

第 3 章

CSS 网页样式基础

【目标任务】

学习目标	<ol style="list-style-type: none">1. 掌握 CSS 基本概念和作用2. 掌握 Dreamweaver 设置 CSS 样式的方法3. 掌握引入 CSS 样式表的方法4. 掌握 CSS 样式的语法规则5. 掌握 CSS 基础选择器的类型及优先级6. 掌握 CSS 文本样式的设置7. 掌握 CSS 复合选择器的应用8. 掌握 CSS 的继承特性9. 掌握 CSS 的优先级10. 掌握 CSS 的层叠性
重点知识	<ol style="list-style-type: none">1. 外部、内部和行内 CSS 样式表的运用2. CSS 基础选择器与复合选择器3. CSS 样式语法规则4. CSS 文本样式的设置5. CSS 的层叠特性6. CSS 的继承特性7. CSS 的优先级的判断方法
项目实战	项目 3-1 百度搜索结果网页局部样式设置 项目 3-2 端午节习俗新闻页面
实训作业	实训任务 3-1 Web 前端试学班广告页局部样式设置 实训任务 3-2 “商品推荐”页局部样式设置

【知识技能】

3.1 什么是 CSS 样式

什么是 CSS 样式呢？在本书第 2 章中曾经提到，直接给 `<body>`、`<p>`、``、`` 等 HTML 标签设置属性的方式使网页代码变得臃肿、维护不方便，并不符合 Web 标准的理念，在实际项目开发时，常用 CSS 的方式来格式化网页文档。

CSS 于 1994 年由哈坤·利(Hakon Wium Lie)提出,当时伯特·波斯(Bert Bos)正在设计一个名为 Argo 的浏览器,于是他们决定一起设计 CSS,以解决 HTML 为了满足页面设计者的需求,增加了多种显示功能属性,使得 HTML 网页代码变得越来越杂乱、越来越臃肿的难题。

CSS 是 Cascading Style Sheets 的缩写,译作层叠样式表。CSS 是一组样式设置规则,用于控制 Web 页面内容的外观,也是一种表示 HTML 或 XML 等文件样式的计算机语言,它的定义是由 W3C 来维护。

关于网页样式的代码通常存储在样式表文件中,与控制网页结构和内容的 HTML 代码分别存放在不同的位置,解决内容与表现分离的问题,符合 Web 标准的要求。

如图 3-1 所示,HTML 标签设置了网页的结构,<p> 标签内设置了网页的内容。应用内部 CSS 样式表控制页面的外观,如字体、颜色、对齐、文字间距和行间距、去掉链接下划线等文字样式;设置列表样式;设置图像阴影和透明度;设置段落或页面的背景颜色、背景图像;控制网页的布局等。

```
1 <!doctype html>
2 <html>
3 <head>
4 <meta charset="utf-8">
5 <title>无标题文档</title>
6 <style type="text/css">
7   p{
8     color:red;
9     text-align:center; ← 内嵌CSS样式代码
10  }
11 </style>
12 </head>
13
14 <body>
15   <p>段落的外观样式由CSS样式代码设置</p>
16 </body>
17 </html>
```

图 3-1 HTML 结构与 CSS 样式

CSS 样式的优点有以下几点。

1. 表达效果丰富,更易于控制页面外观

CSS 支持文字、图像的精确定位,三维层技术以及交互操作等,应用 CSS 样式可以更好地控制页面的外观,如字体、颜色、尺寸、调整文字间距和行间距、去掉链接下划线、设置列表样式、设置图像阴影和透明度,控制网页的布局等。

2. 内容和表现形式分离,代码简练、修改方便,更方便控制页面外观

如果说 HTML 是用于构建网页骨架,那么 CSS 就是为网页装修外观。CSS 可以将页面的内容和表现形式分离,页面内容即 HTML 代码放在 HTML 文件中,用于定义代码表现形式的 CSS 规则,放在另一个文件(外部样式表)或 HTML 文档的另一部分(通常放在文件头部分)。当需要给这些网页定义样式时,只要将样式文件链接至各个网页即可,而不必再把烦琐的样式编写在每个文档结构中,这样使得 HTML 文档的代码更为简练,缩短浏览器的加载时间,同时也方便修改和更新,只改变 CSS 文件中的某一行,整个站点的网页随之改变,无须对每个页面上的每个属性都进行更新。

3. 文档体积小,节约带宽,提高传输速度

相同标签的内容有相同的样式表现,使用传统的方法需要为每个标签分别定义显示格

式,造成大量重复定义,使网页代码臃肿。使用 CSS,对同一类标签只需进行一次格式定义即可,大大缩小文件体积,提高传输速度,节约带宽。

4. 便于信息检索

样式与内容、结构分离,显示细节的描述并不会影响文档中数据的内在结构,搜索引擎对文档进行检索时,更容易检索到有用信息。

5. 可读性好

CSS 文件内,对各种标签的显示集中定义,且定义方式直观易读,使得易学、易用、可读性、可维护性提高。而且,结构化的数据也相对简洁、清晰,突出对内容本身的描述。

6. 极大地提高了工作效率

样式通常保存在外部的 .css 文件中,站点中的所有页面的相同的样式,建立一个样式表,多个页面都可以链接此样式表。通过编辑一个简单的 CSS 文档,外部样式表可同时改变站点中所有页面的布局和外观,使得 HTML 文档的代码更为简练,缩短浏览器的加载时间。同一页面中,样式表可以控制所有相同属性的标签、类、ID,同样,通过修改 style 样式,控制整个页面的外观。

3.2 Dreamweaver CC 中创建样式表并设置 CSS 样式的方法

3.2.1 认识【CSS 样式】面板

在 Dreamweaver CC 中,可以通过【CSS 样式】面板来新建、删除、编辑和应用 CSS 样式,以及附加外部样式表等。单击【窗口】→【CSS 样式】(【Shift+F11】)菜单,打开【CSS 样式】面板,如图 3-2 所示。在【CSS 样式】面板中,可以直观地查看整个文档中定义的所有样式,也可以直接快速更改当前所选样式的属性。



图 3-2 【CSS 样式】面板

【CSS 样式】面板的【全部】选项卡包含两个窗格。其中,上半部分的【所有规则】窗格显示了当前文档中定义的样式和链接到当前文档的样式文件中定义的样式;下半部分的【属性】窗格可以快速编辑【所有规则】窗格中所选 CSS 样式的属性。

通过单击面板左下角的 3 个按钮,可控制属性的显示方式。其中,【类别】视图表示按类

别分组显示属性(如“字体”“背景”“区块”“边框”等),并且已设置的属性位于每个类别的顶部;【列表】视图表示按字母顺序显示属性,并且已设置的属性排在顶部。此外,在类别视图和列表视图模式下,已设置的属性将以蓝色显示。要修改属性,可直接单击选择属性,进行修改。

单击【当前】选项卡按钮,【CSS 样式】面板将显示 3 个窗格(如图 3-3 所示),上部是当前所选内容的 CSS 属性摘要,中部显示了所选属性位于哪个样式中,下部显示了 CSS 样式属性。



图 3-3 【CSS 样式】面板的【当前】选项卡

3.2.2 创建并编辑 CCS 样式的过程

在 Dreamweaver CC 中,可按照以下方法来创建 CSS 样式。

Step01. 在【CSS 样式】面板中单击【新建 CSS 规则】按钮,弹出【新建 CSS 规则】对话框,如图 3-4 所示。

Step02. 在【选择器类型】下拉列表中选择要创建的选择器类型,如图 3-5 所示。输入选择器名称,标签选择器不用定义选择器名称,在下拉列表中进行选择即可。在 Dreamweaver CC 中,【选择器类型】主要共有 4 种,具体内容将在本书后续章节详细讲解。

