的使用；
- 掌握 CSS 常用样式的使用；
- 掌握 CSS 的四种定位方法。

3.1 CSS 样式表

CSS 有三种使用方式，根据声明位置的不同分为内联样式表、内部样式表和外部样式表。

3.1.1 内联样式表

内联样式表又称为行内样式表，通过使用 style 属性为各种 HTML 元素标签添加样式，其作用范围只在指定的 HTML 元素内部。

基本语法格式如下：

```
<元素名 style="属性名称:属性值">
```

如果有多个属性需要同时添加，可用分号隔开，显示如下：

```
<元素名 style="属性名称 1:属性值 1; 属性名称 2:属性值 2; ...;属性名称 N:属性值 N">
```

例如，为某个标题标签<h1>设置样式：

```
<h1 style="color:blue; background-color:yellow">标题一</h1>
```

该声明表示设置当前<h1>和</h1>标签之间的文本字体颜色为蓝色，背景色为黄色。为方便理解本节例题，表 3-1 列出了部分常用 CSS 属性和参考值。

表 3-1 部分常用 CSS 属性和参考值

CSS 属性	含　义	参　考　值
background-color	背景色	颜色名,例如,red 表示红色
color	前景色	同上
font-size	字体大小	例如,16px 表示 16 像素大小的字体
border	边框	例如,3px solid blue 表示宽度为 3 像素的蓝色实线
width	宽度	例如,20px 表示 20 像素的宽度
height	高度	例如,100px 表示 100 像素的高度

更多属性样式请参考 3.5 节。

【例 3-1】 内联样式表的用法示例

使用内联样式表可以为多个元素分别设置各自的样式。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>CSS 内联样式表</title>
    </head>
    <body>
        <h3 style = "color:red">CSS 内联样式表</h3>
        <hr style = "border:3px dashed blue">
        <p style = "font - size:40px; background - color:yellow">
            这是一段测试文字
        </p>
    </body>
</html>
```

运行效果如图 3-1 所示。

【代码说明】

上述代码为<h3>标题标签设置了字体颜色为红色；为<hr>水平线标签设置了线条宽度为 3 像素的蓝色虚线；为<p>段落标签设置了字体大小为 40 像素,背景颜色为黄色。

图 3-1 CSS 内联样式表的使用效果

内联样式表仅适用于改变少量元素的样式,不易于批量使用和维护。试想如果存在多个元素需要设置同样的样式效果,内联样式表并不能做到批量设置,只能单独为每一个元素进行 style 声明,这显然不是一个有效率的做法,并且会造成大量重复代码。此时可以考虑使用内部样式表解决内联样式表重复定义的问题。

3.1.2 内部样式表

内部样式表通常位于<head>和</head>标签内部,通过使用<style>和</style>标签标记各类样式规则,其作用范围为当前整个文档。语法格式如下：

```
<style>
    选择器{属性名称 1:属性值 1; 属性名称 2:属性值 2; ...;属性名称 N:属性值 N }
</style>
```

这里的选择器可用于指定样式的元素标签，例如 body、p、h1-h6 等均可。例如：

```
h1{color:red }
```

该语句可以作用于整个文档，因此文档中所有的 h1 标题都将变为红色字体。

注：在 HTML4.01 版本中会看到将<style>首标签写成<style type="text/css">的形式，在 HTML5 中已简化为<style>。

如果属性内容较多，也可以分行写：

```
<style>
    选择器{
        属性名称 1:属性值 1;
        属性名称 2:属性值 2;
        …
        属性名称 N:属性值 N
    }
</style>
```

其中最后一个属性值后面是否添加分号为可选内容。一般来说，属性之间的分号用于间隔不同的属性声明，因此最后一个属性值无须添加分号。但是为了方便后续添加新的属性，也可以为最后一个属性值添加分号，这种做法不影响 CSS 样式表的正常使用。

【例 3-2】 内部样式表的用法示例

使用内部样式表可以为多个元素批量设置相同的样式。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>CSS 内部样式表</title>
6.          <style>
7.              h3 {
8.                  color: purple
9.              }
10.             p {
11.                 background - color: yellow;
12.                 color: blue;
13.                 width: 300px;
14.                 height: 50px
15.             }
16.         </style>
17.     </head>
18.     <body>
19.         <h3>CSS 内部样式表</h3>
20.         <p>
21.             内部样式表可以批量改变元素样式
22.         </p>
23.         <hr>
24.
25.         <h3>CSS 内部样式表</h3>
26.         <p>
27.             内部样式表可以批量改变元素样式
28.         </p>
29.         <hr>
30.
```

```
31.            <h3>CSS 内部样式表</h3>
32.            <p>
33.                内部样式表可以批量改变元素样式
34.            </p>
35.        </body>
36.    </html>
```

运行效果如图 3-2 所示。

图 3-2　内部样式表的使用效果

【代码说明】

上述代码中包含了标题元素<h3>和段落元素<p>各三个，因为标签名称相同，使用内部样式表可以为其统一设置样式。在内部样式表中，为<h3>标签设置了字体颜色为紫色；为<p>标签设置了背景颜色为黄色、字体颜色为蓝色，宽度 300 像素和高度 50 像素。

由图 3-2 可见，内部样式表克服了内联样式表重复定义的弊端，同一种样式声明可以批量被各类元素使用，有利于样式的后期维护和扩展。

3.1.3　外部样式表

外部样式表为独立的 CSS 文件，其后缀名为.css 或.CSS，在网页文档的首部<head>和</head>标签之间使用<link>标签对其进行引用即可作用于当前整个文档。

在 HTML5 中，对于独立 CSS 文件的引用语法格式如下：

```
<link rel = "stylesheet" href = "样式文件 URL">
```

例如，引用本地 css 文件夹中的 test.css 文件：

```
<link rel = "stylesheet" href = "css/test.css">
```

外部 CSS 文件中的内容无须使用<style></style>标签进行标记，其格式和内部样式表<style>标签内部的内容格式完全一样。

【例 3-3】　外部样式表的用法示例

在本地的 CSS 文件夹中新建一个名称为 my.css 的样式表文件，将样式要求写在该 CSS 文件中，并在 HTML 文档中对其进行引用。

HTML 文档完整代码如下：


视频讲解

```
1.    <!DOCTYPE html>
2.    <html>
3.        <head>
4.            <meta charset = "utf-8">
5.            <title>CSS 外部样式表</title>
6.            <link rel = "stylesheet" href = "css/my.css">
7.        </head>
8.        <body>
9.            <h3>CSS 外部样式表</h3>
10.           <p>
11.               使用了外部样式表规定元素样式
```

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

上述代码包含了标题元素<h3>和段落元素<p>各一个，并在首部标签<head>和</head>之间使用了引用外部样式表的方式对其进行样式的规范。

外部样式表的 CSS 文件完整代码如下：

```
h3{color:orange}
p{background-color:gray; color:white; width:300px; height:50px}
```

在外部样式表中，为<h3>标签设置字体颜色为橙色；为<p>标签设置背景颜色为灰色、字体颜色为白色、宽度 300 像素和高度 50 像素。

运行效果如图 3-3 所示。

同一个网页文档可以引用多个外部样式表。相反，当多个网页文档需要统一风格时，也可引用同一个外部样式表，该方法能极大地提高工作效率。

图 3-3 外部样式表的使用效果

3.1.4 样式表层叠优先级

内联样式表、内部样式表和外部样式表可以在同一个网页文档中被引用，它们会被层叠在一起形成一个统一的虚拟样式表。如果其中有样式条件冲突，CSS 会选择优先级别高的样式条件渲染在网页上。三种样式表的优先级别排序如表 3-2 所示。

表 3-2 样式表优先级别一览表

CSS 样式表类型	优先级别
内联样式表	最高
内部样式表	次高
外部样式表	最低

从表 3-2 可以看出，在元素内部使用的内联样式表拥有最高优先级别，在网页文档首部的内部样式表次之，引用的外部样式表优先级别最低。这也就意味着，元素是以就近原则显示离其最近的样式规则的。

注意：如果三种样式表均不存在，则网页文档会显示当前浏览器的默认效果。

【例 3-4】 样式表优先级测试


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>CSS 样式表优先级测试</title>
        <link rel = "stylesheet" href = "css/my.css">
        <style>
            p {
                background-color: cyan
            }
        </style>
    </head>
```

```
13.      <body>
14.          <h3>CSS 样式表优先级测试</h3>
15.          <p>
16.              该段落字体颜色来自外部样式表;背景颜色来自内部样式表
17.          </p>
18.
19.          <p style = "background - color:yellow">
20.              该段落字体颜色来自外部样式表;背景颜色来自内联样式表
21.          </p>
22.      </body>
23.  </html>
```

外部样式表继续使用例 3-3 的 my. css 文件,其完整代码如下:

```
h3{color:orange}
p{background - color:gray; color:white; width:300px; height:50px}
```

运行效果如图 3-4 所示。

图 3-4 样式表优先级测试效果

【代码说明】

本示例代码包含了一个标题标签<h3>和两个段落标签<p>。在首部标签<head>和</head>中对外部样式表进行了引用,同时也使用了内部样式表规定<p>标签的背景颜色为青色。在第二个段落标签<p>中设置了内联样式表规定了其背景颜色为黄色。在段落标签<p>的背景颜色上刻意使用了与描述有矛盾的规定要求,以便测试样式表的优先级。

在外部样式表中,为<h3>标签设置了字体颜色为橙色;为<p>标签设置了背景颜色为灰色、字体颜色为白色,宽度 300 像素和高度 50 像素。

只在外部样式表中声明的内容有:<h3>标签的字体颜色为橙色;<p>标签的字体颜色为白色、宽 300 像素和高 50 像素。这些样式规定没有与其他声明内容起冲突,因此可以正确显示。在内部样式表中声明的内容是:<p>标签的背景颜色为青色。该内容与外部样式表中规定的<p>标签的背景颜色为灰色矛盾,根据优先级规则,会忽略外部样式表中的相关规定,优先考虑内部样式表的内容。因此第一个<p>元素的背景颜色显示出来的效果是内部样式表规定的颜色。第二个<p>元素中包含了内联样式表的声明:规定<p>标签的背景颜色为黄色,此时与外部样式表、内部样式表的背景颜色要求均不一致,同样根据优先级规则,忽略其他规定直接将其背景颜色显示为黄色。

3.2 CSS 选择器

本节介绍了常用的几种 CSS 选择器:元素选择器、ID 选择器、类选择器、属性选择器。

3.2.1 元素选择器

在 CSS 中最常见的选择器就是元素选择器,即采用 HTML 文档中的元素名称进行样式规定。元素选择器又称为类型选择器,可以用于匹配 HTML 文档中某一个元素类型的所有元素。

例如,匹配所有的段落元素<p>,并将其背景颜色声明为灰色:

```
p{background:gray}
```

在 3.1 节中的各例题使用的均为元素选择器。

3.2.2 ID 选择器

ID 选择器使用指定的 id 名称匹配元素。如果需要为特定的某个元素进行样式设置，可以为其添加一个自定义的 id 名称，然后根据 id 名称进行匹配。ID 选择器和元素选择器语法结构类似，但是声明时需要在 id 名称前面加＃号。其语法规则如下：

```
＃id 名称{属性名称 1:属性值 1; 属性名称 2:属性值 2; ...; 属性名称 N:属性值 N }
```

例如，为某个段落元素<p>添加 id="test"：

```
<p id = "test">这是一个段落</p>
```

然后匹配上述 id="test"的段落元素<p>，并将其字体颜色声明为红色：

```
＃test{color:red}
```

【例 3-5】 ID 选择器的简单应用

为 HTML 元素设置自定义 id 名称，并使用 ID 选择器对其进行 CSS 样式设置。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>ID 选择器的简单应用</title>
6.          <style>
7.              ＃test {
8.                  background - color: cyan;          /* 设置背景颜色为青色 */
9.                  width: 100px;                      /* 设置元素宽度为 100 像素 */
10.                 height: 100px                      /* 设置元素高度为 100 像素 */
11.             }
12.         </style>
13.     </head>
14.     <body>
15.         <h3>ID 选择器的简单应用</h3>
16.         <hr />
17.         <p>
18.             该段落没有定义 id 名称
19.         </p>
20.
21.         <p id = "test">
22.             该段落自定义了 id 名称为 test
23.         </p>
24.     </body>
25. </html>
```

运行效果如图 3-5 所示。

【代码说明】

本示例代码包含两个段落元素<p>，并为第二个段落元素规定了 id="test"。在首部标签<head>和</head>之间使用了 ID 选择器对其进行样式的规范：要求设置背景颜色为

青色,段落宽度和高度均为 100 像素。

由图 3-5 可见只有设置了 id 名称的段落元素<p>实现了样式效果。这种方式就是 ID 选择器的匹配方式,一般适用于为指定的某个 HTML 元素专门设置 CSS 样式效果。

图 3-5 ID 选择器的应用效果

3.2.3 类选择器

类选择器可以将不同的元素定义为共同的样式。类选择器在声明时需要在前面加“.”号,为了和指定的元素关联使用,需要自定义一个 class 名称。其语法规则如下:

```
.class 名称{属性名称 1:属性值 1; 属性名称 2:属性值 2; ...; 属性名称 N:属性值 N }
```

例如,设置一个类选择器用于设置字体为红色:

```
.red{color:red}
```

将其使用在不同的元素上,可以显示统一的效果:

```
<h1 class = "red">这是标题,字体颜色是红色</h1>
<p class = "red">这是段落,字体颜色也是红色</p>
```

类选择器也可以将相同的元素定义为不同的样式。例如,设置两个类选择器,分别用于设置字体为红色和蓝色:

```
.red{color:red}
.blue{color:blue}
```

将其使用在相同的段落元素<p>中,可以显示不同的样式效果:

```
<p class = "red">这是段落 1,字体颜色是红色</p>
<p class = "blue">这是段落 2,字体颜色是蓝色</p>
```

类选择器也可以为同一个元素设置多个样式。

例如,设置两个类选择器,分别用于设置字体为红色和设置背景颜色为蓝色:

```
.red{color:red}
    .bgblue{background - color:blue}
```

将其使用在同一个段落元素<p>中,可以同时应用这两种样式效果:

```
<p class = "red bgblue">本段落的字体颜色是红色,背景颜色是蓝色</p>
```

【例 3-6】 类选择器的简单应用

为 HTML 元素设置自定义 class 名称,并使用类选择器对其进行 CSS 样式设置。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>类选择器的简单应用</title>
6.          <style>
```

```
7.                .red {
8.                    color: red          /* 设置字体颜色为红色 */
9.                }
10.               .blue {
11.                   color: blue         /* 设置字体颜色为蓝色 */
12.               }
13.           </style>
14.       </head>
15.       <body>
16.           <h3 class = "red">类选择器的简单应用</h3>
17.           <hr />
18.           <p class = "blue">
19.               该段落字体将设置为蓝色
20.           </p>
21.
22.           <p class = "red">
23.               该段落字体将设置为红色
24.           </p>
25.       </body>
26.   </html>
```

运行效果如图 3-6 所示。

图 3-6 类选择器的应用效果

【代码说明】

本示例代码包含了一个标题元素<h3>和两个段落元素<p>,并为<h3>元素以及第二个<p>元素规定了相同的类名称 class="red"；为第一个<p>元素规定了类名称 class="blue"。在首部标签<head>和</head>之间使用类选择器对其进行样式的规范：类名称为 red 则要求设置字体颜色为红色；类名称为 blue 则要求设置字体颜色为蓝色。

由图 3-6 可见使用类选择器的匹配方式可以为设置了相同 class 名称的标题元素<h3>和段落元素<p>实现了相同的样式效果(字体为红色)；而两个相同的段落元素<p>也可以因为类名称的不同,实现不一样的样式效果(字体为蓝色和红色)。

3.2.4 属性选择器

从 CSS2 开始引入了属性选择器,属性选择器允许基于元素所拥有的属性进行匹配。其语法规则如下：

```
元素名称[元素属性]{属性名称 1:属性值 1; 属性名称 2:属性值 2; ...; 属性名称 N:属性值 N }
```

例如,只对带有 href 属性的超链接元素<a>设置 CSS 样式：

```
a[href]{
    color: red;
}
```

上述代码表示将所有带有 href 属性的超链接元素<a>设置字体颜色为红色。

也可以根据具体的属性值进行 CSS 样式设置,例如：

```
a[href = "http://www.baidu.com"]{
    color: red;
}
```

上述代码表示将 href 属性值为 http://www.baidu.com 的超链接设置为红色字体样式。

如果不确定属性值的完整内容，可以使用[attribute~=value]的格式查找元素，表示在属性值中包含 value 关键词。例如：

```
a[href~ = "baidu"]{
    color: red;
}
```

上述代码表示将所有 href 属性值中包含 baidu 字样的超链接设置为红色字体样式。

还可以使用[attribute|=value]的格式查找元素，表示以单词 value 开头的属性值。例如：

