第3章

CSS 基 础

CSS(Cascading Style Sheets,层叠样式表)是一种用来定义 HTML 或者 XML 等结构化文档样式的计算机语言,它不仅可以静态地修饰网页,也能与各种脚本语言协同工作,实现对网页元素的动态设置。基于 CSS 技术,可以将页面的内容与表现形式分离。若要进行全局样式更新,只需简单地修改样式,即可实现网站中对应元素样式的自动更新。

3.1 CSS 简介

3.1.1 基本语法

样式表由一系列 CSS 规则组成,规则主要由选择器和声明(一条或者多条)构成。样式表示例如下。

```
selector{declaration 1 ; declaration 2; … ; declaration N}
```

选择器通常是需要改变样式的 HTML 元素,每条声明由一个属性和一个属性值组成。属性是希望设置的样式属性,每个属性有一个值,属性和值之间用冒号分隔,如下所示。

```
selector{property:value}
```

【示例 3-1】 声明示例。

声明示例代码如图 3-1 所示,其作用是将<h1>元素内的文字颜色定义为蓝色,同时将字体大小设置为 12 像素,注意:

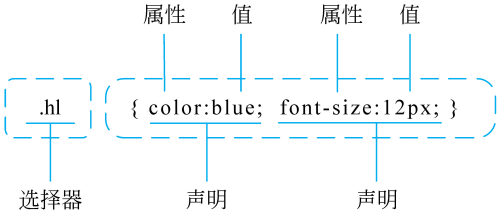

图 3-1 声明示例

(1) 值可以有不同写法和单位。

（2）如果值为若干单词，则需要将值放入引号内。

（3）如果不止一个声明，则需要使用分号将每个声明分隔。

（4）是否包含空格不会影响 CSS 在浏览器的工作效果。

（5）CSS 一般对大小写不敏感。

（6）选择器可以进行分组，用逗号将需要分组的选择器分开。

【示例 3-2】 用 CSS 定义段落颜色为蓝色。

```
p{color: blue;}
p{color: #0000ff;}
p{color: #00f;}
p{color: rgb(0,0,255);}
p{color: rgb(0%,0%,100%);}
```

被分组的选择器可共享相同的声明。示例 3-3 中对所有的标题元素进行了分组，设置所有的标题元素颜色为绿色。

【示例 3-3】 用 CSS 设置标题颜色为绿色。

```
h1,h2,h3,h4,h5,h6{
    color: green;
}
```

3.1.2 引入方式

当读到一个样式表时，浏览器会根据它来格式化 HTML 文档。CSS 样式按其所在位置可以分为行内样式、内部样式和外部样式三类。

1. 行内样式

将 CSS 样式直接书写在 HTML 标签内部，称为行内样式或者内联样式，是 CSS 样式的一种基本形式，如示例 3-4 所示。

【示例 3-4】 行内样式。

```
<h1 style="color:red;font-size:24px;">行内样式</h1>
```

行内样式需要书写在标签的 style 属性中，样式属性和值之间用冒号分隔，多个样式之间用分号分隔。

行内样式存在不足：如果一个页面中有大量标签，其中还包含很多重复标签，仍需要给每个标签都编写行内样式，这是一件非常麻烦的事情。此外，将 HTML 标签与 CSS 样式混杂、耦合在一起，也不利于代码的调试和修改，所以行内样式仅作为 CSS 样式的一种基本形式，并不提倡在实际项目中使用。

2. 内部样式

内部样式将 CSS 样式添加在＜head＞与＜/head＞标签之间，并用＜style＞与＜/style＞标签进行声明。内部样式将 CSS 样式与 HTML 标签分离，使页面更加整洁。

【示例 3-5】　内部样式。

```
<!Doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>内部样式</title>
    <!--将 CSS 样式添加在<head>与</head>标签之间,并用<style>与</style>标签进行
声明-->
    <style>
        p {
            color: red;
            font-size: 24px;
        }
    </style>
</head>
<body>
        <p>内部样式</p>
</body>
</html>
```