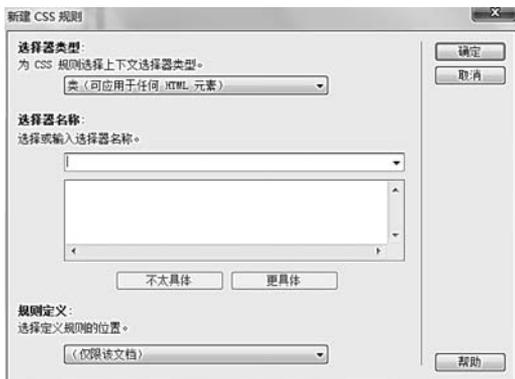


图 3-4 【新建 CSS 规则】对话框

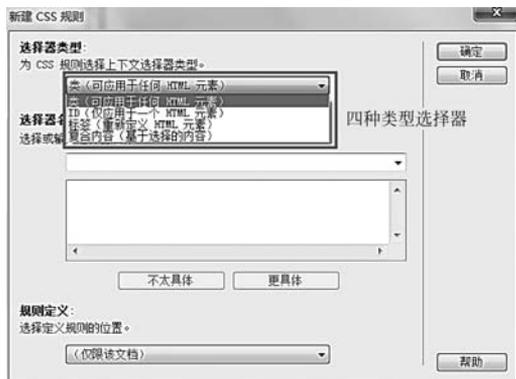


图 3-5 选择器类型

Step03. 在【规则定义】选项组中指定保存样式的位置,如图 3-6 所示。

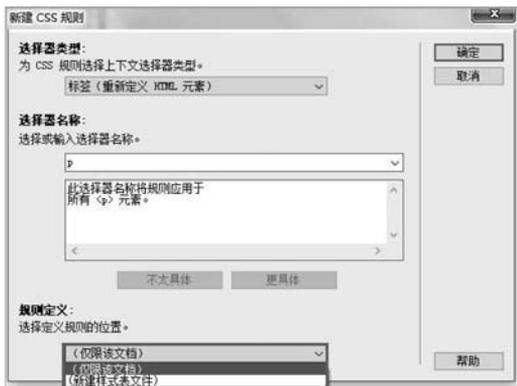


图 3-6 引入样式表的方式

选择“新建样式表文件”,弹出【将样式表另存为】对话框,如图 3-7 所示,创建一个扩展名为.css 的外部样式表文件。

在代码视图中会产生链接外部样式表的代码,如图 3-8 所示。

若在图 3-6 中选择“仅限该文档”,将新建样式保存在指定目录中,并在代码视图生成内部样式表代码,如图 3-9 所示。

Step04. 单击【确定】按钮,弹出【CSS 规则定义】对话框,如图 3-10 所示。在【body 的 CSS 规则定义】对话框左侧的【分类】列



图 3-7 创建外部样式表

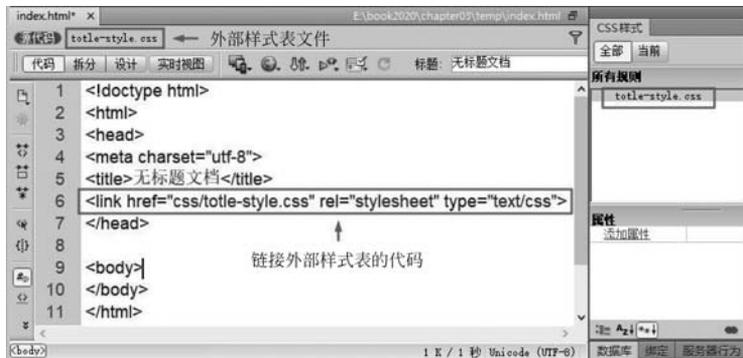


图 3-8 外部样式表

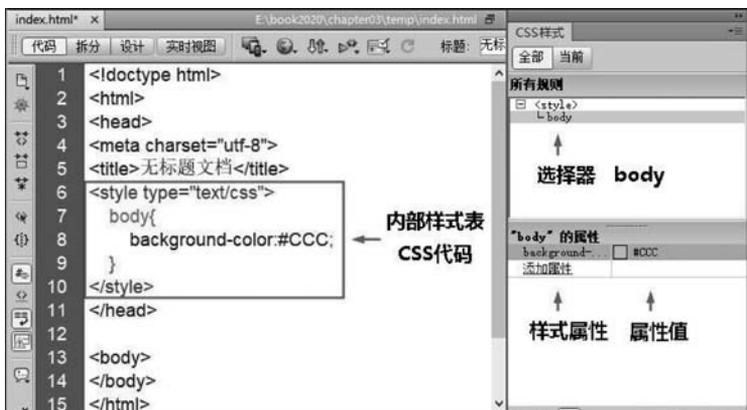


图 3-9 内部样式表

表区选择不同分类,可设置样式的不同属性。单击【确定】按钮。

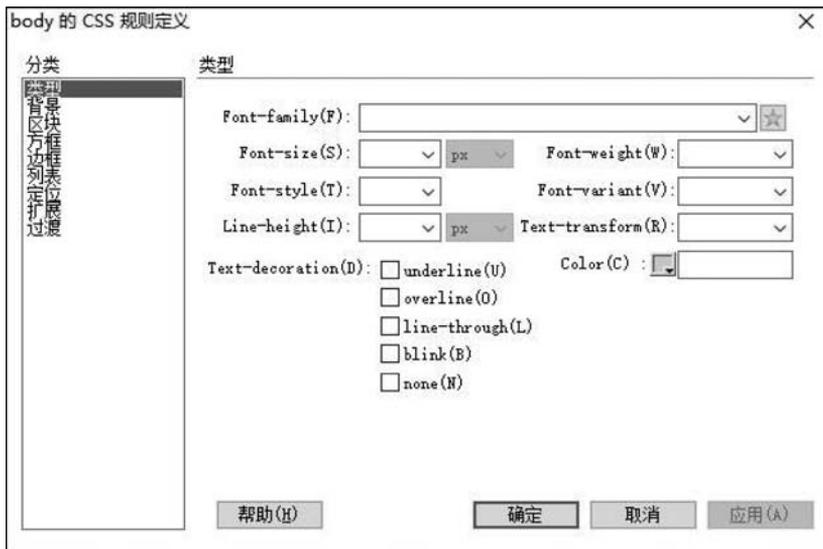


图 3-10 【CSS 规则定义】对话框

3.3 引入 CSS 样式表的方法

在 HTML 中,引入 CSS 样式有 3 种方式: 外部样式表、内部样式表和行内样式。

3.3.1 外部样式表

外部样式表是将所有的样式放在一个或多个以 .css 为扩展名的外部样式表文件中,在 HTML 文件的 <head> 标签中,通过 <link /> 标签将外部样式表文件(扩展名为 .css 的样式表文件)链接到 HTML 文档中,其基本语法格式如下。

```
<head>
<link href = "CSS 文件的路径" type = "text/css" rel = "stylesheet" />
</head>
```

该语法中,<link /> 标签需要放在 <head> 头部标签中,并且必须指定 <link /> 标签的 3 个属性,具体如下。

- href: 定义所链接外部样式表文件的 URL,可以是相对路径,也可以是绝对路径。
- type: 定义所链接文档的类型,这里需要指定为“text/css”,表示链接的外部文件为 CSS 样式表。
- rel: 定义当前文档与被链接文档之间的关系,这里需要指定为“stylesheet”,表示被链接的文档是一个样式表文件。

【例 3-1】 外部样式表的应用(案例文件: chapter03\example\exp3_1. html)
HTML 文档 exp3_1. html 代码如下。

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>外部样式表</title>
    <link href="css/style1.css" rel="stylesheet" type="text/css" />
  </head>
  <body>
    <h2>exp01 标题 2</h2>
    <p>exp01 段落</p>
  </body>
</html>
```

外部 CSS 样式表文件 style1.css 代码如下。

```
@charset "utf-8";
/* CSS Document */
h2{color: red; }
p{color: blue; }
```

第一行代码声明该 CSS 文件使用 utf-8 编码,当文件保存为 utf-8 编码时,其中的中文字体可以正确显示为中文,中文注释文字也能正常显示,否则中文注释显示为乱码。第二行代码是 CSS 样式表的注释,表明这是一个 CSS 样式。

同 HTML 的注释一样,注释起解释说明的作用,浏览器不会解释 CSS 注释,也不会被显示,只有用户打开 CSS 文件才会看见注释与注释内容。CSS 注释的开始使用 /*,结束使用 /*,基本语法格式如下。

```
/* 注释内容 */
```

示例代码如下。

```
h2{color: red; } /* 设置<h2>标签的文字颜色为红色 */
p{color: blue; } /* 设置<p>标签的文字颜色为蓝色 */
```

为了节省篇幅,在本书后续章节中省略这两行代码,只书写关键的 CSS 代码,完整的代码请看随书附带的 CSS 代码。

【例 3-2】 一个外部样式表被多个 HTML 文档调用(案例文件: chapter03\example\exp3_2.html)

外部样式表最大的优点是同一个 CSS 样式表可以被不同的 HTML 页面链接使用,如例 3-1 中 HTML 文档 exp3_1.html 和本例中的 HTML 文档 exp3_2.html 都调用了外部样式表 style1.css,省去了对每一个 HTML 网页都要进行样式设置的麻烦。

HTML 文档 exp3_2.html 代码如下。

```
<!DOCTYPE html>
<html lang = "en">
  <head>
    <meta charset = "UTF-8">
    <title>外部样式表</title>
    <link href = "css/style1.css" rel = "stylesheet" type = "text/css" />
  </head>
  <body>
    <h2>exp02 标题 2</h2>
    <p>exp02 段落</p>
  </body>
</html>
```

【例 3-3】 一个 HTML 文档调用多个外部样式表(案例文件: chapter03\example\exp3_3.html)

一个 HTML 页面也可以通过多个<link />标签链接多个 CSS 外部样式表。
HTML 文档 exp3_3.html 代码如下。

```
<!DOCTYPE html>
<html lang = "en">
  <head>
    <meta charset = "UTF-8">
    <title>外部样式表</title>
    <link href = "css/style1.css" rel = "stylesheet" type = "text/css" />
    <link href = "css/style2.css" rel = "stylesheet" type = "text/css" />
  </head>
  <body>
    <h2>exp03 标题 2</h2>
    <p>exp03 段落</p>
  </body>
</html>
```

外部 CSS 样式表 style1.css 代码如下。

```
h2{color: red; } /* 设置<h2>标签的文字颜色为红色 */
p{color: blue; } /* 设置<p>标签的文字颜色为蓝色 */
```

外部 CSS 样式表 style2.css 代码如下。

```
h2{font-size: 48px} /* 设置<h2>标签的文字字号为 48 像素 */
p{font-size: 24px; } /* 设置<p>标签的文字字号为 24 像素 */
```

两个外部样式表共同控制 HTML 标签的样式,<h2>标签的文字颜色为红色,字号为 48 像素,<p>标签的文字颜色为蓝色,字号为 24 像素。

3.3.2 内部样式表

内部样式表是将 CSS 代码集中写在 HTML 文档的<head>头部标签中,并且用<style>标签定义,其基本语法格式如下。

```
<head>
  <title>标题</title>
  <style type = "text/css" >
    选择器 {属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; ... }
  </style>
</head>
```

<style>标签一般位于<head>标签中<title>标签之后。<style>标签的 type 属性是必需的,取值是唯一的,取值为“text/css”。表示 style 标签对之间的文本内容要作为层叠样式表(css)来解析。

【例 3-4】 内部样式表的应用(案例文件:chapter03\example\exp3_4.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>内部样式表</title>
    <style type = "text/css" >
      h2{text-align: center; }
      p{
        font-size: 16px;
        color: red;
      }
    </style>
  </head>
  <body>
    <h2>标题 2</h2>
    <p>段落</p>
  </body>
</html>
```

内部样式只对其所在的 HTML 页面有效,因此,当单个文档需要特殊的样式时,使用内嵌式是个不错的选择。如果是一个网站,内部样式表不能充分发挥 CSS 代码的重用优势。有关网站统一样式建议使用外部样式表,可被多个网页调用;有关当前网页自身独特的样式,可以使用内部样式表。

3.3.3 行内样式

行内样式又称内联样式,是通过 HTML 标签的 style 属性来控制标签的样式,任何 HTML 标签都拥有 style 属性,用来设置行内式。其基本语法格式如下。

```
<标签名 style="属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; ... ">
    内容
</标签名>
```

【例 3-5】 行内样式表的应用(案例文件: chapter03\example\exp3_5.html)

```
<!DOCTYPE html>
<html lang="en">
  <head>
    <meta charset="UTF-8">
    <title>行内样式</title>
  </head>
  <body>
    <h2>标题 2</h2>
    <p>段落 1</p>
    <p style="color:red;">段落 2, 只有段落 2 的文字变为<span>红色</span> </p>
    <p>段落 3</p>
  </body>
</html>
```