```
img[alt| = "flower"]{
    border:1px solid red;
}
```

上述代码表示为所有 alt 属性值以 flower 字样开头的图像元素设置 1 像素宽的红色实线边框效果。

【例 3-7】 属性选择器的简单应用

使用两个图像元素<img>作为参照对比，仅为其中一个图像元素设置 alt 属性，并使用属性选择器对其进行 CSS 样式设置。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>属性选择器的简单应用</title>
6.          <style>
7.              img[alt = "balloon"] {
8.                  border: 20px solid red;
9.              }
10.         </style>
11.     </head>
12.     <body>
13.         <h3>属性选择器的简单应用</h3>
14.         <hr />
15.         <h4>为设置有 alt 属性的图像元素设置边框效果</h4>
16.         <img src = "image/balloon.jpg" alt = "balloon" />
17.         <img src = "image/balloon.jpg" />
18.     </body>
19. </html>
```

运行效果如图 3-7 所示。

【代码说明】

本示例 HTML5 代码中包含了两个图像元素<img>，并为其中第一个图像元素设置了 alt="balloon"属性。由图 3-7 可见，使用属性选择器的匹配方式可以为设置了 alt 属性的图像元素单独实现样式效果：带有 20 像素宽的红色实线边框；而另外一个图像元素<img>因为没有设置 alt 属性，因此不受任何影响。

图 3-7 属性选择器的应用效果

3.3 语法规则

3.3.1 注释语句

在内部样式表和外部样式表文件中均可以使用/*注释内容*/的形式为 CSS 进行注释，注释内容不会被显示出来。该注释以“/*”开头，以“*/”结尾，支持单行和多行注释。例如：

```
p{
  color:red;                          /*字体设置为红色*/
  background-color:yellow;            /*背景设置为黄色*/
}
/*
  这是一个多行注释
  注释可以存在于 CSS 样式表的任意位置
*/
```

3.3.2 @charset

该语法在外部样式表文件内使用，用于指定当前样式表使用的字符编码。例如：

```
@charset "utf-8";
```

该语句表示外部样式表文件使用了 UTF-8 的编码格式，一般写在外部样式表文件的第一行，并且需要加上分号结束。

3.3.3 !important

!important 用于标记 CSS 样式的使用优先级，其语法规则如下：

```
选择器{样式规则 !important;}
```

例如：

```
p{
    background-color: red !important;
```

```
    background-color: blue;
}
```

上述代码表示优先使用 background-color：red 语句，即段落元素的背景颜色设置为红色。

注意：该规则在同一对大括号内对 IE 6.0 及以下的浏览器无效。如果需要 IE 6.0 及以下浏览器兼容，需要将这两句样式代码分开写。修改后如下：

```
p{
    background-color: blue;
}
p{
    background-color: red !important;
}
```

此时可兼容 IE 6.0 及以下版本的浏览器。

3.4 CSS 取值与单位

3.4.1 数字

数字取值是在 CSS2 中规定的，有三种取值形式，如表 3-3 所示。

表 3-3 CSS 数字取值类型一览表

数字类型	发布版本	解释
<number>	CSS2	浮点数值
<integer>	CSS2	整数值
<percentage>	CSS2	百分比，写法为<number>%的形式。该数值必须有参照物才能换算出具体的数值，是一个相对值

目前所有主流浏览器都支持以上三种取值形式。

3.4.2 长度

长度取值<length>是在 CSS2 中规定的，表示方法为数值接长度单位。可用于描述文本、图像或其他各类元素的尺寸。

长度取值的单位可分为相对长度单位和绝对长度单位。相对单位的长度不是固定的，是根据参照物换算出实际长度，又可分为文本相对长度单位和视口相对长度单位。绝对长度单位的取值是固定的，例如厘米、毫米等，该取值不根据浏览器或容器的大小发生改变。

长度单位的具体情况如表 3-4 所示。

表 3-4 CSS 长度单位一览表

文本相对长度单位		
长度单位	发布版本	解释
em	CSS1	相对于当前对象内文本的字体尺寸
ex	CSS1	相对于字符 x 的高度。一般为字体正常高度的一半
ch	CSS3	数字 0 的宽度
rem	CSS3	相对于当前页面的根元素<html>规定的 font-size 字体大小属性值的倍数

续表

视口相对长度单位		
长度单位	发布版本	解　释
vw	CSS3	相对于视口的宽度。视口为均分为 100vw
vh	CSS3	相对于视口的高度。视口为均分为 100vh
vmax	CSS3	相对于视口的宽度或高度中的较大值。视口为均分为 100vmax
vmin	CSS3	相对于视口的宽度或高度中的较小值。视口为均分为 100vmin
绝对长度单位		
长度单位	发布版本	解　释
cm	CSS1	厘米(centimeters)
mm	CSS1	毫米(millimeters)
q	CSS3	四分之一毫米(quarter-millimeters),1q 相当于 0.25mm
in	CSS1	英寸(inches),1in 相当于 2.54cm
pt	CSS1	点(points),1pt 相当于 1/72in
pc	CSS1	派卡(picas),1pc 相当于 12pt
px	CSS1	像素(pixels),1px 相当于 1/96in

3.4.3 角度

角度取值<angle>是在 CSS3 中规定的,可用于描述元素变形时旋转的角度。

角度单位的具体情况如表 3-5 所示。

表 3-5 CSS 角度单位一览表

角度单位	发布版本	解　释
deg	CSS3	度(degrees),圆形环绕一周为 360deg
grad	CSS3	梯度(gradians),圆形环绕一周为 400grad
rad	CSS3	弧度(radians),圆形环绕一周为 2πrad
turn	CSS3	转、圈(turns),圆形环绕一周为 1turn

3.4.4 时间

时间取值<time>是在 CSS3 中规定的,可用于描述时间长短。

时间单位有两种情况,如表 3-6 所示。

表 3-6 CSS 时间单位一览表

时间单位	发布版本	解　释
s	CSS3	秒(seconds)
ms	CSS3	毫秒(milliseconds),1000 毫秒=1 秒

3.4.5 文本

文本常见有三种取值形式,如表 3-7 所示。

表 3-7 CSS 文本取值类型一览表

文本类型	发布版本	解　释
<string>	CSS2	字符串
<url>	CSS2	图像、文件或浏览器支持的其他任意资源的地址
<identifier>	CSS2	用户自定义的标识名称,例如,为元素自定义 id 名称等

目前所有主流浏览器都支持以上三种取值形式。

3.4.6 颜色

CSS颜色可以用于设置HTML元素的背景颜色、边框颜色、字体颜色等。本节主要介绍了网页中颜色显示的原理——RGB色彩模式和三种常用的颜色表示方式。

1. RGB色彩模式

RBG色彩模式是一种基于光学原理的颜色标准规范，也是目前运用最广泛的工业界颜色标准之一。颜色是通过对红、绿、蓝光的强弱程度不同组合叠加显示出来的，而RGB三个字母正是由红(Red)、绿(Green)、蓝(Blue)三个英文单词首字母组合而成的，代表了这三种颜色光线叠加在一起形成的各式各样的色彩。

目前的显示器大多采用了RGB色彩模式，是通过屏幕上的红、绿、蓝三色的发光极的亮度组合出不同的色彩。因此网页上的任何一种颜色都可以由一组RGB值来表示。

RGB色彩模式规定了红、绿、蓝三种光的亮度值均用整数表示，其范围是[0, 255]，共有256级，其中0为最暗，255为最亮。因此红、绿、蓝三种颜色通道的取值能组合出256×256×256=16 777 216种不同的颜色。目前主流浏览器能支持其中大约16 000多种色彩。

2. 常见颜色表示方式

在CSS中常用的颜色表示方式有：

- 使用RGB颜色的方式，例如，rgb(0,0,0)表示黑色、rgb(255,255,255)表示白色等；
- RGB的十六进制表示法，例如，#000000表示黑色、#FFFFFF表示白色等；
- 直接使用英文单词名称，例如，red表示红色、blue表示蓝色等。

1) RGB颜色

所有浏览器都支持RGB颜色表示法，使用RGB色彩模式表示颜色值的格式如下：

```
rgb(红色通道值, 绿色通道值, 蓝色通道值)
```

以上三个参数的取值范围可以是整数或者百分比的形式。取整数值时完全遵照RGB颜色标准为[0,255]的整数，数字越大，该通道的光亮就越强。

例如，希望获得红色，则将红色通道值设置为最大值255，绿色和蓝色通道值均设置为最小值0。写法如下：

```
rgb(255, 0, 0)
```

如果需要获得绿色或蓝色也是一样的原理，将相关颜色通道值设置为255，其余两个通道值设置为0即可。

因此，如果将其中某个通道颜色值设置的比其他两个值都大，则最终显示出来的颜色就偏向于这种色彩更多。当三个通道的光均为最强时，则显示出白色：rgb(255, 255, 255)。相反，当三个通道的光均为最弱时显示出黑色：rgb(0, 0, 0)。

2) 十六进制颜色

所有浏览器都支持RGB颜色的十六进制表示法，这种方法其实是把原先十进制的RGB取值转换成了十六进制，其格式如下：

```
#RRGGBB
```

以井号(#)开头，后面跟六位数，每两位代表一种颜色通道值，分别是RR(红色)、GG(绿色)和BB(蓝色)的十六进制取值。其中最小值仍然为0，最大值为FF。例如，红色的十六进制码为#FF0000，这表示红色成分被设置为最高值，其他成分为0。

十六进制码中的字母大小写均可，因此红色对应的十六进制码也可以写成＃ff0000，是同样的效果。

当每种颜色通道上的两个字符相同时，CSS 还支持将＃RRGGBB 简写成＃RGB 的形式，即每个通道的十六进制取值只占一个字符。例如，红色是＃FF0000，就可以简写为＃F00。

3）颜色名

一些常用的颜色可以使用相应的英文单词表示（例如，red 表示红色），目前所有浏览器均支持这种表示方式。W3C 组织在 CSS 颜色规范中定义了 17 种 Web 标准色，如表 3-8 所示。

表 3-8 CSS 颜色规定的 17 种标准色

颜色名称	中文名	RGB 十六进制	RGB 十进制
aqua/cyan	青色	＃00FFFF	0，255，255
black	黑色	＃000000	0，0，0
blue	蓝色	＃0000FF	0，0，255
fuchsia	洋红色	＃FF00FF	255，0，255
gray	灰色	＃808080	128，128，128
green	调和绿	＃008000	0，128，0
lime	绿色	＃00FF00	0，255，0
maroon	栗色	＃800000	128，0，0
navy	藏青色	＃000080	0，0，128
olive	橄榄色	＃808000	128，128，0
orange	橙色	＃FFA500	255，165，0
purple	紫色	＃800080	128，0，128
red	红色	＃FF0000	255，0，0
silver	银色	＃C0C0C0	192，192，192
teal	鸭翅绿	＃008080	0，128，128
white	白色	＃FFFFFF	255，255，255
yellow	黄色	＃FFFF00	255，255，0

CSS3 在这 17 种标准色的基础上又新增了 130 多种颜色名称，具体内容可查看附录 C CSS 颜色对照表。目前共计 140 种颜色名称，由于存在两对组异名同色的情况，实际的颜色共计 138 种。

【例 3-8】 CSS 颜色的简单应用

对 RGB、十六进制码和颜色名这三种不同的 CSS 颜色表示方式进行综合应用。

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 颜色的简单应用</title>
        <style>
        /* 设置字体颜色为红色 */
            .red {
                color: #FF0000
            }
        /* 设置字体颜色为蓝色 */
```


视频讲解

```
12.                 .blue {
13.                     color: rgb(0,0,255)
14.                 }
15.             /* 设置字体颜色为橙色 */
16.                 .orange {
17.                     color: orange
18.                 }
19.             </style>
20.         </head>
21.         <body>
22.             <h3>CSS 颜色的简单应用</h3>
23.             <hr />
24.             <p class = "red">
25.                 该段落字体将设置为红色
26.             </p>
27.
28.             <p class = "blue">
29.                 该段落字体将设置为蓝色
30.             </p>
31.
32.             <p class = "orange">
33.                 该段落字体将设置为橙色
34.             </p>
35.         </body>
36.     </html>
```

运行效果如图 3-8 所示。

图 3-8　CSS 颜色的应用效果

【代码说明】

本示例代码包含了三个段落元素<p>,为其规定了不同的类名称分别为 red、blue 和 orange,以便使用类选择器对相同类型的元素实现不一样的样式效果。

在首部标签<head>和</head>之间使用类选择器对其进行样式的规范:类名称为 red 则要求设置字体颜色为红色,使用了十六进制的颜色表达方式;类名称为 blue 则要求设置字体颜色为蓝色,使用了 RGB 颜色表达方式;类名称为 orange 则要求设置字体颜色为橙色,使用了英文名称的颜色表达方式。

由图 3-8 可见,使用不同的颜色表示方法均可以令 HTML 元素显示指定的 CSS 颜色。

3.5 CSS 常用样式

3.5.1 CSS 背景

本节将介绍如何在网页上应用背景颜色和背景图像。

和 CSS 背景有关的属性如表 3-9 所示。

表 3-9　CSS 背景与颜色属性

属性名称	解　释
background-color	设置背景颜色
background-image	设置背景图像

续表

属性名称	解　　释
background-repeat	设置背景图像是否重复平铺
background-attachment	背景图像是否随页面滚动
background-position	放置背景图像的位置
background	上述所有属性的综合简写方式

1. 背景颜色 background-color

CSS 中的 background-color 属性用于为所有 HTML 元素指定背景颜色。例如：

```
p{background-color:gray}          /*将段落元素的背景颜色设置为灰色*/
```

如需要更改整个网页的背景颜色，则对< body >元素应用 background-color 属性。例如：

```
body{background-color:cyan}       /*将整个网页的背景颜色设置为青色*/
```

background-color 属性的默认值是 transparent（透明的），因此如果没有特别规定 HTML 元素的背景颜色，那么该元素就是透明的，以便使其覆盖的元素为可见。关于颜色值的表达方法可参考 3.4.6 节的相关内容。

【例 3-9】 CSS 属性 background-color 的简单应用

使用 background-color 属性为各种 HTML 元素设置背景颜色。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 background-color 的应用</title>
        <style>
            body {
                background-color: silver   /*将整个网页的背景颜色设置为银色*/
            }
            h1 {
                background-color: red      /*设置背景色为红色*/
            }
            h2 {
                background-color: orange   /*设置背景色为橙色*/
            }
            h3 {
                background-color: yellow   /*设置背景色为黄色*/
            }
            h4 {
                background-color: green    /*设置背景色为绿色*/
            }
            h5 {
                background-color: blue     /*设置背景色为蓝色*/
            }
            h6 {
                background-color: purple   /*设置背景色为紫色*/
            }
            p {
                background-color: cyan     /*设置背景色为青色*/
```

```
            }
        </style>
    </head>
    <body>
        <h1>CSS 属性 background-color 的应用</h1>
        <h2>CSS 属性 background-color 的应用</h2>
        <h3>CSS 属性 background-color 的应用</h3>
        <h4>CSS 属性 background-color 的应用</h4>
        <h5>CSS 属性 background-color 的应用</h5>
        <h6>CSS 属性 background-color 的应用</h6>
        <p>
            CSS 属性 background-color 的应用
        </p>
    </body>
</html>
```

运行效果如图 3-9 所示。

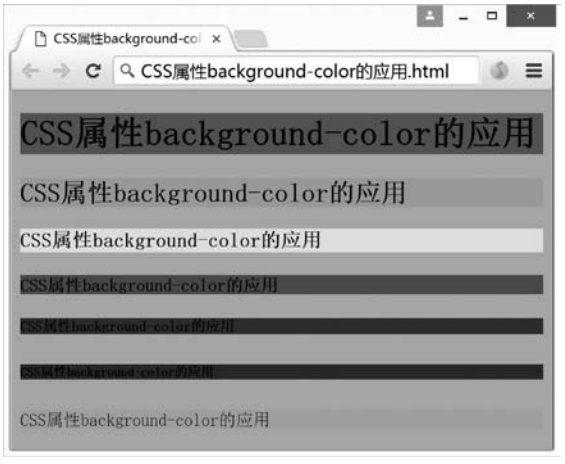


图 3-9 CSS 颜色的应用效果

【代码说明】

上述代码包含了全部的 6 种标题元素<h1>~<h6>以及一个段落元素<p>,使用内部样式表为这些元素以及<body>标签设置了不同的 background-color 属性值。

2. 背景图像 background-image

CSS 中的 background-image 属性用于为元素设置背景图像。例如:

```
p{background-image:url(flower.jpg)}
```

上述代码表示 flower.jpg 图片与 HTML 文档在同一个目录中。

如果引用本地其他文件夹中的图片,给出对应的文件夹路径即可。例如:

```
p{background-image:url(image/flower.jpg)}
```

上述代码表示 flower.jpg 图片在本地的 image 文件夹中,并且 image 文件夹与 HTML 文档存放于同一个目录中。

如果需要更改整个网页的背景图像,则对<body>元素应用 background-image 属性。例如:

```
body{ background-image:url(image/flower.jpg)}
```

