第3章 CSS技术基础

在实际开发中，HTML主要用来显示网页的内容，而网页样式的设计通常交由层叠样式表(Cascading Style Sheet，CSS)来负责。这样做的好处是使得内容与表现分离，便于对网站进行开发与维护，并能提高用户体验。本章主要介绍CSS技术的基础知识，读者重点需要掌握CSS选择器的用法以及如何使用DIV进行网页的布局。

3.1 CSS的基本语法

CSS样式表是由若干条样式规则组成，每一条样式规则由三部分组成：选择器(selector)、属性(property)和值(value)，基本语法如下：

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

selector的含义为指定选择网页中标签的方式，property表示选择器所指定标签包含的属性，value表示属性的值。例如，可以定义如下一个简单的样式表：

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

其作用在网页中的简单流程为，先选择网页中所有的<p>标签，然后设置<p>标签中所有字体的颜色为蓝色，大小为20个像素。

样式规则必须放在一对大括号({})内，可以是一条或多条，属性和值之间用英文冒号(:)分开，每条规则以英文分号(;)结尾。

3.2 在HTML中使用CSS

在HTML中引入CSS样式的方法有3种，分别为行内样式、内部样式表和外部样式表，下面分别介绍这3种方式的优缺点和应用场景。

3.2.1 行内样式

行内样式是通过在HTML标签中添加style属性来引入CSS。

【例3-1】 添加style属性创建页面3-1.html。

```
<body>
```

```
    <p style="color:blue; fontsize:20px">蓝色字体,大小 20 像素</p>
    <p style="color:green; font-size:15px">绿色字体,大小 15 像素</p>
</body>
```

例 3-1 所示代码的显示效果如图 3-1 所示。

图 3-1　应用行内样式

由于行内样式将 CSS 代码直接写在 HTML 中,不能使得内容与样式分离,因此在实际开发中应用的不多。但是,由于行内样式只对当前的 HTML 标签起作用,往往在特定的场景中使用。

3.2.2　内部样式表

内部样式表就是将 CSS 代码写在 HTML 的 <head>标签中,与 HTML 内容位于同一个页面中。

【例 3-2】 利用内部样式表创建页面 3-2.html。

```
<head>
    <meta charset="UTF-8">
    <title>内部样式</title>
    <style>
        h1{
            color: red;
        }
    </style>
</head>
<body>
    <h1>我是红色标题</h1>
</body>
```

3.2.3　外部样式表

外部样式表就是将 CSS 代码保存为一个单独的文件,文件扩展名为.css。例如,可以在 Web 应用工程中创建 out.css 文件,该文件存放到 css 文件夹下,如图 3-3 所示。

图 3-2　应用内部样式表

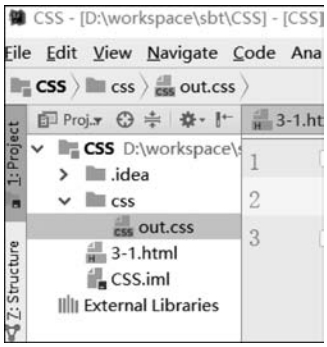

图 3-3　创建外部样式表文件

在 out.css 中添加如下代码：

```
p{
    font-style: italic;
}
```

【例 3-3】 利用外部样式表创建页面 3-3.html。

```
<head>
    <meta charset="UTF-8">
    <title>外部样式表</title>
    <link href="css/out.css" ref="stylesheet" type="text/css">
</head>
<body>
    <p>应用了外部样式表的段落</p>
</body>
```

其中，<link>标签的属性 ref="stylesheet"指在页面中引用外部样式表，type="text/css"是指链接文件的类型是样式表文本，href 属性用来指定 css 文件所在的路径。上述代码通过引用外部样式表的方式为当前页面设置样式，该方式实现了样式和内容的彻底分离。一个外部样式表可以应用到多个页面。当改变这个外部样式表文件时，所在页面的样式会随之改变。这在制作具有大量相同风格的网站时非常有用。