在浏览器中运行,可以看到,只有段落 2 的文字颜色变为红色,可见,行内样式只对其所在的标签及嵌套在其中的子标签起作用。行内样式以标签属性的方式存在,并没有做到结构与表现的分离,代码臃肿,一般很少使用。只有在样式规则较少且只在该元素上使用一次,或者临时修改某个样式规则时使用。

3.3.4 CSS 样式表的混合使用

行内样式表、内部样式表、外部样式表各有优势,实际项目开发中常常需要混合使用。有关整个网站统一风格的样式代码,放置在独立的外部样式文件 *.css 中,可以被各个 HTML 文档调用。某些样式独特的页面,除了链接外部样式文件,还需定义内部样式表,只控制当前页面的样式。某张网页内,需要临时修改样式规则,可暂时采用行内样式。

对于某个 HTML 标签,如果既设置了外部样式,又设置了内部样式和行内样式,如果规定的样式没有冲突,则样式叠加;如果有冲突,则依据“最近优先”原则,如图 3-11 所示。最先考虑行内样式表显示,再考虑内部样式显示,最后采用外部样式显

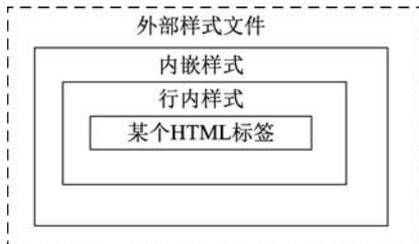


图 3-11 “最近优先”原则

示。最先考虑行内样式表显示,再考虑内部样式显示,最后采用外部样式显

示,没有设置的按照 HTML 的默认样式显示。

【例 3-6】 CSS 样式表的混合应用(案例文件:chapter03\example\exp3_6.html)
HTML 文档代码如下。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> 样式表的优先级 </title>
    <link href = "css/style1.css" rel = "stylesheet" type = "text/css" />
    <link href = "css/style2.css" rel = "stylesheet" type = "text/css" />
    <style type = "text/css" >
      h2{text-align: center; }
      p{
        font-size: 16px;
        color: green;
      }
    </style>
  </head>
  <body>
    <h2> 标题 2 </h2>
    <p> 段落 1 </p>
    <p style = "color: red; ">段落 2, 只有段落 2 的文字变为
<span>红色 </span>
  </p>
    <p> 段落 3 </p>
  </body>
</html>
```

外部 CSS 样式表 style1.css 代码如下。

```
h2{color: red; } /* 设置<h2>标签的文字颜色为红色 */
p{color: blue; } /* 设置<p>标签的文字颜色为蓝色 */
```

外部 CSS 样式表 style2.css 代码如下。

```
h2{font-size: 48px} /* 设置<h2>标签的文字字号为 48 像素 */
p{font-size: 24px; } /* 设置<p>标签的文字字号为 24 像素 */
```

对于标题 2, style1.css、style2.css 和内部样式表中设置的样式没有冲突,所以,样式叠加显示;对于段落 1 和段落 3,外部样式表和内部样式表中都设置了 color 和 font-size,产生冲突,按照内部样式表设置的样式显示;对于段落 2,font-size 按照内部样式表的设置显示,color 按照行内样式表的设置显示。

3.4 CSS 样式的语法规则

使用 HTML 时,需要遵从一定的规范。CSS 亦是如此,要想熟练地使用 CSS 对网页进行修饰,首先需要了解 CSS 样式规则。CSS 样式具体格式如下。

```
选择器{属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; }
```

在以上的样式规则中,选择器用于指定 CSS 样式作用的 HTML 对象,花括号内是对该对象设置的具体样式。其中,属性和属性值以“键值对”的形式出现,称为一个“声明”,属性是对指定的对象设置的样式属性,如字体大小、文本颜色等。属性和属性值之间用英文“:”连接,多个“键值对”之间用英文“;”进行区分。

以下代码的作用是将 p 元素内的文字颜色定义为红色,同时将字体大小设置为 20 像素。p 是选择器,color 和 font-size 是属性,red 和 20px 是属性值。

```
p {color: red; font-size: 20px; }
```

以上代码的结构如图 3-12 所示。



图 3-12 CSS 语法规则

书写 CSS 样式时,除了要遵循 CSS 样式规则,还必须注意以下几个问题。

1. 选择器区分大小写

CSS 样式中的选择器,除标签选择器外,其他选择器严格区分大小写,标签选择器最好使用小写。

2. 属性和值不区分大小写

属性和值不区分大小写,按照书写习惯,最好采用小写方式。

3. 属性名值对分号隔开

多个声明之间必须用英文状态下的分号隔开,最后一个属性后的分号可以省略。有经验的网页设计人员会在每条声明的末尾都加上分号,从现有的规则中增加声明时,尽可能地减少出错的可能性。示例代码如下。

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

4. 属性值中的双引号

如果属性的值由多个单词组成且中间包含空格,则必须为这个属性值加上英文状态下的双引号。示例代码如下。

```
p{font-family: "Times New Roman"; }
```

5. CSS 注释

在编写 CSS 代码时,为了提高代码的可读性,通常会加上 CSS 注释。注释是用以解释代码,便于开发人员查看,浏览器不会解析它。CSS 注释以“/*”开始,以“*/”结束。示例代码如下。

```
p {
    text-align: center; /* 文本居中对齐 */
    color: black;      /* 文字颜色为黑色 */
    font-size: 24px;   /* 文字字号为 24 像素 */
}
```

6. CSS 代码缩进

CSS 代码的排版最好在每行只描述一个属性,并且注意缩进,这样可以增强样式定义的可读性,建议初学者形成良好的 CSS 书写规范。

3.5 CSS 样式基础选择器

若将 CSS 样式应用于特定的 HTML 元素,需要找到该目标元素。在 CSS 中,执行这一任务的样式规则部分被称为选择器。

3.5.1 标签选择器

标签选择器是指用 HTML 标签名称作为选择器,按标签名称分类,为页面中某一类标签指定统一的 CSS 样式。基本语法格式如下。

```
标签名{属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; ... }
```

所有 HTML 标签名都可以作为标签选择器,例如 body、h1~h6、p、span、a、img 等 HTML 标签。用标签选择器定义的样式对页面中该类型的所有标签都有效。在创建或更改一个 HTML 标签的 CSS 样式后,所有使用该标签的文本的样式将得到更新。示例代码如下。

```
p{ font-size: 12px; color: #666; font-family: "宋体"; }
```

以上 CSS 样式代码用于设置 HTML 页面中所有的段落文本的字号大小为 12 像素、颜色为 #666、字体为宋体。

【例 3-7】 标签选择器(案例文件: chapter03\example\exp3_7.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> 标签选择器 </title>
    <style type = "text/css" >
      h1{color: blue; }
      p{color: #333; }
```

```
    </style>
</head>
<body>
  <h1>第 1 个标题</h1>
  <p>第 1 个段落</p>
  <p>第 2 个段落</p>
  <h1>第 2 个标题</h1>
  <p>第 3 个段落</p>
  <p>第 4 个段落</p>
</body>
</html>
```

为标签<h1>设置文本颜色为蓝色,所有的<h1>标签内的文本都变成蓝色;为标签<p>设置文本颜色为#333,所有的段落文本颜色都是#333。

标签选择器最大的优点是能快速为页面中同类型的标签统一样式,但这也是它的缺点,不能设计差异化样式。如果使用标签选择器定义了<p>标签的样式,则网页中所有<p>标签的内容都变成定义的样式。若有一个<p>标签内容要求不按照预先定义的标签样式显示,就需要用到类选择器。

3.5.2 类选择器

类选择器又称自定义 CSS 样式,类选择器主要用于定义一些特殊的样式,可以多次被相同的标签调用,也可以被多种不同的标签调用,一个标签还可以同时调用多个类。类选择器名称必须以英文的句点(.)开头,类名使用字母、数字、下划线,第一个字符不能是数字,严格区分大小写。基本语法格式如下。

```
.类名{属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; ... }
```

类名即为 HTML 元素的 class 属性值,大多数 HTML 元素都可以定义 class 属性。在 HTML 文档中,通过标签的 class 属性调用类。注意,在 HTML 标签中调用类时,不用加英文句点。基本语法格式如下。

```
<开始标签 class = "类名"> 内容... </结束标签>
<开始标签 class = "类名 1 类名 2"> 内容... </结束标签>
```

【例 3-8】 类选择器(案例文件: chapter03\example\exp3_8.html)

```
<!DOCTYPE html>
<html lang = "en">
  <head>
    <meta charset = "UTF-8">
    <title>类选择器</title>
    <style type = "text/css">
      h1{color: blue; }
```

```

        p{color: #333; }
        .special_1{color: red; }
        .special_2{font-size: 24px; }
    </style>
</head>
<body>
    <h1 class = "special_1">第 1 个标题 </h1>
    <p>第 1 个段落 </p>
    <p class = "special_1">第 2 个段落 </p>
    <p class = "special_1">第 3 个段落 </p>
    <p class = "special_1 special_2">第 4 个段落 </p>
    <h1>第 2 个标题 </h1>
    <p>第 5 个段落 </p>
    <p>第 6 个段落 </p>
</body>
</html>

```

定义类名为 .special_1 和 .special_2 的类选择器并设置了 CSS 样式,.special_1 多次被 <p> 标签调用,也被多种标签(<h1> 和 <p> 标签)调用,第四个段落 <p> 标签同时调用了两个类,两个类名之间要用空格隔开。以下写法是错误的。

```
<p class = "special_1" class = "special_2">内容 ... </p>
```

3.5.3 ID 选择器

ID 选择器的使用方法和类选择器基本相同,不同之处在于网页中的 ID 值是唯一的,同一个页面不能出现相同的 ID 名。因此,ID 选择器只能在 HTML 中使用一次,只用来对单一元素定义单独的样式。通常情况下,对页面中比较唯一、固定且不会在同一个页面内重复出现的对象使用 ID 来标志,如布局中的 logo、导航、主体包含块、版权等。基本语法格式如下。

```
# id 名 { 属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; ... }
```