【例 3-10】 CSS 属性 background-image 的简单应用

设置网页的背景图片和段落元素<p>的背景图片。


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 background-image 的应用</title>
        <style>
            body {
                background-image: url(image/sky.jpg)
            }
            p {
                background-image: url(image/balloon.jpg);
                width: 210px;
                height: 250px
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 background-image 的应用</h3>
        <hr />
        <!-- 段落元素用于显示热气球 -->
        <p>
            这是一个段落
        </p>
    </body>
</html>
```

运行效果如图 3-10 所示。

图 3-10 CSS 颜色的应用效果图

【代码说明】

上述代码为网页设置了背景图像,图像来源于本地 image 文件夹中的 sky.jpg 图片;并且为段落元素<p>设置了背景图像,图像来源于本地 image 文件夹中的 balloon.jpg 图片。

由图 3-10 可见,网页的背景图片会自动在水平和垂直两个方向上进行重复平铺的显示效果。实际上所有 HTML 元素的背景图片都会默认进行重复平铺。因此为达到更好的视觉效果,为段落元素<p>设置了与背景图片 balloon.jpg 一样的宽和高,即宽度 210 像素、高度 250 像素。

CSS 还可以规定是否允许背景图像重复平铺,相关属性会在下一节中进行描述。

3. 背景图像平铺方式 background-repeat

CSS 中的 background-repeat 属性用于设置背景图像的平铺方式。如果不设置该属性,则默认背景图像会在水平和垂直方向上同时被重复平铺(如例 3-10 的运行效果)。该属性有四种不同的取值,如表 3-10 所示。

表 3-10 CSS 属性 background-repeat 取值

属 性 值	解 释	属 性 值	解 释
repeat-x	水平方向平铺	repeat	水平和垂直方向都平铺
repeat-y	垂直方向平铺	no-repeat	不平铺,只显示原图

【例 3-11】 CSS 属性 background-repeat 的简单应用

沿用例 3-10 中的背景图像 sky.jpg 设置为网页背景图片,并要求不平铺背景图片。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 background-image 的应用</title>
        <style>
            body {
                background-image: url(image/sky.jpg);
                background-repeat: no-repeat;            /* 背景图像不平铺 */
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 background-image 的应用</h3>
    </body>
</html>
```

运行效果如图 3-11 所示。

【代码说明】

和例 3-10 不同的是,本示例代码修改了内部 CSS 样式表中的 body 样式,为其新增了 background-repeat 属性,并将属性值设置为 no-repeat 表示不平铺图像。

但是这种方式设置的背景图像是固定的宽高,在浏览器窗口尺寸大于图像尺寸时会有多余出来的空白区域。因此可以考虑将 CSS 属性 background-image 与 background-color 配合使用,这样背景图片无法覆盖到的空白区域将显示与图片相近的背景颜色。

4. 固定/滚动背景图像 background-attachment

CSS 中的 background-attachment 属性用于设置背景图像是固定在屏幕上还是随着页

图 3-11 CSS 属性 background-repeat 的应用效果

面滚动。该属性有两种取值，如表 3-11 所示。

表 3-11 CSS 属性 background-attachment 取值

属 性 值	解 释
scroll	背景图像随着页面滚动
fixed	背景图像固定在屏幕上

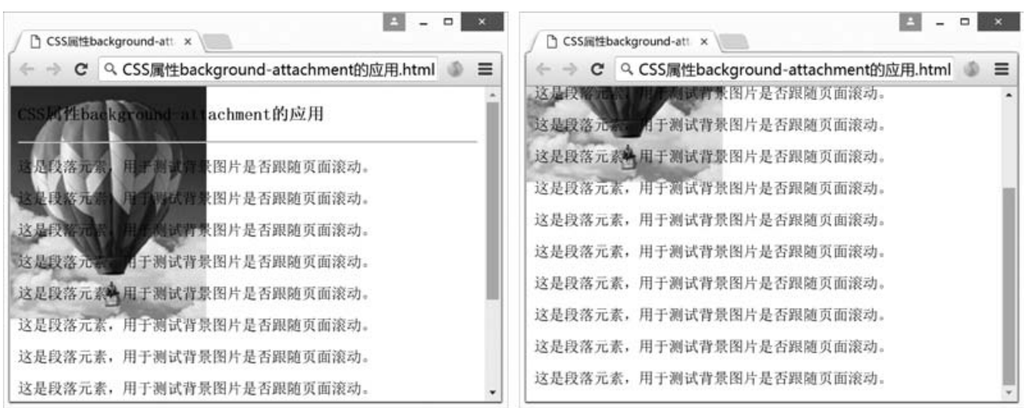
【例 3-12】 CSS 属性 background-attachment 的简单应用

使用本地 image 文件夹中的 balloon.jpg 作为网页背景图片，并设置为不重复平铺图像。


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 background-attachment 的应用</title>
        <style>
            body {
                background-image: url(image/balloon.jpg);
                background-repeat: no-repeat;      /* 背景图像不平铺 */
                background-attachment: scroll;     /* 背景图像随页面滚动 */
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 background-attachment 的应用</h3>
        <hr />
        <p>
            这是段落元素，用于测试背景图片是否跟随页面滚动。
        </p>
        <p>
            这是段落元素，用于测试背景图片是否跟随页面滚动。
```

```
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
            <p>
                这是段落元素,用于测试背景图片是否跟随页面滚动。
            </p>
        </body>
    </html>
```

运行效果如图 3-12 所示。



(a) 页面滚动前　　(b) 页面滚动后

图 3-12　CSS 属性 background-attachment 的应用效果

【代码说明】

本示例在页面上设置足够多的段落元素<p>以便让浏览器形成滚动条,将背景图片的 background-attachment 属性设置为 scroll,测试其运行效果。

由图 3-12 可见，当 background-attachment 的属性值为 scroll 时，背景图像会随着页面一起滚动。可以将该属性值改为 fixed 重新进行测试，在页面滚动时背景图片不随着文字内容一起移动。

5. 定位背景图像 background-position

默认情况下，背景图像会放置在元素的左上角。CSS 中的 background-position 属性用于设置背景图像的位置，可以根据属性值的组合将图像放置到指定位置上。该属性允许使用两个属性值组合的形式对背景图像进行定位。其基本格式如下：

```
background-position: 水平方向值  垂直方向值
```

水平和垂直方向的属性值均可使用关键词、长度值或者百分比的形式表示。

1）关键词定位

在 background-position 属性值中可以使用的关键词共有 5 种，如表 3-12 所示。

表 3-12 CSS 属性 background-position 关键词

属 性 值	解 释	属 性 值	解 释
center	水平居中或垂直居中	left	水平方向左对齐显示
top	垂直方向置顶显示	right	水平方向右对齐显示
bottom	垂直方向底部显示		

使用关键词组合的方式定位图像，需要从表示水平方向和垂直方向的关键词中各选一个组合使用，例如，background-position:left top 表示背景图像在元素左上角的位置。

关键词指示的方向非常明显，例如 left 和 right 就是水平方向专用，而 top 和 bottom 是垂直方向专用。因此关键词的组合可以不分先后顺序，例如 left top 和 top left 就表达完全相同的含义。关键词 center 既可表示水平居中也可表示垂直居中，组合使用时取决于另一个关键词是水平还是垂直方向，center 则用于补充对立方向。

关键词定位的方式也可以简写为单个关键词的形式，这种情况会默认另一个省略的关键词为 center。例如，简写形式 left 就等价于 left center 或 center left，表示水平方向左对齐、垂直方向居中显示。

2）长度值定位

长度值定位方法是以元素内边距区域左上角的点作为原点，然后解释背景图像左上角的点对原点的偏移量。例如，background-position: 100px 50px 指的是背景图像左上角的点距离元素左上角向右 100 像素同时向下 50 像素的位置。

3）百分比定位

百分比数值定位方式更为复杂，是将 HTML 元素与其背景图像在指定的点上重合对齐，而指定的点是用百分比的方式进行解释的。

例如，background-position: 0% 0%指的是背景图像左上角的点放置在 HTML 元素左上角原点上。而 background-position:66% 33%指的是 HTML 元素和背景图像水平方向 2/3 的位置和垂直方向 1/3 的位置上的点对齐。

一般来说，使用百分比定位方式都是用两个参数值组合定位的，第一个参数值表示水平方向的位置；第二个参数值表示垂直方向的位置。如果简写为一个参数值，则只表示水平方向的位置，省略的垂直方向位置默认为 50%。这种方法类似于关键词定位法简写时使用 center 补全省略的关键词。

【例 3-13】 CSS 属性 background-position 的综合应用

综合应用了关键词、百分比和长度值三种方式进行背景图像的定位。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 background-position 的应用</title>
        <style>
            div {
                width: 660px;
            }
            p {
                width: 200px;
                height: 200px;
                background-color: silver;
                background-image: url(image/football.png);
                background-repeat: no-repeat;
                float: left;
                margin: 10px;
                text-align: center;
            }
            #p1_1 {
                background-position: left top
            }/* 图像位于左上角,也可以写作 top left */
            #p1_2 {
                background-position: top
            }/* 图像位于顶端居中,也可以写作 top center 或 center top*/
            #p1_3 {
                background-position: right top
            }/* 图像位于右上角,也可以写作 top right */

            #p2_1 {
                background-position: 0%
            }/* 图像位于水平方向左对齐并且垂直居中,也可以写作 0% 50% */
            #p2_2 {
                background-position: 50%
            }/* 图像位于正中心,也可以写作 50% 50% */
            #p2_3 {
                background-position: 100%
            }/* 图像位于水平方向右对齐并且垂直居中,也可以写作 100% 50% */

            #p3_1 {
                background-position: 0px 100px
            }/* 图像位于左下角 */
            #p3_2 {
                background-position: 50px 100px
            }/* 图像位于底端并水平居中*/
            #p3_3 {
                background-position: 100px 100px
            }/* 图像位于右下角 */
        </style>
    </head>
    <body>
        <h3>CSS 属性 background-position 的应用</h3>
        <hr />
```

```
54.          <div>
55.              <p id="p1_1">left top</p>
56.              <p id="p1_2">top</p>
57.              <p id="p1_3">right top</p>
58.
59.              <p id="p2_1">0%</p>
60.              <p id="p2_2">50%</p>
61.              <p id="p2_3">100%</p>
62.
63.              <p id="p3_1">0px 100px</p>
64.              <p id="p3_2">50px 100px</p>
65.              <p id="p3_3">100px 100px</p>
66.          </div>
67.      </body>
68.  </html>
```

运行效果如图 3-13 所示。

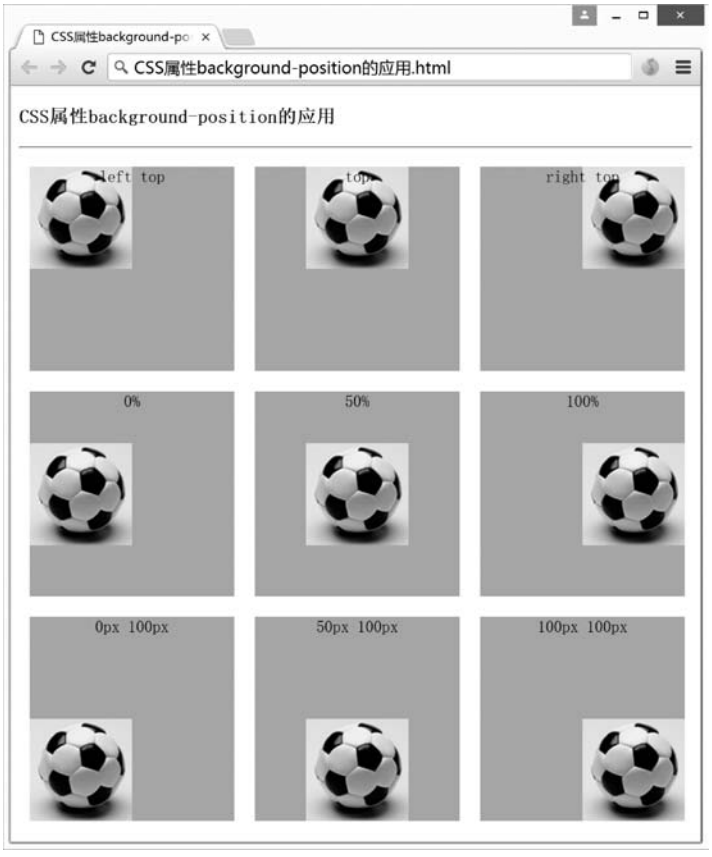


图 3-13 CSS 属性 background-position 的应用效果

【代码说明】

本示例使用区域元素 包含了九个段落元素<p>,用于测试背景图像的定位。将这些段落元素分为三组,每行三个为一组使用其中一种背景图像定位方式:第一行使用关键词定位;第二行为百分比定位;第三行是长度值定位。

事先为段落元素<p>设置统一样式:宽和高均为 200 像素、背景色为银色、文字内容水平居中显示、背景图像来源于本地 image 文件夹中的 football.png 并且不重复显示。然后

使用 ID 选择器分别为每一个段落元素设置不同的背景图像位置。

几种特殊的百分比数值可以与关键词定位法等价使用,具体内容如表 3-13 所示。

表 3-13 百分比与等价关键词定位一览表

百分比定位	等价关键词定位	背景图像位置
0% 0%	left top 或 top left	元素的左上角
0% 50%或 0%	left center 或 left	水平方向左对齐,垂直方向居中
0% 100%	left bottom 或 bottom left	元素的左下角
50% 0%	center top 或 top	水平方向居中,垂直方向置顶
50% 50%或 50%	center center 或 center	元素的正中心
50% 100%	center bottom 或 bottom	水平方向居中,垂直方向底端
100% 0%	right top 或 top right	元素的右上角
100% 50%或 100%	right center 或 right	水平方向右对齐,垂直方向居中
100% 100%	right bottom 或 bottom right	元素的右下角

如果已知 HTML 元素的宽度和高度,可以将表 3-13 换算成长度值。由于实际情况下每个 HTML 元素的尺寸不一样,因此这里不再专门列出。

6. 背景简写 background

CSS 中的 background 属性可以用于概括其他五种背景属性,将相关属性值汇总写在同一行。当需要为同一个元素声明多项背景属性时,可以使用 background 属性进行简写。声明顺序如下:

```
[background-color] [background-image] [background-repeat] [background-attachment]
[background-position]
```

属性值之间用空格隔开,如果其中某个属性没有规定可以省略不写。例如:

```
p{
    background-color:silver;
    background-image:url(image/football.png);
    background-repeat:no-repeat;
}
```

上述代码使用 background 属性可以简写为:

```
p{ background: silver url(image/football.png) no-repeat }
```

其效果完全相同。

3.5.2 CSS 框模型

CSS 框模型又称为盒状模型(Box Model),用于描述 HTML 元素形成的矩形盒子。每个 HTML 元素都具有元素内容、内边距、边框和外边距。CSS 框模型的结构如图 3-14 所示。

图 3-14 中最内层的虚线框里面是元素的实际内容;包围它的一圈称为内边距,内边距的最外层实线边缘称为元素的边框;

图 3-14 CSS 框模型的结构

边框外层的一圈空白边称为外边距，外边距是该元素与其他元素之间保持的距离。其中最外层的虚线部分是元素外边距的临界线。默认情况下，元素的内边距、边框和外边距均为 0。

在 CSS 中元素的宽度(width)和高度(height)属性指的是元素内容的区域，也就是图 3-14 中的最内层虚线框的宽度和高度。增加内外边距或边框的宽度不会影响元素的宽度和高度属性值，但是元素占用的总空间会增大。

1. 内边距 padding

1) 设置各边内边距

在 CSS 中，可以使用 padding 属性设置 HTML 元素的内边距。元素的内边距也可以被理解为元素内容周围的填充物，因为内边距不影响当前元素与其他元素之间的距离，它只能用于增加元素内容与元素边框之间的距离。

padding 属性值可以是长度值或者百分比值，但是不可以使用负数。

例如，为所有的段落元素<p>设置各边均为 20 像素的内边距：

```
p{padding:20px}
```

使用百分比值表示的是该元素的上一级父元素宽度(width)的百分比。例如：

```
<div style = "width:100px">
    <p style = "padding:20 % ">这是一个段落</p>
</div>
```

此时使用了内联样式表为段落元素<p>设置内边距为父元素宽度的 20%。该段落元素<p>的父元素为块级元素 ，因此段落元素<p>各边的内边距均为是 元素宽度的 20%，即 20 像素。

padding 属性也可以为元素的各边分别设置内边距。例如：

```
p{padding: 10px 20px 0 20 % }
```

此时规定的属性值按照上右下左的顺时针顺序为各边的内边距进行样式定义。因此本例表示上边内边距为 10 像素；右边内边距为 20 像素；下边内边距为 0；左边内边距为其父元素宽度的 20%。

2) 单边内边距

如果只需要为 HTML 元素的某一个边设置内边距，可以使用 padding 属性的 4 种单边内边距属性，如表 3-14 所示。

表 3-14 CSS 单边内边距属性

属性名称	解释	属性名称	解释
padding-top	设置元素的上边内边距	padding-left	设置元素的左边内边距
padding-bottom	设置元素的下边内边距	padding-right	设置元素的右边内边距