内部样式优于行内样式,可以实现 CSS 样式与 HTML 标签的分离。但是,内部样式只对当前页面有效,如果多个页面中有很多相同的样式,CSS 代码冗余度较大,因此内部样式也并不提倡使用。

3. 外部样式

外部样式是存储在一个单独的外部 CSS 文件中的 CSS 规则。利用网页文件头部中的＜link＞标签,该文件被链接到 Web 站点中的一个或多个页面上。

使用外部样式表,可通过改变一个 CSS 文件来改变整个站点的外观。外部样式表可以在任何文本编辑器中被编辑,但是不能包含任何 HTML 标签,以“.css”扩展名进行保存。

当样式需要应用于很多页面时,外部样式表将是最理想的选择。

【示例 3-6】　外部样式。

```
<!Doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>外部样式</title>
    <!--<link>标签引用外部 css 文件,为文档获取相应的样式-->
    <link rel="stylesheet" href="color.css">
</head>
<body>
        <p>外部样式</p>
</body>
</html>
```

在这段代码中,使用＜link＞元素引入外部样式,rel 属性用于设置链接的关系,这里设

置为样式表，href 属性用于设置外部样式的文件路径。

3.2 CSS 选择器

CSS 选择器是一种模式，用于选择需要设置样式的 HTML 元素。使用 CSS 选择器可以对 HTML 页面中的标签实现一对一、一对多或者多对一的控制。

CSS 选择器有标签选择器、ID 选择器、类选择器、复合选择器、伪元素选择器以及伪类选择器等。

3.2.1 标签选择器

标签选择器用于控制标签的样式，是指用 HTML 标签名称作为选择器，然后按标签名称为页面中某一类标签指定统一的 CSS 样式，创建或更改标签的 CSS 规则后，所有标签名对应的元素的样式都会立即更新，语法格式如下所示。

```
标签名{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

标签选择器的优点是能快速地为页面中同类型的标签统一样式；但同时，这也是它的缺点：不够灵活，不能提供差异化样式。

3.2.2 ID 选择器

ID 选择器通过 HTML 元素的 ID 属性对它进行唯一性标识，ID 选择器定义时名称前加“#”，且名称必须是字母或下画线开头，不能是数字，其语法格式如下所示。

```
#ID 名{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

【示例 3-7】 CSS 中 ID 选择器的应用。

```
#red {color: red;}
#green {color: green;}
```

在上述 HTML 代码中，ID 为 red 的<p>元素内容显示为红色，而 ID 为 green 的<p>元素内容显示为绿色。

【示例 3-8】 HTML 中 ID 选择器的应用。

```
<p id="red">这个段落是红色。</p>
<p id="green">这个段落是绿色。</p>
```

在 Web 页面中，多个 HTML 标签可以设置相同 ID，浏览器可正常解析，但是，当页面中 JavaScript 根据 document.getElementById("ID")定位标签时，无法准确定位，如示例 3-9 所示。

【示例 3-9】 页面中 JavaScript 定位元素的示例。

```
<!Doctype html>
<html>
```

```
<head>
    <meta charset="utf-8">
    <title>页面中 JavaScript 定位元素的示例</title>
    <style>
        #red {
            color: red;
            font-size: 24px;
        }
    </style>
    <script language"javascript">
        window.onload = function () {
            var aa = document.getElementById("red");
            console.log(aa.innerHTML);              //仅在控制台输出第一个 ID 选择器
        }
    </script>
</head>
<body>
    <p id="red">第一个 ID 选择器</p>
    <p id="red">第二个 ID 选择器</p>
</body>
</html>
```

运行代码后，浏览器控制台只输出第一个 ID 为 red 的元素内容，并不能输出所有 ID 为 red 的元素内容。

3.2.3　类选择器

类选择器是用户自定义名称的选择器，使样式作用于被 class 属性限定的 HTML 元素。类选择器在网页中可以应用任意多次，定义时名称前加“.”，且名称必须是字母和下画线开头，不能是数字，其语法格式如下所示。

```
.类名{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

类选择器可以为元素对象定义单独或相同的样式，使用时需要设置 HTML 元素的 class 属性，并指定属性值为类选择器的名称，具体使用方法如下所示。