ID 选择器使用“#”进行标识,后面紧跟 id 名。id 名即为 HTML 元素的 id 属性值,大多数 HTML 元素都可以定义 id 属性,id 的命名使用字母、数字、下画线,第一个字符不能是数字,严格区分大小写。

在 HTML 文档中,通过标签的 id 属性调用 id 选择器,注意,在 HTML 标签中调用类时,不用加 #。基本语法格式如下。

```
<开始标签 id = "id 名">内容 ... </结束标签 >
```

【例 3-9】 ID 选择器(案例文件: chapter03\example\exp3_9.html)

```

<!DOCTYPE html>
<html lang = "en" >
    <head>
        <meta charset = "UTF-8" >

```

```

<title> ID 选择器</title>
<style type = "text/css">
    p {font-size: 20px;
        color: black;
        text-align: center;
    }
    div{width: 960px; border: 1px solid red; }/* 设置所有 div 标签的宽度和边框 */
    #nav{height: 50px; }/* 设置高度 */
    #banner{height: 80px; }
    #main{height: 200px; }
    #footer{height: 50px; }
</style>
</head>
<body>
    <div id = "nav"> <p> 导航区 </p> </div>
    <div id = "banner"> <p> banner 区 </p> </div>
    <div id = "main"> <p> 主内容区 </p> </div>
    <div id = "footer"> <p> 页脚区 </p> </div>
</body>
</html>

```

例 3-9 的运行效果如图 3-13 所示。



图 3-13 ID 选择器

不同于类选择器的多次调用，一个标签应用多个 id 是错误的。以下写法是错误的。

```
<p id = "logo nav">这种用法是错误的</p>
```

注意，在很多浏览器中，同一个 id 应用于多个标签，浏览器并不报错，但这种做法是不合理的。id 不能多次使用，而 class 类选择器却可以重复使用，因为 id 的优先级高于 class，所以 id 应该按需使用，而不能滥用。

3.5.4 通配符选择器

通配符选择器使用“*”号表示，它是所有选择器中作用范围最为广泛的，能匹配页面中

所有的元素。通配符选择器经常用于 CSS 样式重置(reset),清除浏览器的默认样式。基本语法格式如下。

```
* {属性 1: 属性值 1; 属性 2: 属性值 2; 属性 3: 属性值 3; }
```

使用通配符选择器定义 CSS 样式,清除所有 HTML 标签的默认边距。示例代码如下。

```
* {
    margin: 0;           /* 定义所有标签的外边距 */
    padding: 0;        /* 定义所有标签的内边距 */
}
```

通配符选择器在 CSS 中的优先级是最低的。实际项目开发中,不建议使用通配符选择器,因为它设置的样式对所有的 HTML 标签都生效,这样反而降低了代码的执行速度,比较耗费资源。常连接一个外部的 reset.css 文件,针对不同的标签进行相关的清零设置,请参考本书附带的文件 chapter03\example\css\reset.css。

3.6 常用 CSS 文本样式

学习 HTML 时,可以使用标签的属性简单控制文本的显示样式,但是这种方式烦琐且不利于代码的共享和移植,不符合 Web 标准的要求,因此,CSS 提供了相应的文本样式属性,使用 CSS 可以更轻松方便地控制文本样式。常用的文本样式属性有以下几种。

3.6.1 font-style(字体风格)

font-style 属性用于定义字体风格,如设置斜体、倾斜或正常字体,可用属性值如表 3-1 所示。

表 3-1 font-style 属性

属 性 值	描 述
normal	默认值,浏览器会显示标准的字体样式
italic	浏览器会显示文字斜体的字体样式
oblique	浏览器会将文字倾斜,包括无斜体属性的文字

【例 3-10】 font-style(案例文件: chapter03\example\exp3_10.html)

```
<!DOCTYPE html>
<html lang = "en">
  <head>
    <meta charset = "UTF-8">
    <title>font-style</title>
    <style type = "text/css">
```

```

        p {font-size: 20px; }
        .it{font-style: italic; }
        .ob{font-style: oblique; }
        .normal{font-style: normal; }
    </style>
</head>
<body>
    <p>p 标签的默认值</p>
    <p class = "it"> italic 样式</p>
    <p class = "ob"> oblique 样式</p>
    <i>i 标签的默认值</i>
    <br> <br>
    <i class = "normal">利用"font-style: normal"取消 i 标签的默认斜体样式
</i>
</body>
</html>

```

例 3-10 的运行效果如图 3-14 所示。



图 3-14 font-style 效果图

由图 3-14 可见,italic 与 oblique 显示效果几乎一样,实际项目开发中,常用 italic。使用 CSS 的 font-style: normal 可以取消 <i> 标签的斜体样式。

3.6.2 font-variant(变体)

font-variant 属性一般用于使英文字符表现为小型大写字母,仅对英文字符有效。可用属性值如表 3-2 所示。

表 3-2 font-variant 属性

属性值	描述
normal	默认值,浏览器会显示标准的字体
small-caps	浏览器会显示小型大写的字体,即所有的小写字母均会转换为大写。但是所有使用小型大写字体的字母与其余文本相比,其字体尺寸更小

【例 3-11】 font-variant(案例文件: chapter03\example\exp3_11.html)

```

<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> font-variant</title>
    <style type = "text/css" >
      .var{font-variant: small-caps; }
    </style>
  </head>
  <body>
    <p>小写 www.http://hsnc.edu.com</p>
    <p class = "var" >小写变小型大写 www.http://hsnc.edu.com</p>
    <p>大写 WWW.HTTP: //HSNC.EDU.COM</p>
  </body>
</html>

```

例 3-11 的运行效果如图 3-15 所示。



图 3-15 font-variant 效果图

由图 3-15 可见,“font-variant: small-caps”的效果大小跟小写字母一样,但样式是大写的。

3.6.3 font-weight(字体粗细)

font-weight 属性用于设置文本的粗细,可用属性值如表 3-3 所示。

表 3-3 font-weight 属性

属 性 值	描 述
normal	默认值。定义标准的字符
bold	定义粗体字符
bolder	定义更粗的字符
lighter	定义更细的字符
100~900(100 的整数倍)	定义由细到粗的字符。其中 400 等同于 normal,700 等同于 bold,值越大字体越粗

【例 3-12】 font-weight(案例文件: chapter03\example\exp3_12.html)

```
<!DOCTYPE html>
<html lang = "en" >
<head>
  <meta charset = "UTF-8" >
  <title>font-weight</title>
  <style type = "text/css" >
    p{font-size: 24px; }
  </style>
</head>
<body>
  <p>默认值: 文字粗细 </p>
  <p style = "font-weight: 400; ">400: 文字粗细 </p>
  <p style = "font-weight: bold; ">bold: 文字粗细 </p>
  <p style = "font-weight: 700; ">700: 文字粗细 </p>
  <p style = "font-weight: 900; ">900: 文字粗细 </p>
  <p style = "font-weight: bolder; ">bolder: 文字粗细 </p>
  <p style = "font-weight: lighter; ">lighter: 文字粗细 </p>
</body>
</html>
```

例 3-12 的运行效果如图 3-16 所示。



图 3-16 font-weight 效果图

由图 3-16 可见,数字值 400 相当于关键字 normal,也就是默认的样式,数值 700 相当于关键字 bold,即加粗样式。

3.6.4 font-size(字号大小)

font-size 属性用于设置字号大小,该属性的值可以使用相对长度单位,也可以使用绝对长度单位,具体如表 3-4 所示。

表 3-4 font-size 属性

相对长度单位	描 述
em	相对于父元素文本字号的倍数,默认 1em=16px
rem	相对于 HTML 根元素文本字号的倍数,CSS3 新增
px	像素
绝对长度单位	描 述
in	英寸
cm	厘米
mm	毫米
pt	点

相对长度单位比较常用,绝对长度单位较少使用。px 表示像素,各种浏览器的默认字号都是 16px,默认 1em=16px。rem 是 CSS3 新增的字号单位,其相对的是 Html 根元素。font-size 也可以设置为基于父元素的百分比值。

【例 3-13】 font-size(案例文件:chapter03\example\exp3_13.html)

```
<!DOCTYPE html>
<html>
  <head>
    <meta charset = "UTF-8" >
    <title>font-size</title>
    <style type = "text/css" >
      p{font-size: 24px; }
      span{font-size: 2em; }
    </style>
  </head>
  <body>
    <p>p 标签设置的文本字号 <span> span 标签内的文本 </span> </p>
  </body>
</html>
```

例 3-13 的运行效果如图 3-17 所示。



图 3-17 font-size 效果图

由图 3-17 可见, 标签设置字号大小是 2em, 是相对于其父元素 <p> 标签字号大小来计算的。

3.6.5 line-height (行高)

line-height 称为行高, line-height 属性用于设置行与行之间的距离, 即字符的垂直间距, 行间距、行高与文字字号之间的关系如图 3-18 所示。line-height 常用单位有 3 种, 分别为像素 px、相对值 em 和百分比 %。初学者使用较多的是像素 px, em 是相对于当前文字字号大小来计算的。



图 3-18 行间距、行高与文字字号之间的关系

line-height 与 font-size 的计算值之差(行距)分为两半, 分别加到一个文本行内容的顶部和底部。因此, 把 line-height 设置为与包含框相同的大小可以实现单行文字的垂直居中。

【例 3-14】 line-height(案例文件: chapter03\example\exp3_14.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>line-height</title>
    <style type = "text/css" >
      p{font-size: 24px; }
    </style>
  </head>
  <body>
    <p>第 1 段: Web 标准不是某一个标准, 而是一系列标准的集合. 网页主要由三部分组成: 结构、表现和行为。</p>
    <p style = "line-height: 48px; ">第 2 段: Web 标准不是某一个标准, 而是一系列标准的集合. 网页主要由三部分组成: 结构、表现和行为。</p>
```