例如，设置段落元素<p>的上边内边距为 20 像素：

```
p{padding - top: 20px}
```

【例 3-14】 CSS 属性 padding 的应用

测试段落元素<p>使用内边距属性 padding 的不同效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 padding 的应用</title>
        <style>
            p {
                width: 200px;
                margin: 10px;
                background-color: orange;
            }
            .style01 {
                padding: 20px
            }
            .style02 {
                padding: 10px 50px
            }
            .style03 {
                padding-left: 50px
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 padding 的应用</h3>
        <hr />
        <p>
            该段落没有使用内边距,默认值为 0
        </p>
        <p class="style01">
            该段落元素的各边内边距均为 20 像素
        </p>
        <p class="style02">
            该段落元素的上下边内边距均为 10 像素、左右边内边距均为 50 像素
        </p>
        <p class="style03">
            该段落元素的左边内边距为 50 像素
        </p>
    </body>
</html>
```

运行效果如图 3-15 所示。

图 3-15 CSS 属性 padding 的应用效果

【代码说明】

本示例使用了四个段落元素<p>进行对比试验,其中第一个段落元素没有做 CSS 样式设置,作为原始参考。其余三个段落元素分别进行了三种不同情况的内边距设置:

- 使用 padding 属性加单个属性值的形式为各边同时设置相同的内边距。
- 使用 padding 属性加两个属性值的形式分别为上下边和左右边设置不同的内边距。
- 使用 padding-left 属性设置单边的内边距。

事先为段落元素<p>设置统一样式:宽度为

200 像素、背景色为橙色以及外边距为 10 像素。然后使用类选择器为段落元素设置不同的 padding 属性值。

注意：如果为元素填充背景颜色或背景图像，则其显示范围是边框以内的区域，包括元素实际内容和内边距。

2. 边框 border

使用 CSS 边框的相关属性可以为 HTML 元素创建不同宽度、样式和颜色的边框。和 CSS 边框有关的属性如表 3-15 所示。

1）边框宽度 border-width

CSS 中的 border-width 属性用于定义 HTML 元素边框的宽度。该属性有四种取值，如表 3-16 所示。

表 3-15 CSS 边框属性一览表

属性名称	解释
border-width	设置边框的宽度
border-style	设置边框的样式
border-color	设置边框的颜色
border	上述所有属性的综合简写方式

表 3-16 CSS 属性 border-width 取值

属性值	解释
thin	较窄的边框
medium	中等宽度的边框
thick	较宽的边框
像素值	自定义像素值宽度的边框

注：该属性必须和边框样式 border-style 属性一起使用方可看出效果。

【例 3-15】 CSS 属性 border-width 的简单应用

实验 CSS 属性 border-width 不同取值的显示效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 border-width 的简单应用</title>
        <style>
            p {
                width: 200px;
                height: 50px;
                border-style: solid;
            }
            .thin {
                border-width: thin
            }
            .medium {
                border-width: medium
            }
            .thick {
                border-width: thick
            }
            .one {
                border-width: 1px
            }
            .ten {
                border-width: 10px
            }
        </style>
    </head>
```

```
29.      <body>
30.          <h3>CSS 属性 border - width 的简单应用</h3>
31.          <hr />
32.          <p class = "one">
33.              边框宽度为 1 像素
34.          </p>
35.          <p class = "thin">
36.              边框宽度为 thin
37.          </p>
38.          <p class = "medium">
39.              边框宽度为 medium
40.          </p>
41.          <p class = "thick">
42.              边框宽度为 thick
43.          </p>
44.          <p class = "ten">
45.              边框宽度为 10 像素
46.          </p>
47.      </body>
48.  </html>
```

运行效果如图 3-16 所示。

图 3-16　CSS 属性 border-width 的应用效果

【代码说明】

本例中包含了五个段落元素<p>,并使用 CSS 类选择器为其设置不同的 border-width 属性值。为达到更好的显示效果,预先为所有段落元素设置了统一 CSS 样式:宽 200 像素、高 50 像素,并且设置边框为实线。

2) 边框样式 border-style

CSS 中的 border-style 属性用于定义 HTML 元素边框的样式。该属性有 10 种取值,如表 3-17 所示。

表 3-17 CSS 属性 border-style 取值一览表

属 性 值	解 释	属 性 值	解 释
none	定义无边框效果	groove	定义 3D 凹槽边框效果
dotted	定义点状边框效果	ridge	定义 3D 脊状边框效果
dashed	定义虚线边框效果	inset	定义 3D 内嵌边框效果
solid	定义实线边框效果	outset	定义 3D 外凸边框效果
double	定义双线边框效果	inherit	从父元素继承边框样式

【例 3-16】 CSS 属性 border-style 的简单应用

实验 CSS 属性 border-style 不同取值的显示效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 border-style 的应用</title>
        <style>
            p {
                width: 200px;
                height: 30px;
                border-width: 5px;
            }
            #p01 {
                border-style: none
            }
            #p02 {
                border-style: dotted
            }
            #p03 {
                border-style: dashed
            }
            #p04 {
                border-style: solid
            }
            #p05 {
                border-style: double
            }
            #p06 {
                border-style: groove
            }
            #p07 {
                border-style: ridge
            }
            #p08 {
                border-style: inset
            }
            #p09 {
                border-style: outset
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 border-style 的应用</h3>
        <hr />
```

```
        <p id="p01">无边框效果</p>
        <p id="p02">点状边框效果</p>
        <p id="p03">虚线边框效果</p>
        <p id="p04">实线边框效果</p>
        <p id="p05">双线边框效果</p>
        <p id="p06">3D 凹槽边框效果</p>
        <p id="p07">3D 脊状边框效果</p>
        <p id="p08">3D 内嵌边框效果</p>
        <p id="p09">3D 外凸边框效果</p>
    </body>
</html>
```

运行效果如图 3-17 所示。

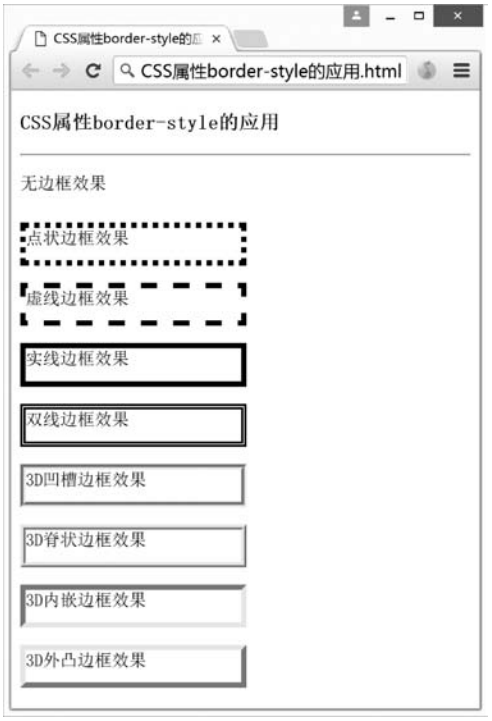


图 3-17　CSS 属性 border-style 的应用效果

【代码说明】

本例中包含了九个段落元素<p>,并使用 ID 选择器为其设置不同的 border-style 属性值。为达到更好的显示效果,预先为所有段落元素设置了统一的 CSS 样式:宽 200 像素、高 30 像素,并且设置边框宽度为 5 像素。由图 3-17 可见,根据 border-style 不同的取值,可以获得不同的显示效果。

border-style 属性也可以单独为元素的各边设置边框样式。例如:

```
p{border-style: solid dashed dotted double}
```

此时规定的属性值按照上右下左的顺时针顺序为各边的边框进行样式定义。因此本例表示上边框为实线;右边框为虚线;下边框为点状线;左边框为双线。

如果各边的边框有部分重复的样式值,可以使用简写的方式。如果简写为三个属性值的样式,则左右边框共用中间的属性值。例如:

```
p{border-style: solid dashed double}
```

本例表示上边框为实线;左右边框为虚线;下边框为双线。

如果简写为两个属性值的样式,则上下边框共用第一个属性值,左右边框共用第二个属性值。例如:

```
p{border-style: solid dashed }
```

本例表示上下边框均为实线,左右边框均为虚线。

3) 边框颜色 border-color

CSS 中的 border-color 属性用于定义 HTML 元素边框的颜色。其属性值为正常的颜色值即可,例如 red 表示红色边框等。关于颜色的写法可以参考 3.4.6 节,此处不再赘述。

【例 3-17】 CSS 属性 border-color 的简单应用

实验 CSS 属性 border-color 不同取值的显示效果。


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 border-color 的应用</title>
        <style>
            p {
                width: 200px;
                height: 30px;
                border-width: 10px;
                border-style: solid;
            }
            #p01 {
                border-color: red
            }
            #p02 {
                border-color: rgb(0,255,0)
            }
            #p03 {
                border-color: #00F
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 border-color 的应用</h3>
        <hr />
        <p id="p01">
            红色边框效果
        </p>
        <p id="p02">
            绿色边框效果
        </p>
        <p id="p03">
            蓝色边框效果
        </p>
    </body>
</html>
```

运行效果如图 3-18 所示。

图 3-18 CSS 属性 border-color 的应用效果

【代码说明】

本例中包含了三个段落元素<p>，并使用 ID 选择器为其设置不同的 border-color 属性值。为达到更好的显示效果，预先为所有段落元素设置了统一 CSS 样式：宽 200 像素、高 30 像素，并且设置边框宽度为 10 像素，边框风格是实线边框。

为了演示颜色的不同表达方式，这三种边框分别使用了关键词、RGB 和十六进制码的方式表示红色 red、绿色 rgb(0,255,0)和蓝

色#00F。其中蓝色的十六进制码是简写的形式,完整写法为#0000FF。

4) 边框简写 border

CSS 中的 border 属性可以用于概括其他三种边框属性,将相关属性值汇总写在同一行。当需要为同一个元素声明多项边框属性时可以使用 border 属性进行简写。属性值无规定顺序,彼此之间用空格隔开,如果其中某个属性没有规定可以省略不写。例如:

```
p{
    border-width: 1px;
    border-style: solid;
    border-color: red
}
```

上述代码使用 border 属性可以简写为:

```
p{ border: 1px solid red}
```

其效果完全相同。

3. 外边距 margin

1) 设置各边外边距

在 CSS 中,可以使用 margin 属性设置 HTML 元素的外边距。元素的外边距也可以被理解为元素内容周围的填充物,因为内边距不影响当前元素与其他元素之间的距离,它只能用于增加元素内容与元素边框之间的距离。

margin 属性值可以是长度值或百分比,包括可以使用负数。例如,为所有的标题元素<h1>设置各边均为 10 像素的外边距:

```
h1{margin:10px}
```

和内边距 padding 属性类似,使用百分比值表示的也是当前元素上级父元素的宽度(width)百分比。例如:

```
<div style="width:300px">
    <p style="margin:10%">这是一个段落</p>
</div>
```

此时使用了内联样式表为段落元素<p>设置外边距为父元素宽度的 10%。该段落元素<p>的父元素为块级元素 ,因此段落元素<p>各边的外边距均为是 元素宽度的 10%,即 30 像素。

margin 属性同样也可以为元素的各边分别设置外边距。例如:

```
p{margin: 0 10% 20px 30px}
```

此时规定的属性值按照上右下左的顺时针顺序为各边的外边距进行样式定义。因此本例表示上边外边距为 0 像素;右边外边距为其父元素宽度的 10%;下边外边距为 20 像素;左边外边距为 30 像素。

如果在设置外边距时各边有部分重复值,可以写成简写的方式。

简写为三个属性值的样式,则左右边外边距共用中间的属性值。例如:

```
p{margin: 10px 0 30px}
```

本例表示上边外边距为 10 像素;左右边外边距为 0;下边外边距为 30 像素。

简写为两个属性值的样式，则上下边外边距共用第一个属性值、左右边外边距共用第二个属性值。例如：

```
p{margin: 20px 30px}
```

本例表示上下边外边距为 20 像素；左右边外边距为 30 像素。

2）单边外边距

如果只需要为 HTML 元素的某一个边设置外边距，可以使用 margin 属性的 4 种单边外边距属性，如表 3-18 所示。

表 3-18 CSS 单边外边距属性一览表

属性名称	解释	属性名称	解释
margin-top	设置元素的上边外边距	margin-left	设置元素的左边外边距
margin-bottom	设置元素的下边外边距	margin-right	设置元素的右边外边距

例如，设置段落元素<p>的左边外边距为 10 像素：

```
p{margin-left: 10px}
```

注意：不同的浏览器对于 HTML 元素的边距设置虽然基本都是默认为 8 像素，但是有细微的差异。其中 IE 和 Netscape 浏览器对<body>标签定义了默认外边距 margin 属性为 8px；而 Opera 浏览器相反是把内边距 padding 的默认值定义成了 8px。为了保证网页的 HTML 元素兼容各种浏览器，建议自定义<body>标签中的 margin 和 padding 属性值。

【例 3-18】 CSS 属性 margin 的应用

测试 元素使用外边距属性 margin 的不同效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>CSS 属性 margin 的应用</title>
        <style>
            .box {
                border: 1px solid;
                width: 300px;
                margin: 10px;
            }
            .yellow {
                background-color: yellow
            }
            .style01 {
                margin: 20px
            }
            .style02 {
                margin: 10px 50px
            }
            .style03 {
                margin-left: 100px
            }
        </style>
    </head>
```

```
    <body>
        <h3>CSS 属性 margin 的应用</h3>
        <hr />
        <div class = "box">
            <div class = "yellow">
                该段落没有使用外边距,默认值为 0
            </div>
        </div>
        <div class = "box">
            <div class = "style01 yellow">
                该段落元素的各边外边距均为 20 像素
            </div>
        </div>
        <div class = "box">
            <div class = "style02 yellow">
                该段落元素的上下边外边距均为 10 像素、左右边外边距均为 50 像素
            </div>
        </div>
        <div class = "box">
            <div class = "style03 yellow">
                该段落元素左外边距为 100 像素
            </div>
        </div>
    </body>
</html>
```

运行效果如图 3-19 所示。

图 3-19　CSS 属性 margin 的应用效果

【代码说明】

本示例使用了四组区域元素 进行对比试验,每组均为一个带有实线外框的 内部嵌套一个具有背景颜色和文字内容的 元素进行位置对照。

事先为作为外框的父元素 定义 class="box",并设置统一样式:宽度为 300 像素、边框为宽 1 像素的实线,并设置了 10 像素的外边距;然后为子元素 定义 class="yellow"并统一设置背景颜色为黄色。

其中第一组中的子元素 没有做外边距的样式设置,作为原始参考。其余三个段落元素分别进行了三种不同情况的外边距设置:

- 使用 margin 属性加单个属性值的形式为各边同时设置相同的外边距。
- 使用 margin 属性加两个属性值的形式分别为上下边和左右边设置不同的外边距。
- 使用 margin-left 属性设置单边的外边距。

3) 外边距合并

外边距合并又称为外边距叠加,指的是如果两个元素的垂直外边距相连接会发生重叠合并,其高度是合并前这两个外边距中的较大值。

因此外边距合并主要指的就是上下外边距的合并,存在以下三种可能:

- 当元素 B 出现在元素 A 下面时,元素 A 的下边距会与元素 B 的上边距发生重叠

合并。

- 当元素 B 包含在元素 A 内部时，如果元素 B 的上/下内边距均为 0，也会发生上/下外边距合并现象。
- 当空元素没有边框和内边距时，上下外边距也会发生合并。

注意：只有普通块级元素的垂直外边距才会发生合并，如果是特殊情况，例如浮动框、行内框或者绝对定位之间的外边距是不会发生合并的。

3.5.3 CSS 文本

本节将介绍如何对网页上的文本内容进行修饰。和 CSS 文本有关的属性如表 3-19 所示。

表 3-19 CSS 文本属性一览表

属性名称	解释
text-indent	设置文本缩进
text-align	设置文本对齐方式(左对齐、居中、右对齐)
text-decoration	设置文本装饰(下画线、删除线、上画线)
text-transform	设置文本大小写的转换
letter-spacing	设置字符间距

1. 文本缩进 text-indent

CSS 中的 text-indent 属性用于为段落文本设置首行缩进效果。例如，为段落元素<p>设置 20 像素的首行缩进：

```
p{text-indent: 20px}
```

【例 3-19】 CSS 属性 text-indent 的简单应用


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 text-indent 的应用</title>
        <style>
            p {
                text-indent: 2em;
                border: 1px solid;
                width: 200px;
                padding: 10px;
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 text-indent 的应用</h3>
        <hr />
        <p>
        这是一个用于测试首行缩进效果的段落元素.当前缩进了两个字符的距离。
        </p>
    </body>
</html>
```

运行效果如图 3-20 所示。

图 3-20 CSS 属性 text-indent 的应用效果

【代码说明】

本示例包含了一个简单的段落元素<p>，并设置其样式为：带有 1 像素的实线边框，宽 200 像素，各边内边距为 10 像素，并使用 text-indent 属性为其设置了 2em 的首行缩进。em 是一个相对长度单位，表示的是原始字体大小的倍数，因此当前的 2em 表示的正好是两个字符的距离。