```
<p class="blue">这是类选择器</p>
```

类选择器的使用不局限于某个元素，而是适用于所有具有 class 属性的元素。例如，以上为<p>元素设置的类选择器同样可以作用于<h1>元素，使用方法如下所示。

```
<h1 class="blue">这里也可以使用类选择器</h1>
```

【示例 3-10】 类选择器的使用方法。

```
<!Doctype html>
<html>
<head>
```

```
    <meta charset="utf-8">
    <title>类选择器的使用方法</title>
    <style>
        .red{
            color:red;
            font-size:20px;
        }
    .blue{
        color:blue;
        font-size:25px;
    }
    </style>
</head>
<body>
    <p class="red">红色字段</p>
    <p class="blue">蓝色字段</p>
    <h3 class="blue">H3 蓝色字段</h3>
</body>
</html>
```

运行这段代码后，效果如图 3-2 所示。

图 3-2　类选择器的使用方法

类选择器可为任何具有 class 属性的元素设置样式。当页面中包含多个相同元素，且大部分元素使用相同样式，个别元素使用不同样式时，可以先使用标签选择器为所有元素设置相同样式，再使用类选择器为个别元素设置不同样式。

【示例 3-11】　类选择器设置不同样式。

```
<!Doctype html>
<html>
<head>
    <meta charset="utf-8">
    <title>类选择器设置不同样式</title>
    <style>
        p{
            color:blue;
            font-size:24px;
        }
        .red{
```

```
            color:red;
            font-size:30px;
        }
    </style>
</head>
<body>
    <p>默认段落样式</p>
    <p>默认段落样式</p>
    <p class="red">特殊段落样式</p>
    <p>默认段落样式</p >
    <p>默认段落样式</p>
</body>
</html>
```

运行代码，效果如图 3-3 所示。

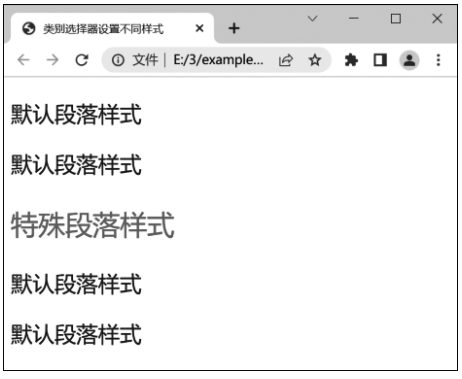

图 3-3　类选择器设置不同样式

3.2.4　复合选择器

复合选择器由两个或多个基本选择器，通过不同的方式组合而成，可选择更准确、更精细的目标元素，包括上下文关系选择器、交集选择器和并集选择器。

1. 上下文关系选择器

上下文关系选择器简称关系选择器，也称为派生选择器，常见的关系选择器有后代选择器、子代选择器和相邻兄弟选择器。

(1) 后代选择器。

后代选择器可选择元素的后代元素，语法格式如下所示。

```
选择器 1 选择器 2{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

外层标签写在前面，内层标签写在后面，中间用空格分隔。当标签发生嵌套时，内层标签就成为外层标签的后代。

【示例 3-12】　后代选择器。

```
<!Doctype html>
```

```
<html>
    <head>
    <meta charset="utf-8">
    <title>后代选择器</title>
    <style>
        p span{
            color:red;
        }
        span{
            color:green;
        }
    </style>
</head>
<body>
    <p>嵌套<span>标记</span>的颜色</p>
    没有嵌套<span>标记</span>的颜色
</body>
</html>
```