```

    <p style = "line-height: 2em; ">第 3 段: Web 标准不是某一个标准, 而是一系列标准的集
    合. 网页主要由三部分组成: 结构、表现和行为。</p>
    </body>
</html>

```

例 3-14 的运行效果如图 3-19 所示。



图 3-19 line-height 效果图

“line-height: 48px;”与“line-height: 2em;”效果相同。line-height 属性值不能使用负值,为了网页的效果美观,line-height 的值是 font-size 值的 1.2~1.5 倍。

3.6.6 font-family(字体)

font-family 属性用于设置元素的字体,前提条件是用户计算机必须安装相应的字体,才能正确显示。font-family 可以同时指定多个字体,中间以逗号隔开,如果用户浏览器不支持第一个字体,则会尝试下一个字体,若所有字体用户浏览器都不支持,则采用浏览器默认的字体显示,示例代码如下。

```

p{ font-family: "宋体"; }
h1{font-family: "黑体","微软雅黑"; }

```

以上的代码将网页中所有段落文本的字体设置为宋体,一级标题首选黑体,如果用户计算机中没有安装该字体则选择微软雅黑,当指定的字体都没有安装时,则使用浏览器默认字体。

使用 font-family 设置字体时,需要注意以下几点。

- 各种字体之间必须使用英文状态下的逗号隔开。
- 中文字体需要加英文状态下的引号,英文字体一般不需要加引号。当需要设置英文字体时,英文字体名必须位于中文字体名之前。示例代码如下。

```

h1{font-family: "宋体"; }
p{font-family: Arial, "宋体"; }

```

- 如果字体名中包含空格、#、\$ 等符号,则该字体必须加英文状态下的单引号或双引号,示例代码如下。

```
p{font-family: "Times New Roman"; }
```

- 尽量使用系统默认字体,保证在任何用户的浏览器中都能正确显示。

3.6.7 font(综合设置字体样式)

font 属性属于复合属性,在一个声明中设置 3.6.1~3.6.6 节中所有字体属性,基本语法格式如下。

```
选择器{font: font-style font-variant font-weight font-size/line-height font-family; }
```

使用 font 属性时,必须按以上语法格式中的顺序书写,各个属性以空格隔开,font-size 与 line-height 之间采用“/”分割。示例代码如下。

```
p{ font: italic small-caps bold 18px/45px "微软雅黑","宋体" ; }
```

以上代码等价于以下代码。

```
p{
  font-style: italic;
  font-variant: small-caps;
  font-weight: bold;
  font-size: 18px;
  line-height: 45px;
  font-family: "微软雅黑","宋体";
}
```

其中,font-size 和 font-family 的值是必须的,其他不需要设置的属性可以省略(省略则取默认值)。例如,下面代码中表示为 h1 标签设置加粗、18 像素字号、字体为黑体。

```
h1{ font: bold 18px "黑体"; }
```

3.6.8 color(文本颜色)

color 属性用于设置文本的颜色,取值方式有如下几种。

1. 预定义颜色值

英文单词 pink, red, green, blue 等。

```
p{ color: red; }
```

2. 十六进制的颜色值

十六进制的颜色值以“#”开头的 6 位十六进制数值表示一种颜色。6 位数字分为 3 组,每组两位,依次表示红、绿、蓝 3 种颜色的强度。实际工作中,十六进制是最为常用的方式。

```
p{ color: #ff0000; }
```

3. 十六进制颜色的缩写

当十六进制的颜色值的 3 组数中两两相等时,可以进行缩写。示例如下。

#ff0000 缩写为 #f00

#cc3366 缩写为 #c36

4. RGB 颜色值

RGB 颜色值的语法是 `rgb(red, green, blue)`。每个参数 (`red`、`green` 以及 `blue`) 定义颜色的强度,取值是介于 0~255 的整数,或者是百分比值(0%~100%)。注意,即使百分比值为 0,百分号也不能省略。

```
p{ color: rgb(255,0,0); }
p{ color: rgb(100%,0%,0%); }
```

一些典型颜色对应关系如表 3-5 所示。

表 3-5 典型颜色对应关系

预定义颜色	十六进制	十六进制缩写	RGB
red	#ff0000	#f00	rgb(255,0,0)
green	#00ff00	#0f0	rgb(0,255,0)
blue	#0000ff	#00f	rgb(0,0,255)
white	#ffffff	#fff	rgb(255,255,255)
black	#000000	#000	rgb(0,0,0)

5. RGBA 透明色

RGBA 颜色值是 RGB 颜色值的扩展,带有一个 alpha 通道,它规定了颜色的不透明度。RGBA 颜色值的语法如下。

```
rgba(red, green, blue, alpha)
```

alpha 参数是介于 0.0(完全透明)至 1.0(完全不透明)的数字。`red`、`green`、`blue` 取值是 0~255 的整数。RGBA 透明色需要浏览器版本为 IE 9+、Firefox 3+、Chrome、Safari 以及 Opera 10+。示例代码如下。

```
p{ color: rgba(100%,0%,0%,.5); }
p{ color: rgba(255,0,0,.5); }
```

3.6.9 text-decoration(文本装饰)

`text-decoration` 属性用于设置文本的装饰效果,如下划线、上画线和删除线等装饰效果,常使用 `text-decoration: none` 于取消超链接默认的下划线样式。可用属性值如表 3-6 所示。

表 3-6 text-decoration 属性

属 性 值	描 述
none	默认值,没有装饰(正常文本)
underline	下画线
overline	上画线
line-through	删除线

【例 3-15】 text-decoration(案例文件: chapter03\example\exp3_15.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>text-decoration</title>
    <style type = "text/css" >
      p{font-size: 20px; }
    </style>
  </head>
  <body>
    <span>正常</span>
    <span style = "text-decoration: underline; ">下画线</span>
    <span style = "text-decoration: overline; ">上画线</span>
    <a href = "#">默认超链接</a>
    <a href = "#" style = "text-decoration: none; ">去掉超链接下画线</a>
  </body>
</html>
```

例 3-15 的运行效果如图 3-20 所示。

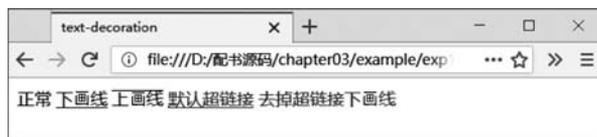


图 3-20 text-decoration 效果图

3.6.10 text-align(水平对齐方式)

text-align 属性用于设置文本内容的水平对齐方式,相当于 HTML 标签属性中的 align 对齐属性,可用属性值如表 3-7 所示。

表 3-7 text-align 属性值

属 性 值	描 述
left	默认值,左对齐
right	右对齐

属性值	描述
center	居中对齐
justify	两端对齐

【例 3-16】 text-align(案例文件：chapter03\example\exp3_16.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>text-align</title>
    <style type = "text/css" >
      h1{text-align: center; }
      h3{text-align: right; }
      p{text-align: right; }
      span{text-align: center; }
    </style>
  </head>
  <body>
    <h1>h1 标签居中对齐</h1>
    <h2>h2 标签默认左对齐</h2>
    <h3>h3 标签右对齐</h3>
    <p>p 标签段落右对齐<span>span 标签</span> </p>
  </body>
</html>
```

例 3-16 的运行效果如图 3-21 所示。



图 3-21 text-align 效果图

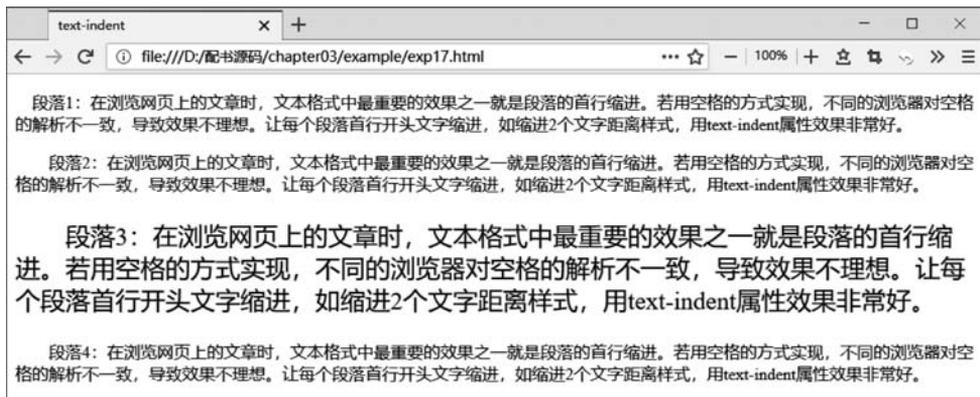


图 3-22 text-indent 效果图

当前文本的字号乘 2。段落 4 采用 em 做单位, 缩进值是当前文本字号的 2 倍。

3.7 CSS 控制列表样式

定义无序或有序列表时, 可以通过 HTML 标签的属性控制列表的项目符号, 但是这种方式并不符合 Web 标准的理念。CSS 中有一系列的列表样式属性, 可以控制列表的项目符号等样式。

list-style-type 属性可以设置无序和有序列表的项目符号或编号, list-style-type 属性值如表 3-8 所示, 基本语法格式如下。

```
选择器{list-style-type: 属性值; }
```

表 3-8 list-style-type 属性值

	属 性 值	描 述
无序列表(ul)	none	不显示任何符号
	disc	显示“●”
	circle	显示“○”
	square	显示“■”
有序列表(ol)	none	不显示任何编号
	decimal	阿拉伯数字 1,2,3,...
	upper-alpha	大写英文字母 A,B,C,...
	lower-alpha	小写英文字母 a,b,c,...
	upper-roman	大写罗马数字 I, II, III, ...
	lower-roman	小写罗马数字 i, ii, iii, ...

list-style-image 属性可以为各个列表项设置项目图像, 属性值通过 url 的方式指定列表项目的图像路径, 基本语法格式如下。

```
选择器{list-style-image: url(图像路径); }
```

list-style-position 属性可以控制列表项目符号的位置,属性值默认为 outside,表示列表项目符号位于列表文本以外。属性值还可以设置为 inside,表示列表项目符号位于列表文本以内,代码如下。