2. 文本对齐 text-align

CSS 中的 text-align 属性用于为文本设置对齐效果。该属性有四种取值，如表 3-20 所示。

表 3-20 CSS 属性 text-align 取值

属 性 值	解 释	属 性 值	解 释
left	文本内容左对齐	center	文本内容居中显示
right	文本内容右对齐	justify	文本内容两端对齐

其中 justify 的取值在多数浏览器上显示会存在问题，因为 CSS 本身并没有规定如何将文字向两端拉伸，拉伸的依据是各类浏览器本身的规则。并且目前 CSS 尚未规定连字符的处理方式。因此为了各类浏览器的兼容效果，应慎用该属性值。

【例 3-20】 CSS 属性 text-align 的简单应用


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 text-align 的应用</title>
        <style>
            div {
                border: 1px solid;
                width: 300px;
                padding: 10px;
            }
            .center {text-align: center}/*文本居中对齐*/
            .left {text-align: left}/*文本左对齐*/
            .right {text-align: right}/*文本右对齐*/
        </style>
    </head>
    <body>
        <h3>CSS 属性 text-align 的应用</h3>
        <hr />
        <div>
            <p class="center">
                文字居中对齐
            </p>
            <p class="left">
                文字左对齐
            </p>
```

```
            <p class = "right">
                文字右对齐
            </p>
        </div>
    </body>
</html>
```

运行效果如图 3-21 所示。

图 3-21 CSS 属性 text-align 的应用效果

【代码说明】

本示例包含了三个段落元素< p >,分别用于测试文字居中对齐、左对齐和右对齐的显示效果。在三个段落元素的外面使用了一个< div >元素进行总体嵌套处理,该< div >元素设置为:带有 1 像素实线的边框,宽度为 500 像素并且内边距为 10 像素。

以类选择器的方式定义了 CSS 属性 text-align 的三种效果,类名称定义为和属性值一样的文字内容,并将其应用于三个不同的段落元素中,获得最终效果。

3. 文本装饰 text-decoration

CSS 中的 text-decoration 属性用于为文本添加装饰效果,例如下画线、删除线和上画线等。该属性有四种取值,如表 3-21 所示。

表 3-21 CSS 属性 text-decoration 取值

属 性 值	解 释	属 性 值	解 释
underline	为文本添加下画线	overline	为文本添加上画线
line-through	为文本添加删除线	none	正常状态的文本

【例 3-21】 CSS 属性 text-decoration 的简单应用


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>CSS 属性 text - decoration 的应用</title>
        <style>
            .underline {text - decoration: underline}          /* 下画线 */
            .line - through {text - decoration: line - through}  /* 删除线 */
            .overline {text - decoration: overline}            /* 上画线 */
        </style>
    </head>
    <body>
        <h3>CSS 属性 text - decoration 的应用</h3>
        <hr />
        <p class = "underline">
            为文字添加下画线
        </p>
        <p class = "line - through">
            为文字添加删除线
        </p>
```

```
        <p class = "overline">
            为文字添加上画线
        </p>
    </body>
</html>
```

运行效果如图 3-22 所示。

图 3-22 CSS 属性 text-transform 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 text-decoration 不同属性值的效果。以类选择器的方式定义了 CSS 属性 text-decoration 的三种效果(上画线、删除线和下画线),类名称定义为和属性值一样的文字内容,并将其应用于三个不同的段落元素中。

一般来说,文本默认情况下就是 text-decoration 属性值为 none 的状态,无须特别声明,因此未在本示例中展示。但是 none 属性值适用于去掉超链接文本内容的下画线,具体用法请参考 3.5.5 节。

4. 文本转换 text-transform

CSS 中的 text-transform 属性用于设置文本的大小写。该属性有四种取值,如表 3-22 所示。

表 3-22 CSS 属性 text-transform 取值一览表

属 性 值	解 释	属 性 值	解 释
uppercase	将文本中每个字母都转换为大写	capitalize	将文本中的首字母转换为大写
lowercase	将文本中每个字母都转换为小写	none	将文本保持原状不做任何转换

【例 3-22】 CSS 属性 text-transform 的简单应用


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset = "utf-8">
        <title>CSS 属性 text - transform 的应用</title>
        <style>
            .uppercase {text - transform: uppercase}      /* 全大写 */
            .lowercase {text - transform: lowercase}      /* 全小写 */
            .capitalize {text - transform: capitalize}    /* 单词首字母大写 */
        </style>
    </head>
    <body>
        <h3>CSS 属性 text - transform 的应用</h3>
        <hr />
        <p class = "uppercase">
            hello javaScript
        </p>
        <p class = "lowercase">
            HELLO JAVASCRIPT
        </p>
        <p class = "capitalize">
```

```
            hello javaScript
        </p>
    </body>
</html>
```

运行效果如图 3-23 所示。

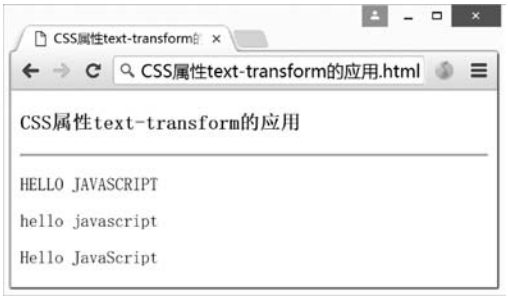


图 3-23 CSS 属性 text-transform 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 text-transform 不同属性值的效果。以类选择器的方式定义了 CSS 属性 text-transform 的三种效果(全大写、全小写和首字母大写),类名称定义为和属性值一样的文字内容,并将其应用于三个不同的段落元素中。

一般来说,文本默认情况下就是 text-transform 属性值为 none 的状态,无须特别声明,因此未在本示例中展示。

5. 字符间距 letter-spacing

CSS 中的 letter-spacing 属性用于设置文本中字符的间距,其属性值为长度值。例如,将标题元素<h1>设置成字间距为 10 像素的宽度:

```
h1{letter-spacing:10px}
```

【例 3-23】 CSS 属性 letter-spacing 的简单应用


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 letter-spacing 的应用</title>
        <style>
            .style01 {
                letter-spacing: 1em
            }
            .style02 {
                letter-spacing: 2em
            }
            .style03 {
                letter-spacing: -5px
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 letter-spacing 的应用</h3>
        <hr />
        <p class="style01">
            文字字间距为 1em
        </p>
        <p class="style02">
            文字字间距为 2em
        </p>
        <p class="style03">
```

```
28.                 文字字间距为 - 5px
29.             </p>
30.         </body>
31.     </html>
```

运行效果如图 3-24 所示。

图 3-24 CSS 属性 letter-spacing 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 letter-spacing 不同属性值的效果。以类选择器的方式定义了 CSS 属性 letter-spacing 的三种效果(字间距 1em、2em 和-5px 的情况),类名称定义为 style01、style02 和 style03,并将其应用于三个不同的段落元素中。

由图 3-24 可见,letter-spacing 的属性值允许是负数,但是需要适度,否则字符会全部堆积在一起无法识别。

3.5.4 CSS 字体

本节将介绍如何对字体进行样式设置。和 CSS 字体有关的属性如表 3-23 所示。

表 3-23 CSS 字体属性一览表

属性名称	解释
font-family	设置字体系列
font-style	设置字体风格(正常、斜体、倾斜三种)
font-variant	设置字体变化(小型尺寸的大写字母等)
font-weight	设置字体的粗细
font-size	设置字体尺寸
font	上述所有属性的综合简写方式

1. 字体系列 font-family

在 CSS 中,将字体分为两类:一类是特定字体系列(family-name);另一类是通用字体系列(generic family)。特定字体系列指的是拥有具体名称的某一种字体,比如宋体、楷体、黑体、Times New Roman、Arial 等;而通用字体系列指的是具有相同外观特征的字体系列。

除了常见的各种特定字体外,CSS 规定了五种通用字体系列:

- Serif 字体;
- Sans-serif 字体;
- Monospace 字体;
- Cursive 字体;
- Fantasy 字体。

【例 3-24】 CSS 属性 font-family 的简单应用


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>CSS 属性 font - family 的应用</title>
6.          <style>
```

```
7.                .style01 {
8.                    font-family: "AR DELANEY"
9.                }
10.               .style02 {
11.                   font-family: "French Script MT"
12.               }
13.               .style03 {
14.                   font-family: "微软雅黑 Light"
15.               }
16.           </style>
17.       </head>
18.       <body>
19.           <h3>CSS 属性 font-family 的应用</h3>
20.           <hr />
21.           <p class="style01">
22.               AR DELANEY
23.           </p>
24.           <p class="style02">
25.               French Script MT
26.           </p>
27.           <p class="style03">
28.               微软雅黑 Light
29.           </p>
30.       </body>
31.   </html>
```

运行效果如图 3-25 所示。

图 3-25 CSS 属性 font-family 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 font-family 不同属性值的效果。在首部标签<head>和</head>之间以类选择器的方式定义了 CSS 属性 font-family 的三个不同属性值,其中类名称定义为 style01、style02 和 style03,并将其应用于三个不同的段落元素中。

属性值为从系统中任选的三款较有特色的字体:"AR DELANEY""French Script MT"以及"微软雅黑 Light",分别用于显示描边、花体字和黑体字效果。由于这三款字体的名称都是多个单词组成中间有空格,因此属性值必须加上引号。如果字体名称为单个单词(例如 Arial),引号可以省略不写。

2. 字体风格 font-style

CSS 中的 font-style 属性可以用于设置字体风格是否为斜体字。该属性有三种取值,如表 3-24 所示。

表 3-24 CSS 属性 font-style 取值一览表

属 性 值	解 释
normal	正常字体
italic	斜体字
oblique	倾斜字体

【例 3-25】 CSS 属性 font-style 的简单应用

视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 font-style 的应用</title>
        <style>
            .style01 {font-style: normal}       /*正常字体*/
            .style02 {font-style: italic}       /*斜体字*/
            .style03 {font-style: oblique}      /*倾斜字体*/
        </style>
    </head>
    <body>
        <h3>CSS 属性 font-style 的应用</h3>
        <hr />
        <p class="style01">
            正常字体
        </p>
        <p class="style02">
            斜体字
        </p>
        <p class="style03">
            倾斜字体
        </p>
    </body>
</html>
```

运行效果如图 3-26 所示。

图 3-26 CSS 属性 font-style 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 font-style 不同属性值的效果。在首部标签<head>和</head>之间以类选择器的方式定义了 CSS 属性 font-style 的三个不同属性值(正常字体、斜体字和倾斜字体),其中类名称定义为 style01、style02 和 style03,并将其应用于三个不同的段落元素中,从而显示最终效果。

3. 字体变化 font-variant

CSS 中的 font-variant 属性可以用于设置字体变化。该属性有两种取值,如表 3-25 所示。

表 3-25 CSS 属性 font-variant 取值一览表

属 性 值	解 释
normal	正常字体
small-caps	小号字的大写字母

如果当前页面的指定字体不支持 small-caps 这种形式,则显示为正常大小字号的大写字母。

【例 3-26】 CSS 属性 font-variant 的简单应用

视频讲解

```
<!DOCTYPE html>
<html>
```

```
    <head>
        <meta charset = "utf-8">
        <title>CSS 属性 font - variant 的应用</title>
        <style>
            .style01 {font - variant: normal}
            .style02 {font - variant: small - caps}     /* 全大写,但是比正常大写字母小一号 */
        </style>
    </head>
    <body>
        <h3>CSS 属性 font - variant 的应用</h3>
        <hr />
        <p class = "style01">
            Normal
        </p>
        <p class = "style02">
            Small Caps
        </p>
        <p class = "style02">
            small caps
        </p>
    </body>
</html>
```

运行效果如图 3-27 所示。

图 3-27 CSS 属性 font-variant 的应用效果

【代码说明】

本示例包含了三个段落元素<p>,用其测试 CSS 属性 font-variant 不同属性值的效果。在首部标签<head>和</head>之间以类选择器的方式定义了 CSS 属性 font-style 的两个不同属性值(normal 和 small-caps),其中类名称定义为 style01 和 style02,并将其应用于三个不同的段落元素中,从而显示最终效果。

其中,class="style01"的段落元素<p>显示为正常字体效果,用于作对比案例。第二个和第三个段落元素<p>使用了相同的 class="style02",但是由图 3-27 可见,首字母的显示效果不同。原因是当文本内容中原先就存在大写字母时,使用 small-caps 属性值会将这些大写字母的字号显示为正常字体大小,而其他小写字母转换为大写字母后是小一号的字体大小,从而形成了一种特有的风格。

4. 字体粗细 font-weight

CSS 中的 font-weight 属性用于控制字体的粗细程度。该属性有五种取值,如表 3-26 所示。

表 3-26 CSS 属性 font-weight 取值

属 性 值	解 释
normal	标准正常字体,也是 font-weight 的默认值
bold	加粗字体
bolder	更粗的字体
lighter	更细的字体
100—900	[100,900]的整数,每个数字相差 100。数字越大字体越粗。其中 400 等同于 normal,700 等同于 bold

【例 3-27】 CSS 属性 font-weight 的简单应用


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 font-weight 的应用</title>
        <style>
            .style01 {
                font-weight: normal
            }
            .style02 {
                font-weight: bold
            }
            .style03 {
                font-weight: 100
            }
            .style04 {
                font-weight: 400
            }
            .style05 {
                font-weight: 900
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 font-weight 的应用</h3>
        <hr />
        <p class="style01">
            测试段落(正常字体)
        </p>
        <p class="style02">
            测试段落(粗体字)
        </p>
        <p class="style03">
            测试段落(100)
        </p>
        <p class="style04">
            测试段落(400)
        </p>
        <p class="style05">
            测试段落(900)
        </p>
    </body>
</html>
```

运行效果如图 3-28 所示。

图 3-28 CSS 属性 font-weight 的应用效果

【代码说明】

本示例包含了五个段落元素<p>,用其测试 CSS 属性 font-weight 不同属性值的效果。在首部标签<head>和</head>之间以类选择器的方式定义了 CSS 属性 font-weight 的五个不同属性值(normal、bold、100、400 和 900),并将其应用于这五个段落元素中。

由图 3-28 可见,使用数值的方式可以有更多的

选择。就浏览器的实际显示效果而言，100～400 的显示效果相似，500～900 的显示效果相似。

5. 字体大小 font-size

在 CSS 中，font-size 属性用于设置字体大小。font-size 的属性值为长度值，可以使用绝对单位或相对单位。绝对单位使用的是固定尺寸，不允许用户在浏览器中更改文本大小，采用了物理度量单位，例如，cm、mm、px 等；相对单位是相对于周围的参照元素进行设置大小，允许用户在浏览器中更改字体大小，字体相对单位有 em、ch 等。例如：

```
p{font-size:30px}
h1{font-size: 2em}
h2{font-size:120%}
```

关于字体大小的设置，常见用法是使用 px、em 或百分比(%)来显示字体尺寸。

- px——含义为像素，1px 指的是屏幕上显示的一个小点，它是绝对单位。
- em——含义为当前元素的默认字体尺寸，是相对单位。浏览器默认字体大小是 16px，因此在用户未作更改的情况下，1em＝16px。
- %——含义为相对于父元素的比例，例如，20%指的就是父元素宽度的 20%，也是一个相对单位。

【例 3-28】 CSS 属性 font-size 的简单应用


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 font-size 的应用</title>
        <style>
            .style01 {
                font-size: 16px
            }
            .style02 {
                font-size: 1em
            }
            .style03 {
                font-size: 32px
            }
            .style04 {
                font-size: 2em
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 font-size 的应用</h3>
        <hr />
        <p class="style01">
            测试段落，字体大小为 16 像素
        </p>
        <p class="style02">
            测试段落，字体大小为 1em
        </p>
        <p class="style03">
            测试段落，字体大小为 32 像素
        </p>
```

```
33.          <p class = "style04">
34.              测试段落,字体大小为 2em
35.          </p>
36.      </body>
37.  </html>
```

运行效果如图 3-29 所示。

图 3-29　CSS 属性 font-size 的应用效果

【代码说明】

本示例包含了四个段落元素<p>,用其测试 CSS 属性 font-size 不同属性值的效果。在首部标签<head>和</head>之间以类选择器的方式定义了 CSS 属性 font-size 的两种不同类型取值：绝对值 px 和相对值 em,并将其应用于这四个段落元素中。

由图 3-29 可见,font-size 属性值声明为 1em 与 16px 的前两个段落元素的字体大小显示效果是完全一样的,同样 2em 与 32px 也是一样。这是由于浏览器默认的字体大小为 16 像素,因此在用户未作更改的情况下,1em 等同于 16px。

6. 字体简写 font

CSS 中的 font 属性可以用于概括其他五种字体属性,将相关属性值汇总写在同一行。当需要为同一个元素声明多项字体属性时,可以使用 font 属性进行简写。声明顺序如下：

```
[font-style]  [font-variant]  [font-weight]  [font-size]  [font-family]
```

属性值之间用空格隔开,如果其中某个属性没有规定可以省略不写。例如：

```
p{
  font-style:italic;
  font-weight:bold;
  font-size:20px;
}
```

上述代码使用 font 属性可以简写为：

```
p{font: italic bold 20px}
```

其效果完全相同。

3.5.5　CSS 超链接

HTML 中的超链接元素<a>和其他元素类似,有一些通用 CSS 属性可以设置,比如字体大小、字体颜色、背景颜色等。除此之外,超链接元素<a>还可以根据其所处的四种不同的状态分别设置 CSS 样式。超链接的四种状态如表 3-27 所示。