运行代码，效果如图 3-4 所示。

图 3-4　后代选择器

（2）子代选择器。

子代选择器，也称父子选择器，是特殊的后代选择器，用来选择某元素的直接后代，语法格式如下所示。

```
选择器 1>选择器 2{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

父子选择器之间用大于号“＞”分隔，子代选择器一般放在后代选择器之后，否则，后代选择器的效果会被子代选择器覆盖，如示例 3-13 所示。

【示例 3-13】　子代选择器。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width, initial-scale=1.0">
    <title>子代选择器</title>
    <style>
        #links a {
            color: red;
            /* 元素 links 的所有超链接为红色 */
        }
```

```
        #links>a {
            color: blue;
            /* 元素 links 的首个超链接被修改为蓝色 */
        }
    </style>
</head>
<body>
    <div id="links">
        <a href="">超链接 1</a>
        <div>
            <a href="">超链接 2</a>
            <a href="">超链接 3</a>
        </div>
    </div>
</body>
</html>
```

运行代码，效果如图 3-5 所示。

图 3-5　子代选择器

下述代码中，因为所有＜li＞都是＜ul＞的后代，因此，所有列表项都为蓝色。

```
ul>li{
      color:blue;
    }
```

（3）相邻兄弟选择器。

相邻兄弟选择器可选择紧接在另一元素后的元素，且二者有相同父元素，语法格式如下所示。

```
选择器 1+选择器 2{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

选择器使用相邻兄弟结合符“＋”分隔，例如，“h1 ＋ p {color:red;}”表示选择紧接在＜h1＞标签后出现的段落，设置其字体颜色为红色。其中，＜h1＞和＜p＞标签拥有共同的父标签＜body＞。

2. 交集选择器

交集选择器由两个选择器构成，其中第一个为标签选择器，第二个为类选择器或 ID 选择器，两个选择器之间不能有空格，语法格式如下所示。

```
选择器 1 选择器 2{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

【示例 3-14】 交集选择器。

```
<!Doctype html>
<html>
<head>
    <meta charset-"utf-8">
    <title>交集选择器</title>
    <style>
        div{
        width:400px;
        height:100px;
        margin-left:50px;
        color:white;
        font-size:24px;
        font-weight:bold;
        text-align:center;
        line-height:100px;
        background:lightgreen;
        }
        .red{
            background:lightred;
        }
        .blue{
            background:lightblue;
        }
        div.green{/*交集选择器*/
            color:black;
        }
    </style>
</head>
<body>
    <div class="red">红底白字</div>
    <div class="green">绿底黑字</div>
    <div class="blue">蓝底白字</div>
</body>
</html>
```

运行代码，效果如图 3-6 所示。

图 3-6 交集选择器

3. 并集选择器

并集选择器可同时选中多个基本选择器所选元素范围。任何形式的选择器都可以组成“并集”，多个选择器之间通过逗号“，”连接，使用并集选择器与单独使用各个基本选择器的效果一样，语法格式如下所示。

```
选择器 1,选择器 2{属性 1:属性值 1; 属性 2:属性值 2; …; 属性 n:属性值 n;}
```

【示例 3-15】　并集选择器。

```
<!Doctype html>
<html>
<head>
    <meta charset-"utf-8">
    <title>并集选择器</title>
    <style>
        h2,li,.class1,#id{/* 并集选择器 */
            color:blue;
            font-size:24px;
        }
    </style>
</head>
<body>
    <h2>第一行数据</h2>
    <p class="class1">第二行数据</p>
    <div id="id">第三行数据</div>
</body>
</html>
```

运行代码，效果如图 3-7 所示。

图 3-7　并集选择器

3.2.5　伪元素选择器

伪元素（Pseudo-element）是 HTML 中并不存在的元素，CSS 伪元素用于设置元素指定部分的样式，包括首字母、首行等，例如，“::first-letter”伪元素用于向某个选择器中的第一个字母添加特殊样式。

在 CSS 3 中，伪元素由两个冒号“::”开头，后接伪元素的名称，放在选择器之后，用于选择指定的元素。考虑到兼容性，CSS2 中的伪元素使用一个冒号“:”。

伪元素的语法如下所示。

```
selector:: pseudo-element{property:value;}
```

CSS 类也可以与伪元素配合使用，语法如下所示。

```
selector.class::pseudo-element{property:value;}
```