```
选择器{list-style-position: inside; }
```

【例 3-18】 CSS 控制列表样式(案例文件: chapter03\example\exp3_18.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 控制列表样式 </title>
    <style type = "text/css" >
      li{border: 1px solid red; }
      # test1{list-style-type: circle; }
      # test2{list-style-image: url(images/heart.png); }
      # test3{list-style-position: inside; }
    </style>
  </head>
  <body>
    <ul id = "test1" >
      <li>首页</li>
      <li>HTML</li>
      <li>CSS</li>
    </ul>
    <ul id = "test2" >
      <li>首页</li>
      <li>HTML</li>
      <li>CSS</li>
    </ul>
    <ul id = "test3" >
      <li>首页</li>
      <li>HTML</li>
      <li>CSS</li>
    </ul>
  </body>
</html>
```

list-style 是一个复合属性,可以综合设置列表的 list-style-type、list-style-position 和 list-style-image 属性,基本语法格式如下。

```
list-style: list-style-type list-style-position list-style-image
```

示例代码如下。

```
ul{ list-style: circle inside; }
```

使用复合属性 `list-style` 时,通常按以上语法格式中的顺序书写,各个样式之间以空格隔开,不需要的样式可以省略。

由于各个浏览器对 `list-style-type` 属性的解析不一致,仅靠 `list-style-type` 无法实现列表项目符号大小的控制,因此在实际网页制作过程中不推荐使用 `list-style-type` 属性来控制项目符号。

实际项目开发中,使用较多的是通过“`list-style: none;`”取消列表默认的项目符号或项目编号,再使用盒子模型的原理来设计列表的样式,有关盒子模型的知识将在本书第 4 章讲解。

取消列表的项目符号或项目编号的代码如下。

```
ul{list-style-type: none; }
```

或者使用 `list-style` 复合属性,代码如下。

```
ul{list-style: none; }
```

3.8 CSS 复合选择器

3.5 节中讲解了常用的标签选择器、类选择器 `class` 选择器、ID 选择器,使用这些基础选择器基本可以选中目标元素,但是在实际网页制作时,一个网页中有很多标签,好多标签存在嵌套关系,如果仅使用 CSS 的基础选择器,不能很好地组织页面样式。

CSS 复合选择器也称为派生选择器,是通过依据元素在其位置的上下文关系来定义样式,由两个或多个基础选择器,通过不同的方式组合而成,可以使标签更加简洁。对 HTML 标签也实现了更强、更方便的选择功能。

3.8.1 群组选择器

在声明 CSS 选择器时,若某些选择器的 CSS 声明是相同的,这时可以使用群组选择器(又称并集选择器),利用集体声明的方法,把相同属性和值的选择器组合同时声明。标签选择器、`class` 类选择器和 ID 选择器等都可以作为并集选择器的一部分,各个选择器之间通过英文状态下的逗号连接。基本语法格式如下。

```
p, h1, .special, #nav{ color: red; }
```

以上代码声明了段落、标题 1 标签内的文字,以及设置了 `class` 属性值为 `special` 和 `id` 属性值为 `nav` 的标签内的文字颜色为 `red`。群组选择器实际上是对 CSS 的一种简化写法,减

少样式重复定义,使代码更加简洁。它的作用与单独声明每一个标签的 CSS 样式是一样的。以下代码与群组选择器声明效果是一样的,代码如下。

```
p{ color: red; }
h1{color: red; }
.special {color: red; }
#nav {color: red; }
```

【例 3-19】 群组选择器(案例文件: chapter03\example\exp3_19.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> 群组选择器 </title>
    <style type = "text/css" >
      span,.special,#tit{color: red; text-decoration: underline; }
    </style>
  </head>
  <body>
    <h1 id = "tit"> 网页设计与制作 </h1>
    <h2> 专题 1 网页设计基础 </h2>
    <h3> 第一节: Web 标准 </h3>
    <p class = "special"> Web 标准不是某一个标准,而是一系列标准的集合. </p>
    <p> Web 标准主要由三部分组成:
    <span> 结构化标准语言 </span> 主要包括 HTML, XHTML 和 XML;
    <span> 表现标准语言 </span> 主要包括 CSS;
    <span> 行为标准 </span> 主要包括 W3C DOM, ECMA Script 等.
    </p>
  </body>
</html>
```

设置了标签内的文本;设置了 class 属性且属性值为 special 的标签内的文本;设置了 id 属性且属性值为 tit 的标签内的文本被设置了红色、下画线。例 3-19 的运行效果如图 3-23 所示。



图 3-23 群组选择器

3.8.2 标签指定选择器

标签指定选择器又称交集选择器,是由两个选择器直接连接构成。第一个选择器为标签选择器,第二个选择器为 class 选择器或 ID 选择器,两个选择器之间不能有空格,必须连续书写。其结果是选中二者各自元素范围的交集,如 p.special,选择所有“class=special”的段落,即指定设置了<p>段落那些 class 属性值或者 id 属性值为指定值的标签。

例 3-20 的 HTML 网页代码中,网页正文中有很多<p>段落,有的<p>段落设置了 class=“special”,即设置了此段落的 class 类属性的值为“special”,其他标签如<h1>标签也设置了 class=“special”属性。若需要选中设置了 class=“special”的段落,而不选择设置了 class=“special”的<h1>标签,基础选择器就无法实现这一要求,必须使用标签指定式选择器。

【例 3-20】 标签指定选择器(案例文件:chapter03\example\exp3_20.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> 标签指定选择器 </title>
    <style type = "text/css" >
      p. spe{color: red; text-decoration: underline; }
    </style>
  </head>
  <body>
    <h1 class = "spe" > 网页设计与制作 </h1>
    <h2> 专题 1 网页设计基础 </h2>
    <h3> 第一节: Web 标准 </h3>
    <p class = "spe" > Web 标准不是某一个标准,是一系列标准的集合. </p>
    <p> Web 标准主要由三部分组成:
    <span class = "spe" > 结构化标准语言 </span> 包括 HTML, XHTML 和 XML,
    <span class = "spe" > 表现标准语言 </span> 主要包括 CSS,
    <span class = "spe" > 行为标准 </span> 包括 W3C DOM、ECMAScript 等.
    </p>
  </body>
</html>
```



p.special 仅选择了所有 class 类属性值为“special”的段落,而没有选中 class=“special”的<h1>标签和标签。例 3-20 的运行效果如图 3-24 所示。

图 3-24 标签指定选择器

3.8.3 后代选择器

当标签发生嵌套时,内层元素就成为外层元素的后代。网页中的导航栏会将导航栏中的各个超链接[<a>标签](#),放在无序列表中的标签内嵌套。如例 3-21 的网页 HTML 代码中,标签就有<a>标签,但是专题学习位于列表之外,若想选中标签下的所有<a>标签,而不选择其他位置的<a>标签,基础选择器就无法实现这一要求。后代选择器又称包含选择器,用来选择元素的后代元素,其写法是把父元素写在前面,后代元素写在后面,中间用空格分隔。

【例 3-21】 后代选择器(案例文件: chapter03\example\exp3_21.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>后代选择器</title>
    <style type = "text/css" >
      ul a{color: red; }
    </style>
  </head>
  <body>
    <ul class = "nav" >
      <li> <a href = "#" > 首页 </a> </li>
      <li>
        <a href = "#" > 课程概要 </a>
        <ul>
          <li> <a href = "#" > 课程简介 </a> </li>
          <li> <a href = "#" > 教学大纲 </a> </li>
          <li> <a href = "#" > 授课计划 </a> </li>
          <li> <a href = "#" > 考核方案 </a> </li>
        </ul>
      </li>
      <li>
        <a href = "#" > 学习资源 </a>
        <ul>
          <li> <a href = "#" > 项目案例 </a> </li>
          <li> <a href = "#" > 技术拓展 </a> </li>
          <li> <a href = "#" > 教程下载 </a> </li>
        </ul>
      </li>
    </ul>
    <a href = "#" > 专题学习 </a>
```

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

定义标签中的所有后代标签<a>标签文字为红色,但处在网页中其他位置的<a>标签(如本例中的“专题学习”)不受影响。

在后代选择器中,“ul a”选择器就是“ul 标签中的 a 标签”,可以解释为“作为 ul 元素后代的所有 a 元素”。使用后代选择器,直接对所需要的元素进行设置,可以避免过多地为标签设置 id 和 class 属性,代码更为简洁。后代选择器还可以多级包含,示例代码如下。

```
ul li ul li a {color: red; }
ul ul li a {color: red; }
```

3.8.4 子选择器

后代选择器能选择标签的所有后代元素,而子选择器只能选择作为某元素子元素的元素,如在例 3-21 的 HTML 代码中,若只想选中标签下的一级导航标签,不想选中二级导航里的 标签,使用后代选择器就不合适了。子选择器使用大于号(子结合符)将父元素和子元素连接起来,表示只选择某个元素的子元素。

【例 3-22】 子选择器(案例文件: chapter03\example\exp3_22.html)
HTML 结构同例 3-21,CSS 代码如下。

```
<style type = "text/css" >
    .nav>li>a {color: red; text-decoration: none; } /* 选择一级导航 */
    .nav>li>ul>li>a {color: green; } /* 选择二级导航 */
</style>
```

例 3-22 的运行效果如图 3-25 所示。

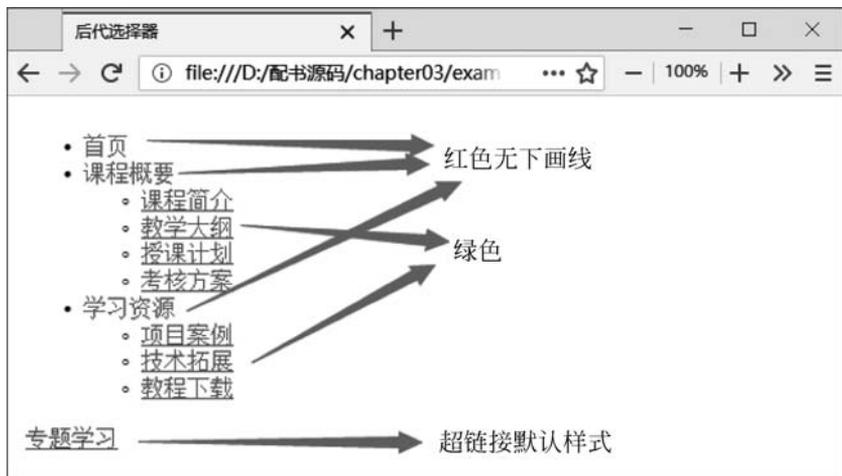


图 3-25 子选择器

子结合符两边可以有空白符,以下写法都是正确的,建议不要有空格。