表 3-27　超链接的四种状态一览表

状态名称	解　释	状态名称	解　释
a:link	未被访问的超链接	a:hover	鼠标悬浮在上面的超链接
a:visited	已被访问的超链接	a:active	正在被点击的超链接

为超链接设置不同状态的 CSS 样式时必须遵循两条规则：一是 a:hover 的声明必须在 a:link 和 a:visited 之后；二是 a:active 的声明必须在 a:hover 之后；否则声明有可能失效。

【例 3-29】 超链接不同状态的简单 CSS 应用


视频讲解

```
<!DOCTYPE html>
<html>
<head>
<meta charset = "utf-8">
<title>CSS 属性超链接的应用</title>
<style>
a:link,a:visited{
   display:block;                    /* 块级元素 */
   text-decoration:none;             /* 取消下画线 */
   color:white;                      /* 字体为白色 */
   font-weight:bold;                 /* 字体加粗 */
   font-size:25px;                   /* 字体大小为 25px */
   background-color:#7BF;            /* 设置背景颜色 */
   width:200px;                      /* 宽度 200 像素 */
   height:30px;                      /* 高度 30 像素 */
   text-align:center;                /* 文本居中显示 */
   line-height:30px;                 /* 行高 30 像素 */
}
a:hover,a:active{
   background-color:#0074E8;         /* 设置背景颜色 */
}
</style>
</head>
<body>
<h3>CSS 属性超链接的应用</h3>
<hr />
<a href = "http://www.baidu.com">百度</a>
</body>
</html>
```

运行效果如图 3-30 所示。

(a) 未被访问的超链接效果　　(b) 鼠标悬浮在上面的超链接效果

图 3-30 CSS 属性 font-image 的应用效果

【代码说明】

本示例包含了单一的超链接元素<a>作为示例。然后使用 CSS 内部样式表的形式分别为超链接的四种状态设置样式要求。当有多个状态使用同一个样式时，可以在一起进行声明，之间用逗号隔开即可。本例中未访问和已访问状态共用一种样式，鼠标悬浮在上面和正在点击状态共用另一种样式。

为使超链接元素形成仿按钮风格，为其定义 display 属性为 block，使之成为块级元素，从而可以为其设置尺寸。本例设置 text-decoration 属性值为 none，取消了超链接原有的下

画线样式,并在鼠标悬浮和点击状态中设置了元素背景颜色的加深,从而实现动态效果。

3.5.6 CSS 列表

CSS 对于 HTML 列表元素的样式设置主要在于规定各项列表前面的标志(marker)类型。在之前第 2 章中提到了三种列表类型:有序列表、无序列表和定义列表。其中有序列表默认的标记样式为标准阿拉伯数字(1,2,3,4,…),而无序列表默认的标记样式是实心圆点。和列表有关的属性如表 3-28 所示。

表 3-28 CSS 列表属性一览表

属性名称	解释
list-style-type	设置列表标志类型
list-style-image	设置列表标志图标
list-style-position	设置列表标志位置
list-style-type	上述所有属性的综合简写方式

1. 样式类型 list-style-type

CSS 中的 list-style-type 属性可以用于设置列表的标志样式。该属性在 CSS2 版本已有 21 种取值内容,如表 3-29 所示。

表 3-29 CSS 属性 list-style-type 常见取值一览表

属性值	解释
none	无标记符号
disc	list-style-type 属性的默认值,样式为实心圆点
circle	空心圆
square	实心方块
decimal	阿拉伯数字(1,2,3,4,…)
decimal-leading-zero	带有 0 开头的阿拉伯数字(01,02,03,04,…),该属性值不被 IE 浏览器支持
upper-roman	大写罗马数字(Ⅰ,Ⅱ,Ⅲ,Ⅳ,…)
lower-roman	小写罗马数字(ⅰ,ⅱ,ⅲ,ⅳ,…)
upper-alpha	大写英文字母(A,B,C,D,…)
lower-alpha	小写英文字母(a,b,c,d,…)
upper-latin	大写拉丁文字母(A,B,C,D,…),该属性值不被 IE 浏览器支持
lower-latin	小写拉丁文字母(a,b,c,d,…),该属性值不被 IE 浏览器支持
lower-greek	小写希腊字母(alpha,beta,gamma,…),该属性值不被 IE 浏览器支持
hebrew	传统的希伯来编号方式
armenian	传统的亚美尼亚编号方式
georgian	传统的乔治亚编号方式
cjk-ideographic	W3C 组织称其为简单的表意数字,经测试在安装有中文字体的系统上运行可显示汉字(一,二,三,四,…)
hiragana	日语平假名(日本字母的草体字)的编号
katakana	日语片假名的编号
hiragana-iroha	日语平假名-伊吕波形的编号
katakana-iroha	日语片假名-伊吕波形的编号

其中拉丁字母与英文字母显示效果完全相同,只不过拉丁字母的属性值不被 IE 浏览器所支持。从表格的最后四行可以看到日语的平假名(hiragana)和片假名(katakana)编号都带有伊吕波(iroha)型的变种。伊吕波是源自日本的一首古老的歌谣,最早见于 1079 年,这

首歌包含了全部日语音节，类似于英文 ABC 字母歌。这里日语编号的多样性可以理解为中文序号有一、二、三、四也有甲、乙、丙、丁这种概念。

【例 3-30】 CSS 属性 list-style-type 的应用

本示例用于演示对不同列表设置 21 种 CSS 属性 list-style-type 属性值的效果。


```
<!DOCTYPE html>
<html>
<head>
<meta charset="utf-8">
<title>CSS 属性 list-style-type 的应用</title>
<style>
div{
  border:1px solid;
  width:235px;
  height:125px;
  float:left;
  margin:5px;
}
.none{ list-style-type: none}
.disc{ list-style-type: disc}
.circle{ list-style-type:circle}
.square{ list-style-type: square}
.decimal{ list-style-type: decimal}
.decimal-leading-zero{ list-style-type:decimal-leading-zero}
.upper-roman{ list-style-type: upper-roman}
.lower-roman{ list-style-type: lower-roman}
.upper-alpha{ list-style-type:upper-alpha}
.lower-alpha{ list-style-type: lower-alpha}
.upper-latin{ list-style-type:upper-latin}
.lower-latin{ list-style-type: lower-latin}
.lower-greek{ list-style-type: lower-greek}
.hebrew{ list-style-type: hebrew}
.armenian{ list-style-type: armenian}
.georgian{ list-style-type: georgian}
.cjk-ideographic{ list-style-type: cjk-ideographic}
.hiragana{ list-style-type: hiragana}
.hiragana-iroha{ list-style-type: hiragana-iroha}
.katakana{ list-style-type: katakana}
.katakana-iroha{ list-style-type: katakana-iroha}
</style>
</head>
<body>
<h3>CSS 属性 list-style-type 的应用</h3>
<hr />
<div>
<h4>属性值为 none</h4>
<ul class="none">
<li>跳水</li>
<li>举重</li>
<li>击剑</li>
</ul>
</div>

<div>
<h4>属性值为 disc</h4>
```

```
51.  <ul class = "disc">
52.  <li>跳水</li>
53.  <li>举重</li>
54.  <li>击剑</li>
55.  </ul>
56.  </div>
57.
58.  <div>
59.  <h4>属性值为 circle</h4>
60.  <ul class = "circle">
61.  <li>跳水</li>
62.  <li>举重</li>
63.  <li>击剑</li>
64.  </ul>
65.  </div>
66.
67.  <div>
68.  <h4>属性值为 square</h4>
69.  <ul class = "square">
70.  <li>跳水</li>
71.  <li>举重</li>
72.  <li>击剑</li>
73.  </ul>
74.  </div>
75.
76.  <div>
77.  <h4>属性值为 decimal</h4>
78.  <ul class = "decimal">
79.  <li>跳水</li>
80.  <li>举重</li>
81.  <li>击剑</li>
82.  </ul>
83.  </div>
84.
85.  <div>
86.  <h4>属性值为 decimal - leading - zero</h4>
87.  <ul class = "decimal - leading - zero">
88.  <li>跳水</li>
89.  <li>举重</li>
90.  <li>击剑</li>
91.  </ul>
92.  </div>
93.
94.  <div>
95.  <h4>属性值为 upper - roman</h4>
96.  <ul class = "upper - roman">
97.  <li>跳水</li>
98.  <li>举重</li>
99.  <li>击剑</li>
100. </ul>
101. </div>
102.
103. <div>
104. <h4>属性值为 lower - roman</h4>
105. <ul class = "lower - roman">
106. <li>跳水</li>
```

```
107. <li>举重</li>
108. <li>击剑</li>
109. </ul>
110. </div>
111.
112. <div>
113. <h4>属性值为 upper-alpha</h4>
114. <ul class="upper-alpha">
115. <li>跳水</li>
116. <li>举重</li>
117. <li>击剑</li>
118. </ul>
119. </div>
120.
121. <div>
122. <h4>属性值为 lower-alpha</h4>
123. <ul class="lower-alpha">
124. <li>跳水</li>
125. <li>举重</li>
126. <li>击剑</li>
127. </ul>
128. </div>
129.
130. <div>
131. <h4>属性值为 upper-latin</h4>
132. <ul class="upper-latin">
133. <li>跳水</li>
134. <li>举重</li>
135. <li>击剑</li>
136. </ul>
137. </div>
138.
139. <div>
140. <h4>属性值为 lower-latin</h4>
141. <ul class="lower-latin">
142. <li>跳水</li>
143. <li>举重</li>
144. <li>击剑</li>
145. </ul>
146. </div>
147.
148. <div>
149. <h4>属性值为 lower-greek</h4>
150. <ul class="lower-greek">
151. <li>跳水</li>
152. <li>举重</li>
153. <li>击剑</li>
154. </ul>
155. </div>
156.
157. <div>
158. <h4>属性值为 hebrew</h4>
159. <ul class="hebrew">
160. <li>跳水</li>
161. <li>举重</li>
162. <li>击剑</li>
```

```
163. </ul>
164. </div>
165.
166. <div>
167. <h4>属性值为 armenian</h4>
168. <ul class = "armenian">
169. <li>跳水</li>
170. <li>举重</li>
171. <li>击剑</li>
172. </ul>
173. </div>
174.
175. <div>
176. <h4>属性值为 georgian</h4>
177. <ul class = "georgiann">
178. <li>跳水</li>
179. <li>举重</li>
180. <li>击剑</li>
181. </ul>
182. </div>
183.
184. <div>
185. <h4>属性值为 cjk - ideographic</h4>
186. <ul class = "cjk - ideographic">
187. <li>跳水</li>
188. <li>举重</li>
189. <li>击剑</li>
190. </ul>
191. </div>
192.
193. <div>
194. <h4>属性值为 hiragana</h4>
195. <ul class = "hiragana">
196. <li>跳水</li>
197. <li>举重</li>
198. <li>击剑</li>
199. </ul>
200. </div>
201.
202. <div>
203. <h4>属性值为 katakana</h4>
204. <ul class = "katakana">
205. <li>跳水</li>
206. <li>举重</li>
207. <li>击剑</li>
208. </ul>
209. </div>
210.
211. <div>
212. <h4>属性值为 hiragana - irohaic</h4>
213. <ul class = "hiragana - iroha">
214. <li>跳水</li>
215. <li>举重</li>
216. <li>击剑</li>
217. </ul>
218. </div>
```

```
219.
220. <div>
221. <h4>属性值为 katakana - iroha</h4>
222. <ul class = "katakana - iroha">
223. <li>跳水</li>
224. <li>举重</li>
225. <li>击剑</li>
226. </ul>
227. </div>
228. </body>
229. </html>
```

运行效果如图 3-31 所示。

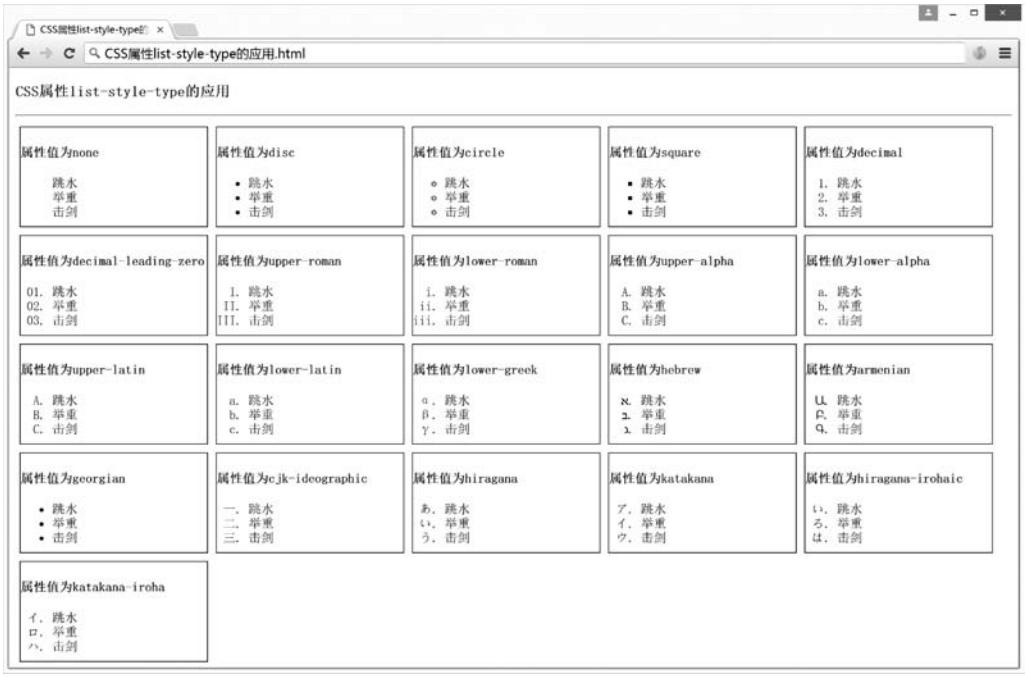


图 3-31 CSS 属性 list-style-type 的应用效果

【代码说明】

本示例包含了 21 组列表元素<ul>,每组列表的项目标签<li>均用于显示了三种运动类型(跳水、举重、击剑)作为示例。使用区域元素 对每组列表效果进行分块显示,并事先为 元素设置统一标准:带有 1 像素宽的实线边框,宽 235 像素、高 125 像素,各边外边距为 5 像素,浮动方式为左对齐。

以类选择器的方式定义了 CSS 属性 list-style-type 的 21 种效果,类名称定义为和属性值一样的文字内容。将这 21 种效果分配给本示例中的 21 组列表,即可查看最终效果。

注意:部分列表样式在 IE 浏览器中不被支持,建议使用 Chrome 或其他浏览器进行预览。

2. 样式图片 list-style-image

CSS 中的 list-style-image 属性可以用于设置列表的标志图标。标志图标可以是来源于本地或者网络的图像文件。如果已使用 list-style-image 属性声明了列表的标志图标,则不能同时使用 list-style-type 属性声明列表的标志类型,否则后者将无显示效果。

【例 3-31】 CSS 属性 list-style-image 的简单应用

使用自定义图片制作列表的标志图标。

视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 list-style-image 的应用</title>
        <style>
            .arrow {
                list-style-image: url(image/icon01.png)
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 list-style-image 的应用</h3>
        <hr />
        <ul class="arrow">
            <li>选项一</li>
            <li>选项二</li>
            <li>选项三</li>
        </ul>
    </body>
</html>
```

运行效果如图 3-32 所示。

【代码说明】

本示例包含了列表元素<ul>与其内部的三个列表选项元素<li>作为示例。列表图标 icon01.png 来源于与 HTML 文档在同一目录下的 image 文件夹。使用 list-style-image 属性可以为列表定义多样化的图标。

图 3-32 CSS 属性 list-style-image 的应用效果

3. 样式位置 list-style-position

CSS 中的 list-style-position 属性用于定义列表标志的位置,有三种属性值,如表 3-30 所示。

表 3-30 CSS 属性 list-style-position 属性值一览表

属 性 值	解 释
outside	list-style-position 属性的默认值,表示列表标志放置在文本左侧
inside	表示列表标志放置在文本内部,多行文本根据标志对齐
inherit	继承父元素的 list-style-position 属性值

【例 3-32】 CSS 属性 list-style-position 的简单应用

使用列表元素<ul>对比 list-style-position 属性值为 outside 和 inside 的显示区别。

视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
```

```
5.          <title>CSS 属性 list-style-position 的应用</title>
6.          <style>
7.              ul {
8.                  width: 280px;
9.                  border: 1px solid
10.             }
11.             .outside {
12.                 list-style-position: outside
13.             }
14.             .inside {
15.                 list-style-position: inside
16.             }
17.         </style>
18.     </head>
19.     <body>
20.         <h3>CSS 属性 list-style-position 的应用</h3>
21.         <hr />
22.         <ul class="outside">
23.             <li>
24.                 本示例的 list-style-position 属性值为 outside。
25.             </li>
26.             <li>
27.                 本示例的 list-style-position 属性值为 outside。
28.             </li>
29.             <li>
30.                 本示例的 list-style-position 属性值为 outside。
31.             </li>
32.         </ul>
33.
34.         <ul class="inside">
35.             <li>
36.                 本示例的 list-style-position 属性值为 inside。
37.             </li>
38.             <li>
39.                 本示例的 list-style-position 属性值为 inside。
40.             </li>
41.             <li>
42.                 本示例的 list-style-position 属性值为 inside。
43.             </li>
44.         </ul>
45.     </body>
46. </html>
```