如果要对文本首行设置特殊样式，可使用如下代码。

```
p::first-line{
    color:#ff0000;
    }
```

在伪元素中，“::before”用于在元素之前添加内容，“::after”用于在元素之后添加内容，“::first-line”用于在文本的首行设置特殊样式，且“::first-line”和“::first-letter”只能用于块级元素。

3.2.6 伪类选择器

伪类不特指某一个元素，而是指一个元素的特殊状态，用于向选择器添加特殊的效果。例如，被单击的元素、鼠标移入的元素等。同一个元素，根据不同的状态，有着不同的样式，其语法如下所示。

```
selector:pseudo-class{property:value;}
```

CSS 类也可以与伪类搭配使用，语法如下所示。

```
selector.class:pseudo-class{ property:value;}
```

例如，超链接单击之前样式设定为“a:link{color:#ff0000;}”，详情见 3.3.5 节。

3.3 CSS 样式

CSS 样式可设置 HTML 页面中的文本、图片内容以及版面布局和外观显示，用以美化网页，常用的 CSS 样式包括字体样式、文本样式、图片样式、背景样式、超链接样式等。

3.3.1 字体样式

字体样式可设置文字的字体族、大小、粗细、是否倾斜、英文字母大小写、字符间距、阴影效果等，字体样式属性和属性值如表 3-1 所示。

表 3-1 字体样式属性和属性值

属　性	属 性 值
font-family	常用字体包括：楷体、宋体、微软雅黑、Arial、Helvetica、sans-serif 等

续表

属　性	属　性　值
font-size	medium，默认值
	length，某个固定值，常用单位为 px、em 和 pt
	百分数(%)，相对值，基于父元素或默认值的一个百分比值
	inherit，继承父元素的字体大小
font-weight	normal 默认值，bold 粗字体，bolder 更粗字体，lighter 更细字体，number(范围为 100～900)，inherit 继承父级字体粗细
font-style	normal 默认值，itatic 斜体，inherit 继承父级字体风格
text-transform	none 默认值，capitalize 单词以大写字母开头，uppercase 全部大写，lowercase 全部小写，inherit 继承父元素
letter-spacing	normal 默认，字符间没有额外空间，length 定义字符间的固定空间(允许使用负值)，inherit 继承父元素
text-shadow	text-shadow：h-shadow、v-shadow、blur、color。其中，水平阴影位置 h-shadow 必需，允许为负值；垂直阴影位置 v-shadow 必需，允许为负值；模糊距离 blur 可选；阴影颜色 color 可选

【示例 3-16】　字体样式。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>字体样式</title>
    <style>
      .明月几时有{font-style:italic;}
      .把酒问青天{font-style:normal;}
    </style>
</head>
<body>
    <h1 class="明月几时有">明月几时有</h1>
    <h1 class="把酒问青天">把酒问青天</h1>
</body>
</html>
```

运行代码，效果如图 3-8 所示。

图 3-8　字体样式代码效果

注意：将元素粗细设置为 400 相当于 normal，设置为 700 相当于 bold。对于中文网页来说，一般多用 bold 和 normal，不建议使用数值。

【示例 3-17】 设置元素粗细。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>设置元素粗细</title>
    <style>
        h1.weight1{font-weight: 100;}
        h1.weight2{font-weight: 200;}
        h1.weight3{font-weight: 300;}
        h1.weight4{font-weight: 400;}
        h1.weight5{font-weight: 500;}
        h1.weight6{font-weight: 600;}
        h1.weight7{font-weight: 700;}
        h1.weight8{font-weight: 800;}
        h1.weight9{font-weight: 900;}
    </style>
</head>
<body>
    <h1 class="weight1">皑如山上雪</h1>
    <h1 class="weight2">皎若云间月</h1>
    <h1 class="weight3">闻君有两意</h1>
    <h1 class="weight4">故来相决绝</h1>
    <h1 class="weight5">今日斗酒会</h1>
    <h1 class="weight6">明旦沟水头</h1>
    <h1 class="weight7">躞蹀御沟上</h1>
    <h1 class="weight8">沟水东西流</h1>
    <h1 class="weight9">凄凄复凄凄</h1>
</body>
</html>
```