```
ul>li
ul>li
ul>li
```

3.9 CSS 的继承性

CSS 样式具有继承性,所谓的继承性,是给某些元素设置样式时,子标签会继承父标签的样式。例如 color 属性设置字体颜色,后代自动继承。如果为<body>标签选择器设置文字颜色,其他标签不设置,则所有的后代标签文字颜色都继承<body>的设置(超链接、<h1>~<h6>等标签由于浏览器有默认样式,根据优先级,不显示继承的样式)。具有继承特性的样式,不必对每一个标签都书写样式属性,只需要在父标签中书写一次,子标签继承此属性,可以在一定程度上简化代码,降低 CSS 样式的复杂性。

字体、字号、文字颜色等文本样式属性具有继承性,可以在 body 元素中统一设置,通过继承影响文档中所有文本。本书第 4 章将要学习的盒子模型以及第 5 章将要学习的定位属性则不具有继承性。

【例 3-23】 CSS 的继承性(案例文件:chapter03\example\exp3_23.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 的继承性</title>
    <style type = "text/css" >
      body{
        color: red;
        font-size: 20px;
      }
    </style>
  </head>
  <body>
    <h2>h2 标签</h2>
    <h3>h3 标签</h3>
    <p>p 标签段落 <span> span 标签</span> </p>
    <a href = "# " >a 标签超链接</a>
  </body>
</html>
```

例 3-23 的运行效果如图 3-26 所示。

<P>标签和标签的文字颜色和字号大小继承了对<body>标签的设置。<h1>~



图 3-26 CSS 的继承性

<h6> 标签和 <a> 标签,先继承 <body> 的设置,又由于浏览器有默认的风格,浏览器默认风格的优先级大于继承的风格优先级,所以 <h1> ~ <h6> 标签只显示继承的文本颜色,字号大小采用浏览器默认的风格,<a> 标签继承 body 的字号大小,文字颜色显示浏览器默认的颜色。

3.10 CSS 的优先级

定义 CSS 样式时,经常出现两个或更多规则应用在同一元素上,这时就会出现优先级的问题。

3.10.1 外部样式表、内部样式表和行内样式的优先级

在 3.3 节的例 3-6 中,可以看到外部样式表、内部样式表、行内样式的优先级是“外部样式表 < 内部样式表 < 行内样式”。

当外部样式表、内部样式表都为 <p> 标签定义相同的 CSS 样式时,根据 CSS 的优先级,显示内部样式表定义的 CSS 样式。当外部样式表、内部样式表和行内样式都为 <p> 标签定义了相同的 CSS 样式时,根据 CSS 的优先级,显示行内样式定义的 CSS 样式。

3.10.2 基础选择器的优先级

CSS 基础选择器包括通配符选择器、标签选择器、类选择器和 ID 选择器。为了研究 CSS 基础选择器的优先级,首先来看一个具体的示例。

【例 3-24】 基础选择器的优先级(案例文件: chapter03\example\exp3_24.html)

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title> 基础选择器的优先级 </title>
    <style>
      * {color: pink; }
```

```

        p{color: red; }
        .special-1{color: blue; }
        #special-2{color: green; }
    </style>
</head>
<body>
    <p>第 1 个段落</p>
    <p class = "special-1">第 2 个段落</p>
    <p class = "special-1" id = "special-2">第 3 个段落</p>
</body>
</html>

```

例 3-24 的运行效果如图 3-27 所示。



图 3-27 基础选择器的优先级

以上示例中,使用不同的选择器对同一个元素设置文本颜色,浏览器根据选择器的优先级规则解析 CSS 样式。第 1 个段落,通配符选择器的优先级低于标签选择器,文字颜色显示红色。第 2 个段落,class 选择器优先级大于标签选择器,文字颜色显示蓝色。第 3 个段落,ID 选择器优先级大于 class 选择器,文字颜色显示绿色。各种样式表中基础选择器的优先级总结如图 3-28 所示。

外部样式表	<	内部样式表	<	行内样式
通配符<标签<类<ID		通配符<标签<类<ID		

图 3-28 基础选择器的优先级

3.10.3 复合选择器的优先级

对于由多个基础选择器构成的复合选择器,优先级的判断需要计算权重。CSS 为每一种基础选择器都分配了一个权重,其中,标签选择器权重为 1,类选择器权重为 10,ID 选择器权重为 100。这就能很好地理解前面讲的基础选择器的优先级“标签<类<id”了。此外,继承样式的权重为 0,即子元素定义的样式会覆盖继承来的样式。一个复合选择器的权重值等于组成它的基础选择器权重的叠加(并集选择器除外)。

【例 3-25】 复合选择器的优先级(案例文件: chapter03\example\exp3_25.html)

```

<!DOCTYPE html>
<html lang = "en" >

```

```

<head>
  <meta charset = "UTF-8" >
  <title>复合选择器的优先级</title>
  <style type = "text/css" >
    body{color: #333; } /* 权重为: 0 */
    ul a{color: red; } /* 权重为: 1+1 */
    ul li a{color: pink; } /* 权重为: 1+1+1 */
    .nav li a{color: black; } /* 权重为: 10+1+1 */
    #jianjie{color: green; } /* 权重为: 100 */
    ul li #jianjie{color: orange; } /* 权重为: 1+1+100 */
  </style>
</head>
<body>
  <ul class = "nav" >
    <li> <a href = "#" > 首页 </a> </li>
    <li>
      <a href = "#" > 课程概要 </a>
      <ul>
        <li> <a href = "#" id = "jianjie" > 课程简介 </a> </li>
        <li> <a href = "#" > 教学大纲 </a> </li>
        <li> <a href = "#" > 授课计划 </a> </li>
      </ul>
    </li>
  </ul>
</body>
</html>

```

例 3-25 的运行结果如图 3-29 所示。



图 3-29 复合选择器的优先级

3.11 CSS 的层叠性

当同一个元素被两个选择器选中时,CSS 会根据选择器的权重决定使用哪一个选择器,权重低的选择器效果会被权重高的选择器效果覆盖,这就是 CSS 的层叠性的表现。所

谓层叠性是指多种 CSS 样式的叠加。CSS 的层叠性分以下几种情况。

3.11.1 样式不冲突的层叠

当多种 CSS 样式不冲突时,多种样式叠加。

【例 3-26】 不冲突的层叠(案例文件: chapter03\example\exp3_26.html)

HTML 结构和 CSS 代码如下。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 层叠性</title>
    <style type = "text/css" >
      p{
        color: red;
        font-size: 14px;
      }
      .special{
        font-weight: bold;
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    <p>段落文本 1</p>
    <p class = "special" id = "special">段落文本 2</p>
    <p>段落文本 3</p>
  </body>
</html>
```

例 3-26 的运行效果如图 3-30 所示。



图 3-30 样式不冲突的层叠

3.11.2 样式冲突且存在优先级差别的层叠

当多种 CSS 样式冲突,且选择器存在优先级差别时,优先级高的选择器定义的样式将覆盖优先级低的选择器定义的样式。

【例 3-27】 存在优先级差别的层叠(案例文件: chapter03\example\exp3_27.html)
HTML 结构和 CSS 代码如下。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 层叠性</title>
    <style type = "text/css" >
      p{
        color: red;
        font-size: 14px; /* 冲突 */
      }
      .special{
        font-size: 24px; /* 冲突 */
        font-weight: bold;
        background-color: yellow;
      }
    </style>
  </head>
  <body>
    <p>段落文本 1</p>
    <p class = "special" id = "special">段落文本 2</p>
    <p>段落文本 3</p>
  </body>
</html>
```

例 3-27 的运行效果如图 3-31 所示。



图 3-31 样式冲突且存在优先级差别的层叠

3.11.3 样式冲突且选择器优先级相同的层叠

当多种 CSS 样式冲突,且选择器优先级相同时,例如,同一选择器被多次定义,采取“后来居上”原则,即最后定义的样式将覆盖先前定义的样式。

【例 3-28】 同一选择器被多次定义(案例文件: chapter03\example\exp3_28.html)
HTML 结构和 CSS 代码如下。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 层叠性</title>
    <style type = "text/css" >
      p{font-size: 24px; color: red; }
      p{color: green; }
      p{color: blue; }
    </style>
  </head>
  <body>
    <p>段落文本 1</p>
    <p class = "special" id = "special" >段落文本 2</p>
    <p>段落文本 3</p>
  </body>
</html>
```

例 3-28 的运行效果如图 3-32 所示。



图 3-32 同一选择器被多次定义

3.11.4 样式冲突且同一标签运用不同类型选择器的层叠

同一标签运用不同类型的选择器时,优先级高的选择器定义的样式覆盖优先级低的选择器定义的样式。

【例 3-29】 同一标签运用不同类型选择器(案例文件: chapter03\example\exp3_29.html)