运行效果如图 3-33 所示。

【代码说明】

本示例包含了两个列表元素<ul>，分别测试 list-style-position 属性值为 outside 和 inside 的两种情况，并使用 CSS 内部样式表设置了列表元素<ul>的样式：宽度为 280 像素，并带有宽 1 像素的实线边框。每个列表元素内部包含了三个列表选项元素<li>作为示例。

由图 3-33 可见，当 list-style-position 属性值为 outside 时，列表左边的标志点是独立于文本外侧的，当列表文字内容较多需要换行时，第二行的文字内容可以和第一行对齐；而当 list-style-position 属性值为 inside 时，列表的标志点是嵌入文本中的，列表文字内容如果换行应和该标志点对齐。

图 3-33　CSS 属性 list-style-position 的应用效果

4. 样式简写 list-style

CSS 中的 list-style 属性可以用于概括其他三种字体属性，将相关属性值汇总写在同一行。当需要为同一个列表元素声明多项列表属性时可以使用 list-style 属性进行简写。声明顺序如下：

```
[list-style-type]  [list-style-position]  [list-style-image]
```

属性值之间用空格隔开，如果其中某个属性没有规定可以省略不写。

例如：

```
ul{
   list-style-type: circle;
   list-style-position: outside
}
```

上述代码使用 list-style 属性可以简写为：

```
ul{ list-style: circle outside}
```

其效果完全相同。

3.5.7　CSS 表格

本节将介绍如何对网页上的表格进行修饰。和 CSS 表格有关的属性如表 3-31 所示。

表 3-31　CSS 表格相关属性

属 性 名 称	解　　释
border-collapse	用于设置表格的边框样式为双线或单线
border-spacing	用于设置表格中双线边框的分割距离
caption-side	用于设置表格中的标题位置
empty-cells	用于定义表格中空单元格边框和背景的显示方式
table-layout	用于规定表格的布局方式，包括固定表格布局和根据内容调整布局

除以上五种属性设置外，在 CSS 中一些通用属性设置同样也可以用于表格元素。例如，字体颜色(color)、背景(background)、文本对齐(text-align)、边框(border)、内边距

(padding)、宽度(width)和高度(height)等,这里不再展开详细说明。

1. 折叠边框 border-collapse

在默认情况下,表格的边框如果设置为实线,则会显示为双层线条的样式效果。CSS 中的 border-collapse 属性用于设置是否将表格的双层边框折叠为单一线条边框,该属性有三种属性值,如表 3-32 所示。

表 3-32 CSS 属性 border-collapse 属性值

属性名称	解释
separate	border-collapse 属性的默认值,边框为分开的双层线条效果
collapse	边框会合并为单一线条的边框
inherit	继承父元素的 border-collapse 属性值

【例 3-33】 CSS 属性 border-collapse 的简单应用

使用表格元素对比 border-collapse 属性值为 separate 和 collapse 的显示效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 border-collapse 的应用</title>
        <style>
            .separate {
                border-collapse: separate
            }
            .collapse {
                border-collapse: collapse
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 border-collapse 的应用</h3>
        <hr />
        <table border="1" class="separate">
            <caption>
                双线边框效果
            </caption>
            <tr>
        <td>年份</td>
        <td>第一季度</td>
        <td>第二季度</td>
        <td>第三季度</td>
      </tr>
            <tr><td>2014</td><td>100</td><td>200</td><td>300</td></tr>
            <tr><td>2015</td><td>150</td><td>250</td><td>350</td></tr>
            <tr><td>2016</td><td>200</td><td>300</td><td>400</td></tr>
        </table>
        <br />

        <table border="1" class="collapse">
            <caption>
                折叠边框效果
            </caption>
            <tr>
        <td>年份</td>
```

```
            <td>第一季度</td>
            <td>第二季度</td>
            <td>第三季度</td>
        </tr>
            <tr><td>2014</td><td>100</td><td>200</td><td>300</td></tr>
            <tr><td>2015</td><td>150</td><td>250</td><td>350</td></tr>
            <tr><td>2016</td><td>200</td><td>300</td><td>400</td></tr>
        </table>
    </body>
</html>
```

运行效果如图 3-34 所示。

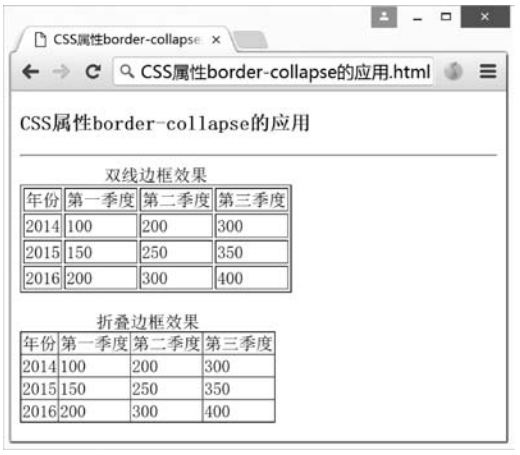


图 3-34 CSS 属性 border-collapse 的应用效果

【代码说明】

本示例包含了两个表格元素<table>，分别用于测试 border-collapse 属性值为 separate 和 collapse 两种情况，并使用 CSS 内部样式表设置了类选择器，将这两种属性值分别应用于其中一个表格元素。每个表格元素中包含了 4 行 4 列的单元格，表格中的数据内容为测试样例与本示例无关。为表格元素添加了属性 border="1"，以便形成宽度为 1 像素的实线边框。

2. 边框距离 border-spacing

CSS 中的 border-spacing 属性用于定义表格中双线边框的分割距离，该属性有三种属性值，如表 3-33 所示。

表 3-33 CSS 属性 border-spacing 属性值

属 性 值	解 释
长度值	表示水平和垂直方向上的距离
长度值 1 长度值 2	长度值 1 用于表示水平方向的距离，长度值 2 用于表示垂直方向的距离
inherit	继承父元素的 border-spacing 属性值

注意：border-spacing 属性只在表格能显示边框并且边框的 border-collapse 属性值为默认值 separate 时生效，否则该属性将被忽略。

【例 3-34】 CSS 属性 border-spacing 的简单应用

为表格设置不同的边框距离。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 border-spacing 的应用</title>
        <style>
            .style01 {
                border-spacing: 10px
            }
            .style02 {
                border-spacing: 50px 10px
            }
```

```
        </style>
    </head>
    <body>
        <h3>CSS 属性 border-spacing 的应用</h3>
        <hr />
        <table border="1" class="style01">
            <caption>
                单个属性值效果:边框距离 10 像素
            </caption>
            <tr>
                <td>年份</td>
        <td>第一季度</td>
        <td>第二季度</td>
        <td>第三季度</td>
            </tr>
            <tr><td>2014</td><td>100</td><td>200</td><td>300</td></tr>
            <tr><td>2015</td><td>150</td><td>250</td><td>350</td></tr>
            <tr><td>2016</td><td>200</td><td>300</td><td>400</td></tr>
        </table>
        <br />

        <table border="1" class="style02">
            <caption>
                两个属性值效果:水平方向 50 像素,垂直方向 10 像素
            </caption>
            <tr>
                <td>年份</td>
        <td>第一季度</td>
        <td>第二季度</td>
        <td>第三季度</td>
            </tr>
            <tr><td>2014</td><td>100</td><td>200</td><td>300</td></tr>
            <tr><td>2015</td><td>150</td><td>250</td><td>350</td></tr>
            <tr><td>2016</td><td>200</td><td>300</td><td>400</td></tr>
        </table>
    </body>
</html>
```

运行效果如图 3-35 所示。

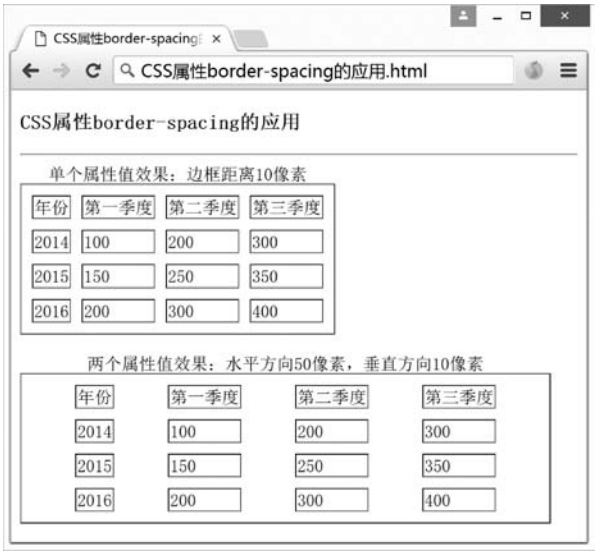


图 3-35 CSS 属性 border-spacing 的应用效果

【代码说明】

本示例包含了两个表格元素<table>,分别用于测试 border-spacing 属性值为单个长度值和两个长度值的情况,并使用 CSS 内部样式表设置了类选择器将这两种情况分别应用于其中一个表格元素。每个表格元素中包含了 4 行 4 列的单元格,表格中的数据内容为测试样例,与本示例无关。

注:由于表格的默认效果就是分割边框(border-collapse 属性值为 separate),因此无须特别声明。只需要为表格元素添加属性 border="1",以便形成宽度为 1 像素的实线边框。

3. 标题位置 caption-side

CSS 中的 caption-side 属性用于定义表格中标题的位置,有三种属性值如表 3-34 所示。

表 3-34 CSS 属性 caption-side 属性值

属 性 值	解 释
top	caption-side 属性的默认值,表示标题在表格上方
bottom	表示标题在表格下方
inherit	继承父元素的 caption-side 属性值

【例 3-35】 CSS 属性 caption-side 的简单应用

为表格设置出现在底端的标题。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 caption-side 的应用</title>
        <style>
            caption {
                caption-side: bottom
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 caption-side 的应用</h3>
        <hr />
        <table border="1">
            <caption>
                我是显示在表格底端的标题
            </caption>
            <tr>
                <td>年份</td>
        <td>第一季度</td>
        <td>第二季度</td>
        <td>第三季度</td>
            </tr>
            <tr>
                <td>2014</td><td>100</td><td>200</td><td>300</td>
            </tr>
            <tr>
                <td>2015</td><td>150</td><td>250</td><td>350</td>
            </tr>
            <tr>
                <td>2016</td><td>200</td><td>300</td><td>400</td>
```

```
33.                 </tr>
34.             </table>
35.         </body>
36.     </html>
```

运行效果如图 3-36 所示。

图 3-36 CSS 属性 caption-side 的应用效果

【代码说明】

本示例包含了单个表格元素<table>，用于测试 caption-side 属性值为 bottom 的情况，并使用 CSS 内部样式表设置了元素选择器，将该属性值应用于表格标题标签<caption>，并为表格元素添加属性 border="1"，以便形成宽度为 1 像素的实线边框。

每个表格元素中包含了 4 行 4 列的单元格，表格中的数据内容为测试样例，与本示例无关。

4. 空单元格 empty-cells

CSS 中的 empty-cells 属性用于定义表格中空单元格边框和背景的显示方式。该属性有三种属性值，如表 3-35 所示。

表 3-35 CSS 属性 empty-cells 属性值一览表

属 性 值	解 释
show	empty-cells 属性的默认值，表示正常显示空白单元格的边框与背景
hide	表示不显示空白单元格的边框与背景
inherit	继承父元素的 empty-cells 属性值

【例 3-36】 CSS 属性 empty-cells 的简单应用

为表格中的空白单元格设置不显示边框的效果。


```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset="utf-8">
5.          <title>CSS 属性 empty-cells 的应用</title>
6.          <style>
7.              table {
8.                  empty-cells: hide
9.              }
10.         </style>
11.     </head>
12.     <body>
13.         <h3>CSS 属性 empty-cells 的应用</h3>
14.         <hr />
15.         <table border="1">
16.             <caption>
17.                 隐藏空单元格的边框效果
18.             </caption>
19.             <tr>
20.                 <td>年份</td>
```

```
21.          <td>第一季度</td>
22.          <td>第二季度</td>
23.          <td>第三季度</td>
24.              </tr>
25.              <tr>
26.                  <td>2014</td><td>100</td><td>200</td><td>300</td>
27.              </tr>
28.              <tr>
29.                  <td>2015</td><td>150</td><td>250</td><td>350</td>
30.              </tr>
31.              <tr>
32.                  <td>2016</td><td>200</td><td>300</td><td></td>
33.              </tr>
34.          </table>
35.      </body>
36.  </html>
```

运行效果如图 3-37 所示。

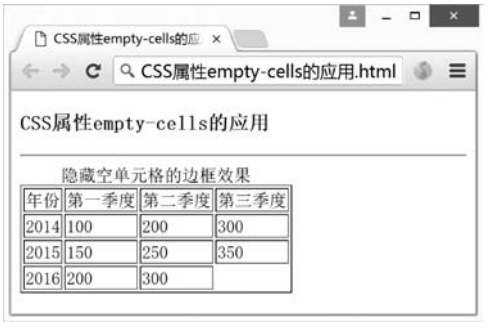


图 3-37 CSS 属性 empty-cells 的应用效果

【代码说明】

本示例包含了单个表格元素<table>,用于测试 empty-cells 属性值为 hide 的情况,并使用 CSS 内部样式表设置了元素选择器,将该属性值应用于表格标签<table>。每个表格元素中包含了 4 行 4 列的单元格,表格中的数据内容为测试样例与本示例无关。

注:由于表格的默认效果就是分割边框(border-collapse 属性值为 separate),因此无须特别声明。只需要为表格元素添加属性 border="1",以便形成宽度为 1 像素的实线边框。

5. 表格布局 table-layout

CSS 中的 table-layout 属性用于规定表格的布局方式,包括固定表格布局和根据内容调整布局。该属性有三种属性值,如表 3-36 所示。

表 3-36 CSS 属性 table-layout 属性值

属 性 值	解 释
automatic	empty-cells 属性的默认值,表示单元格的宽度由内容决定
fixed	单元格的宽度由样式设置决定
inherit	继承父元素的 table-layout 属性值

【例 3-37】 CSS 属性 table-layout 的简单应用

为表格设置不同的边框距离。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset="utf-8">
5.          <title>CSS 属性 table-layout 的应用</title>
6.          <style>
7.              table {
```

```
8.                  width: 100 %
9.              }
10.             .fixed {
11.                 table - layout: fixed
12.             }
13.             .automated {
14.                 table - layout: automated
15.             }
16.         </style>
17.     </head>
18.     <body>
19.         <h3>CSS 属性 table - layout 的应用</h3>
20.         <hr />
21.         <table border = "1" class = "fixed">
22.             <caption>
23.                 固定列宽的表格
24.             </caption>
25.             <tr>
26.                 <td>年份</td><td>第一季度</td><td>第二季度</td><td>第三季度
                    </td>
27.             </tr>
28.             <tr>
29.                 <td>2016</td><td>200</td><td>300</td><td>400000000000000
                    </td>
30.             </tr>
31.         </table>
32.         <br />
33.
34.         <table border = "1" class = "automated">
35.             <caption>
36.                 随内容自动调整列宽的表格
37.             </caption>
38.             <tr>
39.                 <td>年份</td><td>第一季度</td><td>第二季度</td><td>第三季度
                    </td>
40.             </tr>
41.             <tr>
42.                 <td>2016</td><td>200</td><td>300</td><td>400000000000000
                    </td>
43.             </tr>
44.         </table>
45.     </body>
46. </html>
```

运行效果如图 3-38 所示。

【代码说明】

本示例包含了两个表格元素<table>，分别用于测试 table-layout 属性值为 fixed 和 automated 的情况，并使用 CSS 内部样式表设置了类选择器，将这两种情况分别应用于其中一个表格元素，并设置表格元素的宽度为 100%，即与页面等宽。为表格元素添加属性 border="1"，以便形成宽度为 1 像素的实线边框。每个表格元素中包

图 3-38 CSS 属性 table-layout 的应用效果

含了 2 行 4 列的单元格，表格中的数据内容为测试样例与本示例无关。

由图 3-38 可见，当 table-layout 的属性值为 fixed 时，所有单元格的列宽是平均分配的，即使最后一个单元格的测试数据内容较多，也只能溢出单元格而不会改变单元格宽度。而 table-layout 的属性值为 automated 时，单元格的列宽会随着内容的多少自动调整。由于 automated 是 table-layout 属性的默认值，也可以忽略不写。

3.6 CSS 定位

CSS 定位可以将 HTML 元素放置在页面上指定的任意地方。CSS 定位的原理是把页面左上角的点定义为坐标为(0,0)的原点，然后以像素为单位将整个网页构建成一个坐标系统。其中 x 轴与数学坐标系方向相同，越往右数字越大；y 轴与数学坐标系方向相反，越往下数字越大。