运行代码，效果如图 3-9 所示。

图 3-9 设置元素粗细代码效果

3.3.2　文本样式

文本样式有很多种，常见样式包括文本颜色、行高、文本装饰、水平对齐方式等。

1. 文本颜色

文本颜色 color 用于定义文本的颜色，取值方式有以下三种。

(1) 预定义的颜色值名称，如 red、green、blue 等。

(2) 十六进制色彩码，如＃FF0000、＃FF6600、＃29D794 等。

(3) RGB 三基色代码，如红色可以表示为 rgb(255,0,0)或 rgb(100%,0%,0%)。

实际开发中，常用十六进制颜色定义方式。如果使用 RGB 代码的百分比颜色值，取值为 0 时，也不能省略百分号，必须写为 0%。

【示例 3-18】　文本样式。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>文本样式</title>
    <style>
        h1.color{color: red;}
        h1.color1{color: #123abc;}
        h1.color2{color: rgb(21, 210, 147);}
    </style>
</head>
<body>
    <h1 class="color">人生若只如初见</h1>
    <h1 class="color1">何事秋风悲画扇</h1>
    <h1 class="color2">等闲变却故人心</h1>
</body>
</html>
```

运行代码，效果如图 3-10 所示。

图 3-10　文本样式代码效果

2. 行高

CSS 样式属性 line-height 用于设置行间的距离，一般称为行高。行高属性值常用单位有 3 种，分别为像素 px、相对值 em 以及百分比%，实际开发中，多用像素 px，如图 3-11 所示。

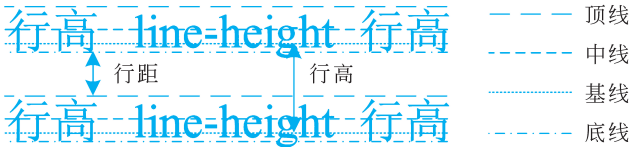

图 3-11　行高属性值示意

3. 文本装饰

CSS 样式属性 text-decoration 用于设置文本的下画线、上画线、删除线等装饰效果，可用属性值包括以下 4 种。

（1）none 表示没有修饰（默认值）。

（2）underline 表示下画线。

（3）overline 表示上画线。

（4）line through 表示删除线。

注意：text-decoration 后可赋多个值，用于给文本添加多种显示效果。例如，希望文字同时有下画线和删除线效果，可以将 underline 和 linethrough 同时赋给 text-decoration，也可利用 text-decoration 的属性值 none 去掉超链接的下画线。

【示例 3-19】 去掉超链接的下画线。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>去掉超链接的下画线</title>
    <style>
        h1.none{
              text-decoration: none;
          }
        h1.underline{
              text-decoration: underline;
          }
        h1.through{
              text-decoration: line-through;
          }
        h1.overline{
              text-decoration: overline;
          }
    </style>
</head>
```

```
    <body>
        <h1 class="none">万丈穹庐人醉</h1>
        <h1 class="underline">星影摇摇欲坠</h1>
        <h1 class="through">归梦隔狼河</h1>
        <h1 class="overline">又被河声搅碎</h1>
    </body>
</html>
```