HTML 结构与 CSS 代码如下。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" >
    <title>CSS 层叠性</title>
    <style type = "text/css" >
      .special{ background-color: yellow; }
      #special{ background-color: pink; }
    </style>
  </head>
  <body>
    <p>段落文本 1</p>
    <p class = "special" id = "special">段落文本 2</p>
    <p>段落文本 3</p>
  </body>
</html>
```

例 3-29 的运行效果如图 3-33 所示。



图 3-33 同一标签运用不同类型选择器

【项目实战】

项目 3-1 百度搜索结果网页局部样式设置(案例文件目录: chapter03\demo\demo3_1)

效果图

项目 3-1 的运行效果如图 3-34 所示。

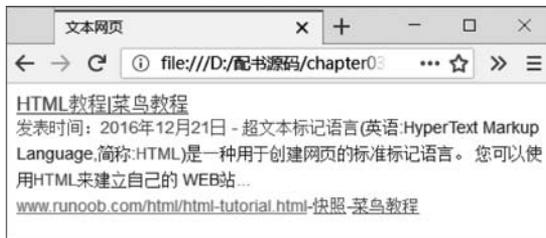


图 3-34 文本网页效果

思路分析

这是一个纯文本的网页,在百度搜索等搜索引擎中搜索的界面效果。效果图中,带下划线的是超链接,文本的颜色主要有红色、灰色、蓝色、绿色和黑色,其中,黑色是浏览器默认的文本颜色,蓝色是浏览器默认的超链接的颜色。第一行,可以用<h2>标签嵌套<a>标签,红色的文字使用标签,最后一行使用<p>标签嵌套3个<a>标签,第一个<a>标签通过<class>选择器(cite)设置文字颜色为绿色,第二和第三个<a>标签通过class选择器(gray)设置为灰色。中间的文字利用<p>标签,设为一个段落,灰色和红色的文字使用标签,设置标签的文字颜色为红色,并用class选择器(gray)设置文字颜色为灰色。HTML结构分析如图3-35所示。

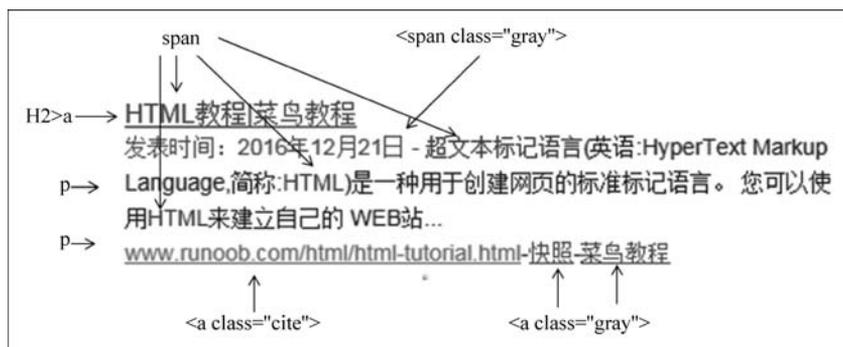


图 3-35 HTML 结构图示

制作步骤

Step01. 根据图 3-35 HTML 结构分析图,编写 HTML 结构代码。

```
<!DOCTYPE html>
<html lang = "en" >
  <head>
    <meta charset = "UTF-8" />
    <title> 文本网页 </title>
  </head>
  <body>
    <h2>
      <a href = "# " > <span>HTML 教程 </span> | 菜鸟教程 </a>
    </h2>
    <p>
      <span class = "gray"> 发表时间: 2016 年 12 月 21 日 - </span> <span>超文本标记语言 </span> (英语: HyperText Markup Language, 简称: <span>HTML</span>) 是一种用于创建网页的标准标记语言. 您可以使用 <span>HTML</span> 来建立自己的 WEB 站 ...
    </p>
    <p>
      <a href = "# " class = "cite"> www.runoob.com/html/html-tutorial.html </a> - <a href = "# " class = "gray"> 快照 </a> - <a href = "http://http://www.runoob.com/" class = "gray"> 菜鸟教程 </a>
    </p>
  </body>
</html>
```

```

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

```

Step02. 建立内部 CSS 样式表,清除浏览器默认的风格,由于本例中使用了<p>、<h2>、<a>、 4 种标签,因为后两种标签不具备 margin 属性和 padding 属性,所以仅对<p>和 标签进行样式设置即可。

```

<head>
  <meta charset = "UTF-8" />
  <title> 本网页</title>
  <style type = "text/css">
    p,h2{margin: 0; padding: 0; }
  </style>
</head>

```

Step03. 通过设置<body> 标签的 CSS 样式,编写可以被继承的文本样式——文字字体。

```

<style type = "text/css">
  p,h2{margin: 0; padding: 0; }
  body{font-family: arial,"宋体"; }
</style>

```

Step04. 设置<h2> 标签的文字样式。

```

<style type = "text/css">
  p,h2{margin: 0; padding: 0; }
  body{font-family: arial,"宋体"; }
  h2{font-size: 16px; font-weight: 400; }
</style>

```

此时,在浏览器中测试页面显示效果如图 3-36 所示。

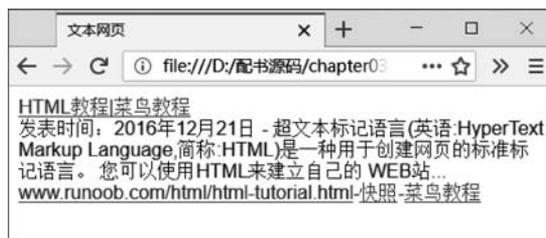


图 3-36 Step04 效果

Step05. 从图 3-36 中可见, 标签内的“HTML 教程”下画线是蓝色的,而效果图

中下画线是红色的。设置标签的文字样式可以解决这个问题。

```
<style type = "text/css" >
  p,h2{margin: 0; padding: 0; }
  body{font-family: arial,"宋体"; }
  h2{font-size: 16px; font-weight: 400; }
  span{color: #c00; }
</style>
```

此时,在浏览器中测试页面显示效果如图 3-37 所示。

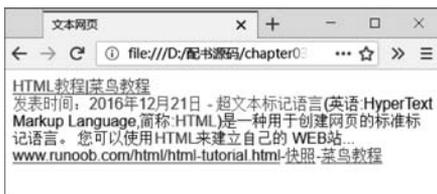


图 3-37 Step05 效果

Step06. 从图 3-37 中可以看到,中间的段落文字颜色是黑色而不是深灰色(#333),标签内的“发表时间:2016年12月21日-”文字颜色是红色。设置段落<p>标签的文字颜色、字号、行间距,为 class = “gray”设置文字颜色为浅灰色(#666)。

```
<style type = "text/css" >
  p,h2{margin: 0; padding: 0; }
  body{font-family: arial,"宋体"; }
  h2{font-size: 16px; font-weight: 400; }
  span{color: #c00; }
  p{
    font-size: 14px;
    color: #333;
    line-height: 22px;
  }
  .gray{color: #666; }
</style>
```

此时,在浏览器中测试页面显示效果如图 3-38 所示。

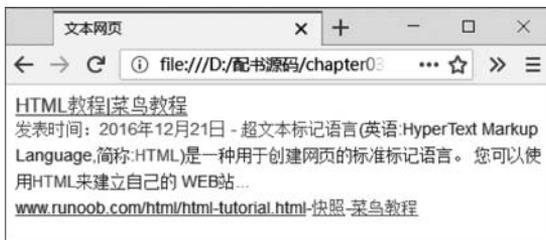


图 3-38 Step06 效果

Step07. 设置最后一行,第一个超链接的文字样式。

```
<style type = "text/css">
    p,h2{margin: 0; padding: 0; }
    body{font-family: arial,"宋体"; }
    h2{font-size: 16px; font-weight: 400; }
    span{color: #c00; }
    p{
        font-size: 14px;
        color: #333;
        line-height: 22px;
    }
    .gray{color: #666; }
    .cite{color: #4e9c62; }
</style>
```

Step08. 浏览器兼容性测试,即将所完成的案例页面在多个浏览器中运行测试其兼容性,本案例在 Google Chrome、火狐、Opera、Safari for Windows 及 IE 11 版块中测试通过。

项目 3-2 端午节习俗新闻页面(案例文件目录: chapter03\demo\demo3_2)

效果图

项目 3-2 的运行效果图如图 3-39 所示。



图 3-39 纯文本网页样式效果

思路分析

在本案例中,整篇文章使用 id 属性为 page 的 <div> 标签组织结构,并通过 ID 选择器设置文章的。利用 <h2> 标签设置大标题,<h3> 标签设置作者、时间等信息,通过群组选择器设置居中样式。<h4> 标签设置小标题,<p> 标签设置段落,并通过群组选择器设置首行缩


```
    }  
  </style>  
</head>
```

Step03. 编写<h2>、<h3>标签文本 CSS 样式,h2 采用了默认的文字大小和加粗,h3 设置文字字号,通过 font-weight 取消浏览器默认的加粗效果,利用群组选择器设置居中。

```
<style type = "text/css" >  
  #page {  
    font-family: "宋体";  
    color: #333;  
    font-size: 16px;  
    line-height: 1.5em;  
  }  
  h2,h3{text-align: center; }  
  h3{  
    font-size: 18px;  
    font-weight: normal;  
  }  
</style>
```

Step04. 通过群组选择器设置<h4>、<p>标签首行缩进两字符。

```
<style type = "text/css" >  
  #page {  
    font-family: "宋体";  
    color: #333;  
    font-size: 16px;  
    line-height: 1.5em;  
  }  
  h2,h3{text-align: center; }  
  h3{  
    font-size: 18px;  
    font-weight: normal;  
  }  
  h4,p {text-indent: 2em; }  
</style>
```

Step05. 通过标签指定选择器为 class="right"设置居右对齐。

```
<style type = "text/css" >  
  #page {  
    font-family: "宋体";  
    color: #333;
```

```

        font-size: 16px;
        line-height: 1.5em;
    }
    h2,h3{text-align: center;}
    h3{
        font-size: 18px;
        font-weight: normal;
    }
    h4,p {text-indent: 2em;}
    p. right{text-align: right;}
</style>

```

Step06. 浏览器兼容性测试,即将所完成的案例页面在多个浏览器中运行测试其兼容性,本案例在 Google Chrome、火狐、Opera、Safari for Windows 及 IE 11 版块中测试通过。

【实训作业】

实训任务 3-1 Web 前端试学班广告页局部样式设置

利用 <p> 标签、标题标签、 标签和 <a> 标签以及 CSS 技术制作页面效果,如图 3-40 所示。



图 3-40 Web 前端试学班广告局部效果

具体要求如下。

1. 首页命名为 index 或 default, 首页扩展名为 .html 或 .htm。
2. 合理使用 <p> 标签、标题标签、 标签和 <a> 标签组织网页结构。
3. 合理使用 CSS 选择器, 选择器命名规范, 设置文本 CSS 样式。

实训任务 3-2 “商品推荐”页局部样式设置

利用所学 HTML 标签和 CSS 技术制作“商品推荐”页面效果,如图 3-41 所示。具体要求如下。

1. 目录结构明晰,有默认图像文件夹(图像文件夹命名为 images 或 img),图像文件命名规范。
2. 根目录下有首页,首页命名为 index 或 default, 首页扩展名为 .html 或 .htm。
3. 合理使用所学 HTML 标签组织网页结构和布局。
4. 合理使用 CSS 选择器,选择器命名规范,设置文本 CSS 样式。



华为荣耀30 Pro ¥3999-5499 ¥4999-6499
来自数码达人悟空大师推荐
超曲OLED飞瀑屏 麒麟9905G SoC
999666人说好

图 3-41 “商品推荐”页面效果