本节主要介绍四种定位的方式：绝对定位、相对定位、层叠效果和浮动。联合使用这些定位方式，可以创建更为复杂和准确的布局。

3.6.1 绝对定位

绝对定位指的是通过规定 HTML 元素在水平和垂直方向上的位置来固定元素，基于绝对定位的元素不占据空间。

使用绝对定位需要将 HTML 元素的 position 属性值设置为 absolute(绝对的)，并使用四种关于方位的属性关键词 left(左边)、right(右边)、top(顶部)、bottom(底端)中的部分内容设置元素的位置。一般来说从水平和垂直方向各选一个关键词即可。

例如，需要将段落元素<p>放置在距离页面顶端 150 像素、左边 100 像素的位置：

```
p{
    position: absolute;
    top:150px;
    left:100px
}
```

注意：绝对定位的位置声明是相对于已定位的并且包含关系最近的祖先元素。如果当前需要被定位的元素没有已定位的祖先元素做参考值，则相对于整个网页。例如，同样是上面关于段落元素<p>的样式声明，如果该段落元素放置在一个已经定位的 元素内部，则指的是距离这个 元素的顶端 150 像素、左边 100 像素的位置。

【例 3-38】 CSS 绝对定位的应用

使用两个相同 CSS 样式的段落元素<p>对比不同的绝对定位效果。


视频讲解

```
1.  <!DOCTYPE html>
2.  <html>
3.      <head>
4.          <meta charset = "utf-8">
5.          <title>CSS 绝对定位的应用</title>
6.          <style>
7.              p {
8.                  position: absolute;
9.                  width: 120px;
10.                 height: 120px;
11.                 top: 100px;
12.                 left: 0px;
13.                 background-color: #C8EDFF;
```

```
14.             }
15.             div {
16.                 position: absolute;
17.                 width: 300px;
18.                 height: 300px;
19.                 top: 80px;
20.                 left: 180px;
21.                 border: 1px solid;
22.             }
23.         </style>
24.     </head>
25.     <body>
26.         <h3>CSS 绝对定位的应用</h3>
27.         <hr />
28.         <p>
29.             该段落是相对于页面定位的,距离页面的顶端 100 像素,距离左边 0 像素
30.         </p>
31.         <div>
32.             我是相对于页面定位的 div 元素,距离顶端 80 像素,距离左边 180 像素
33.             <p>
34.                 该段落是相对于父元素 div 定位的,距离 div 元素的顶端 100 像素,距离 div
                    元素的左边 0 像素
35.             </p>
36.         </div>
37.     </body>
38. </html>
```

运行效果如图 3-39 所示。

图 3-39 CSS 绝对定位的应用效果

【代码说明】

本示例包含了两个样式完全相同的段落元素<p>,用于对比测试直接在页面中使用和嵌入已定位的 元素中的两种情况。并使用 CSS 内部样式表设置了元素选择器,为段落元素<p>和区域元素 定义样式。

其中两个段落元素<p>的统一样式设置为:宽和高均为 120 像素的矩形,带有背景颜色,并为其定义 position 属性值为 absolute 表示绝对定位,要求距离父元素的顶端 100 像

素、左边 0 像素。区域元素 的样式设置为：宽和高均为 300 像素的矩形，并带有宽 1 像素的实线边框。同样为 元素也定义了 position 属性值为 absolute 表示绝对定位，要求其距离父元素顶端 80 像素，左边 180 像素。

由图 3-39 可见，左边的段落元素<p>没有已定位的父元素，因此它的位置是相对于整个页面来计算的；而右边的段落元素<p>是包含于已定位的 元素中，所以其位置是相对于 元素的顶端和左边来进行计算的。因此虽然这两个段落元素<p>具有完全相同的 CSS 样式设置，它们出现在页面上的位置不一样。

3.6.2 相对定位

相对定位与绝对定位的区别在于它的参照点不是左上角的原点，而是该元素本身原先的起点位置。并且即使该元素偏移到了新的位置，也仍然从原始的起点处占据空间。

使用相对定位需要将 HTML 元素的 position 属性值设置为 relative(相对的)，并同样使用四种关于方位的属性关键词 left(左边)、right(右边)、top(顶部)、bottom(底端)中的部分内容设置元素的位置。一般来说，从水平和垂直方向各选一个关键词即可。

例如，需要将段落元素<p>放置在距离元素初始位置顶端 150 像素、左边 100 像素的位置：

```
p{
    position: relative;
    top:150px;
    left:100px
}
```

注意：相对定位的位置是相对于元素自身的正常初始位置而言的。因此即使是内容完全一样的相对定位代码作用于初始位置不同的多个元素上也仅能保证位移的方向一致，并不能代表这些元素最终将出现在相同的位置上。

【例 3-39】 CSS 相对定位的应用

使用三个相同 CSS 样式的段落元素<p>对比不同的相对定位效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 相对定位的应用</title>
        <style>
            div {
                width: 200px;
                height: 380px;
                border: 1px solid;
                margin-left: 50px;
            }
            p {

                width: 150px;
                height: 100px;
                background-color: #C8EDFF;
            }
            .left {
                position: relative;
                left: -50px;
```

```
            }
            .right {
                position: relative;
                left: 130px;
            }
        </style>
    </head>
    <body>
        <h3>CSS 相对定位的应用</h3>
        <hr />
        <div>
            <p>
                正常状态的段落
            </p>
            <p class = "left">
                相对自己正常的位置向左边偏移了 50 像素
            </p>
            <p class = "right">
                相对自己正常的位置向右边偏移了 130 像素
            </p>
        </div>
    </body>
</html>
```

运行效果如图 3-40 所示。

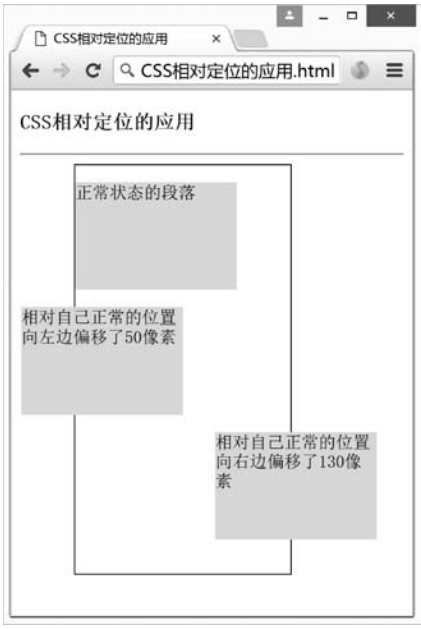


图 3-40 CSS 相对定位的应用效果

【代码说明】

本示例包含了三个样式相同的段落元素<p>，用于对比测试相对定位效果，并声明了一个带有实线边框效果的 元素包含这三个段落元素，以便对比段落元素位置的偏移量。

使用 CSS 内部样式表设置了元素选择器为段落元素<p>和区域元素 定义样式。其中段落元素<p>的统一样式设置为：宽 150 像素、高 100 像素，并带有背景颜色。区域元素 的样式设置为：宽 200 像素、高 380 像素的矩形，并带有 1 像素的实线边框。

第一个段落元素为正常显示效果，不做任何位置偏移设置，以便与后面两个段落元素进行位置对比。在 CSS 内部样式表中使用了类选择器为第二、三个段落元素设置分别向左和右边发生一定量的偏移，并将其 position 属性设置为 relative 表示相对定位模式。

如果这三个段落元素都没有做位置偏移会从上往下左对齐显示在 元素中。由图 3-40 可见，目前只有第一个段落元素显示位置正常，第二、三个段落元素均根据自己的初始位置发生了指定像素的偏移。

3.6.3 层叠效果

在 CSS 中，除了定义 HTML 元素在水平和垂直方向上的位置，还可以定义多个元素在

一起叠放的层次。使用属性 z-index 可以为元素规定层次顺序，其属性值为整数，并且该数值越大将叠放在越靠上的位置。例如，z-index 属性值为 99 的元素一定显示在 z-index 属性值为 10 的元素上面。

【例 3-40】 CSS 层叠效果的应用

使用 CSS 属性 z-index 制作扑克牌叠放效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 属性 z-index 的应用</title>
        <style>
            div {
                width: 182px;
                height: 253px;
                position: absolute;
            }
            #ten {
                background: url(image/ten.jpg) no-repeat;
                z-index: 1;
                left: 20px;
                top: 100px;
            }
            #jack {
                background: url(image/jack.jpg) no-repeat;
                z-index: 2;
                left: 100px;
                top: 100px;
            }
            #queen {
                background: url(image/queen.jpg) no-repeat;
                z-index: 3;
                left: 180px;
                top: 100px;
            }
            #king {
                background: url(image/king.jpg) no-repeat;
                z-index: 4;
                left: 260px;
                top: 100px;
            }
            #ace {
                background: url(image/ace.jpg) no-repeat;
                z-index: 5;
                left: 340px;
                top: 100px;
            }
        </style>
    </head>
    <body>
        <h3>CSS 属性 z-index 的应用</h3>
        <hr />
        <div id="ten"></div>
        <div id="jack"></div>
        <div id="queen"></div>
```

```
50.            <div id = "king"></div>
51.            <div id = "ace"></div>
52.        </body>
53.    </html>
```

运行效果如图 3-41 所示。

图 3-41 CSS 属性 z-index 的应用效果

【代码说明】

本示例包含了五个 元素用于测试层叠效果。首先为 元素设置统一样式：宽度为 182 像素、高度为 253 像素，并使用绝对定位模式。

为这五个 元素分别设置背景图片，图片素材来源于扑克牌红桃 10、J、Q、K、A 的牌面。背景图片为 jpg 格式并且均来源于与 HTML 文档同一目录下的 image 文件夹。使用方位关键词 left 和 top 定位每一个 元素：所有 元素均距离页面顶端 100 像素；距离页面左侧分别为 20 像素、100 像素、180 像素、260 像素和 340 像素，即每个 元素往右边平移 80 像素。

为达到层叠效果，为这五个 元素分别设置 z-index 属性值，从 1 开始到 5 结束。其中红桃 10 对应 z-index:1，因此会显示为叠放在最底层，以此类推每张牌都高一层，直到红桃 A 对应 z-index:5，会显示在最上面。因此最终实现了元素在页面上的层叠效果。

3.6.4 浮动

1. 浮动效果 float

在 CSS 中 float 属性可以用于令元素向左或向右浮动。以往常用于文字环绕图像效果，实际上任何元素都可以应用浮动效果。该属性有四种属性值，如表 3-37 所示。

表 3-37 CSS 属性 float 属性值一览表

属 性 值	解 释
left	元素向左浮动
right	元素向右浮动
none	float 属性的默认值，表示元素不浮动
inherit	继承父元素的 float 属性值

在对元素声明浮动效果后，该浮动元素会自动生成一个块级框，因此需要明确指定浮动元素的宽度，否则会被默认不占空间。元素在进行浮动时会朝着指定的方向一直移动，直到碰到页面的边缘或者上一个浮动框的边缘才会停下来。

如果一行之内的宽度不足以放置浮动元素，则该元素会向下移动直到有足够的空间为止再向着指定的方向进行浮动。

【例 3-41】 CSS 浮动的简单应用

使用 CSS 属性 float 制作文字环绕图片的效果。


视频讲解

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 浮动的应用</title>
        <style>
            div {
                float: left;
                width: 230px;
            }
            p {
                line-height: 30px;
                text-indent: 2em;
            }
        </style>
    </head>
    <body>
        <h3>CSS 浮动的应用</h3>
        <hr />
        <div><img src="image/balloon.jpg" alt="热气球" />
        </div>
        <p>
            18 世纪，法国造纸商孟格菲兄弟在欧洲发明了热气球。他们受碎纸屑在火炉中不断升起的启发，用纸袋把热气聚集起来做实验，使纸袋能够随着气流不断上升。1783 年 6 月 4 日，孟格菲兄弟在里昂安诺内广场做公开表演，一个圆周为 110 英尺的模拟气球升起，飘然飞行了 1.5 英里。同年 9 月 19 日，在巴黎凡尔赛宫前，孟格菲兄弟为国王、王后、宫廷大臣及 13 万巴黎市民进行了热气球的升空表演。同年 11 月 21 日下午，孟格菲兄弟又在巴黎穆埃特堡进行了世界上第一次热气球载人空中飞行，飞行了 25 分钟，飞越半个巴黎之后降落在意大利广场附近。这次飞行比莱特兄弟的飞机飞行早了整整 120 年。在充气气球方面，法国的罗伯特兄弟是最先乘充满氢气的气球飞上天空的。(摘自百度百科热气球词条)
        </p>
    </body>
</html>
```

运行效果如图 3-42 所示。

【代码说明】

本示例包含了区域元素 和段落元素<p>各一个，用于测试段落内容对于图片的环绕效果。区域元素 的 CSS 样式定义为宽 230 像素，并且内部嵌套了一个图像元素<img>，图片来源于本地 image 文件夹中的 balloon.jpg。段落元素<p>作了简单的文本修饰：首行缩进 2 个字符，并且段落行高为 30 像素。

在默认情况下，段落元素的文字内容会显示在图片的正下方。为 元素增加浮动效果：设置 float 属性值为 left，表示向左浮动。此时由图 3-42 可见，段落元素向上移动并

图 3-42 CSS 浮动的应用效果

补在了图片的右侧。从而实现了文字环绕图片的效果。

浮动也可以用于将多个元素排成一行，实现单行分列的效果。

【例 3-42】 CSS 浮动的应用 2

使用 CSS 属性 float 制作扑克牌排成一行展示的效果。


```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS 浮动的应用 2</title>
        <style>
            div {
                width: 182px;
                height: 253px;
                float: left;
            }
            #ten {
                background: url(image/ten.jpg) no-repeat
            }
            #jack {
                background: url(image/jack.jpg) no-repeat
            }
            #queen {
                background: url(image/queen.jpg) no-repeat
            }
            #king {
                background: url(image/king.jpg) no-repeat
            }
            #ace {
                background: url(image/ace.jpg) no-repeat
            }
        </style>
    </head>
```

```
29.        <body>
30.            <h3>CSS 浮动的应用 2</h3>
31.            <hr />
32.            <div id = "ten"></div>
33.            <div id = "jack"></div>
34.            <div id = "queen"></div>
35.            <div id = "king"></div>
36.            <div id = "ace"></div>
37.        </body>
38.    </html>
```

运行效果如图 3-43 所示。

图 3-43　CSS 浮动的应用效果

【代码说明】

本示例是例 3-40 的修改版，将显示为层叠效果的 5 张扑克牌改为在同一行展开。每张扑克牌仍然使用宽 182 像素、高 253 像素的 元素表示。

将这五个用于显示扑克牌面图片的 元素去掉原先的绝对定位与层叠属性 z-index，新增 float 属性值为 left，用于测试 CSS 属性 float 带来的浮动效果。由于 本身是块级元素会自动换行显示，因此本示例中如果没有添加浮动效果，则这些 元素会从上往下垂直排开。使用了 float 属性后， 元素会自动向左进行浮动，直到元素的左边外边缘碰到了页面顶端或前一个浮动框的边框时才会停止。

由图 3-43 可见，在页面足够宽的情况下，五张牌面可以在同一行进行展示。

注意：如果浏览器页面缩放尺寸，则有可能造成宽度过窄容纳不下全部的元素，这会导致其他 元素自动向下移动直到拥有足够的空间才能继续显示。

2. 清理浮动 clear

CSS 中的 clear 属性可以用于清理浮动效果，它可以规定元素的哪一侧不允许出现浮动元素。该属性有五种属性值，如表 3-38 所示。

表 3-38　CSS 属性 clear 属性值一览表

属　性　值	解　　释
left	元素的左侧不允许有浮动元素
right	元素的右侧不允许有浮动元素

续表

属 性 值	解 释
both	左右两侧均不允许有浮动元素
none	clear 属性的默认值，表示允许浮动元素出现在左右两侧
inherit	继承父元素的 clear 属性值

例如，常用 clear:both 来清除之前元素的浮动效果。

```
p{
    clear:both;
}
```

此时该元素不会随着之前的元素进行错误的浮动。

3.7 本章小结

CSS 通过样式表来设置页面样式，根据样式表的声明位置分为内联样式表、内部样式表和外部样式表，其中内联样式表的层叠优先级最高。在 CSS 样式表中可以使用选择器为指定元素设置样式，常用选择器包括元素选择器、ID 选择器、类选择器与属性选择器。

CSS 样式中可定义的取值包括数字、长度、角度、时间、文本及颜色等内容。CSS 常用于对 HTML 元素背景、边距、文本、字体、超链接、列表和表格的相关样式设置。CSS 还包含了四种定位 HTML 元素的方式，分别是绝对定位、相对定位、层叠定位与浮动。

习题 3

1. CSS 样式表有哪几种类型？它们的层叠优先级关系怎样？
2. 常用的 CSS 选择器有哪些？
3. CSS 的注释语句写法是怎样的？
4. CSS 颜色值有哪几种表达方式？
5. CSS 背景图像的平铺方式有哪几种？
6. 如何使用 CSS 为文本添加下画线？
7. 如何使用 CSS 为列表选项设置自定义标志图标？
8. 如何使用 CSS 实现表格为单线条框样式？
9. 如何使用 CSS 设置元素的层叠效果？
10. 元素可以向哪些方向进行浮动？如何清除浮动效果？