运行代码，效果如图 3-12 所示。

图 3-12　去掉超链接的下画线的代码效果

4. 水平对齐方式

水平对齐方式 text-align 属性用于设置文本内容的水平对齐，可用属性值包括：默认值 left 左对齐、right 右对齐和 center 居中对齐。

【示例 3-20】　水平对齐方式 text-align。

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>水平对齐方式 text-align</title>
    <style>
        h1.align_left{
            text-align:left;
        }h1.align_right{
            text-align:right;
        }
        h1.align_center{
            text-align:center;
        }
    </style>
</head>
<body>
    <h1 class="align_left">飞絮飞花何处是</h1>
    <h1 class="align_right">层冰积雪摧残</h1>
    <h1 class="align_center">疏疏一树五更寒</h1>
```

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

运行代码，效果如图 3-13 所示。

图 3-13　水平对齐方式 text-align 代码效果

3.3.3　图片样式

1. 控制图片的大小

在 CSS 中，利用属性宽度 width 和高度 height 设置图片的大小，height 和 width 属性不包括内边距、边框和外边距，它设置的是元素内容的高度和宽度，如图 3-14 所示。图片大小属性如表 3-2 所示。

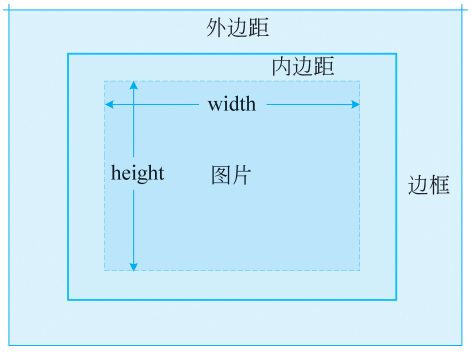

图 3-14　元素内容宽度和高度

表 3-2　图片大小属性

属　性	值	描　述
width	auto	默认，浏览器计算实际宽度
	length	以 px、em 等定义宽度
	%	以包含块(父元素)的百分比定义宽度
	inherit	从父元素继承的 width 属性值
height	auto	默认，浏览器计算高度
	length	以 px、em 等定义高度

续表

属　性	值	描　述
height	%	以包含块(父元素)的百分比定义高度
	inherit	从父元素继承的 height 属性值

不管图片实际大小是多少,都可使用 width 和 height 来定义。

2. 图片边框

在 CSS 中,使用 border 属性定义图片的边框,border 属性允许指定元素边框的样式、宽度和颜色(表 3-3)。

表 3-3　border 属性

属　性	描　述
border-width	指定四个边框的宽度,用于上边框、右边框、下边框和左边框
border-style	指定要显示的边框类型
border-color	设置四个边框的颜色,或者使用 border 简洁写法,如 border: 1px solid gray;

3. 图片水平对齐

text-align 一般用在两个地方:文本水平对齐和图片水平对齐,即 text-align 只对文本和<img>标签有效,对其他标签无效,text-align 属性值如表 3-4 所示。

表 3-4　text-align 属性值

属　性　值	描　述
left	默认值,左对齐
center	居中对齐
right	右对齐

注意: 图片是在父元素中进行水平对齐,因此,若要对图片进行水平对齐,需要在父元素中设置 text-align 属性。

4. 图片垂直对齐

vertical-align 属性设置元素的垂直对齐方式,顶线、中线、基线以及底线位置如图 3-15 所示。vertical-align 属性值如表 3-5 所示。

图 3-15　顶线、中线、基线以及底线位置示意图

表 3-5　vertical-align 属性值

属　性　值	描　　述
top	顶部对齐，把元素的顶端与行中最高元素的顶端对齐
middle	中部对齐，把此元素放置在父元素的中部
baseline	基线对齐，默认值，元素放置在父元素的基线上
bottom	底部对齐，元素及其后代元素的底部与整行的底部对齐

3.3.4　背景样式

1. background-color 设置背景颜色

```
h1{
    background-color:red;
  }
```

2. background-image 设置背景图片

```
body{
     background-image:url(图片路径);
    }
```

url 表示引入图片的路径。

3. background-repeat 设置背景的重复方式

若图片大小不能很好适配浏览器窗口，可指定其重复方式。浏览器默认将图片重复铺满全屏，设置背景图片重复方式格式如下所示。

```
body{
     background-repeat:重复方式;
    }
```

“no-repeat”表示不重复，“repeat ”表示重复，“repeat-x”表示沿 x 轴方向重复，“repeat-y”表示沿 y 轴方向重复。

3.3.5　超链接样式

在浏览器中，默认情况下，超链接字体为蓝色，带有下画线，鼠标单击时字体为红色，单击后为紫色。在 CSS 中，可使用超链接伪类来定义超链接在鼠标单击不同时期的样式，如表 3-6 所示。