3.2.4 3 种方式的优先级

CSS 名为层叠样式表的原因是允许同时给网页中的元素应用多个样式，页面元素的最终样式即为多个样式的叠加效果。由于相互叠加的样式可能引起冲突，因此首先需要了解样式的优先级。

【例 3-4】 利用层叠样式表创建页面 3-4.html。

```
<head>
    <meta charset="UTF-8">
    <title>3 种方式优先级</title>
    <link href="css/out.css" ref="stylesheet" type="text/css">
    <style>
        h1{
            color: red;
        }
    </style>
</head>
<body>
    <h1 style="color:green">我是什么颜色?</h1>
</body>
</html>
```

并在 out.css 文件中添加如下代码：

```
h1{
    color: blue;
}
```

例 3-4 所示代码分别给页面中的< h1 >标签分别应用了行内样式(设置标题为绿色)、内部样式表(设置标题为红色)和外部样式表(设置标题为蓝色),该页面的< h1 >标签最终显示的字体为绿色,这是因为行内样式离该标签最近。又或者将例 3-4 所示代码的行内样式去掉,并将内部样式和外部样式交换位置,如例 3-5 所示。

【例 3-5】 创建页面 3-5. html。

```
< head >
    < meta charset = "UTF - 8">
    < title > 3 种方式优先级</title >
    < style >
        h1{
            color: red;
        }
    </style >
    < link href = "css/out.css" ref = "stylesheet" type = "text/css">
</head >
< body >
    < h1 >我是什么颜色?</h1 >
</body >
```

这一次,由于< h1 >标签离外部样式表最近,因此为蓝色。一般来讲,CSS 中设置样式优先级的方式为“就近原则”。

3.3 CSS 选择器

选择器是 CSS 中一个非常重要的概念,所有 HTML 样式都是通过不同的 CSS 选择器进行控制的。前面学过的选择器,如< p >、< h1 >等都是最简单的标签选择器,接下来介绍更多功能强大的选择器,以帮助我们方便、快速、准确地进行样式设计。

3.3.1 基本选择器

CSS 的基本选择器主要分为标签选择器、ID 选择器和类选择器 3 种。

1. 标签选择器

每种 HTML 标签的名称都可以作为相应的标签选择器名称。例如,前面学过的 h1 选择器用于声明页面中所有< h1 >标签的样式风格,p 选择器用于声明页面中所有< p >标签的样式风格。标签选择器可以同时为页面中多个相同标签指定样式。

2. 类选择器

页面中,元素的 class 属性可以为元素分组。例如下述代码将 3 个< p >标签分为一组。

```
< p class = "title">我是属于 title 组</p >
< p class = "title">我是属于 title 组</p >
< p class = "title">我是属于 title 组</p >
```

类选择器的作用便是声明同一组元素标签的样式。定义类选择器的方式是在 class 属性值的前面加一个句点“.”,例如下述代码定义了一个名为. title 的样式。

```
.title{
```

```
    font-size: 16px;
    color: #00509F;
}
```


其中,定义的样式(字体大小和颜色)会作用于页面中所有 class 属性值为 title 的标签。

3. ID 选择器

在页面中,元素的 id 属性用来唯一标识该元素。同样,ID 选择器主要用来对某个特定元素定义样式。与类选择器不同的是,使用 ID 选择器定义样式时,必须在 ID 名称前加一个“#”号。

【例 3-6】 利用 ID 选择器创建页面 3-6.html。

```
<head>
    <meta charset="UTF-8">
    <title>ID 选择器</title>
    <style>
        #wzu{
            color:red;
        }
    </style>
</head>
<body>
    <p id = "wzu">ID 属性为 WZU 的段落</p>
    <p id = "zju">ID 属性为 ZJU 的段落</p>
</body>
```

例 3-6 所示代码为页面中 id 属性值为 wzu 的元素设置样式(字体设置成红色)。

3.3.2 高级选择器

本节主要介绍 CSS 常用的高级选择器,如层次选择器、并集选择器和交集选择器。

1. 层次选择器

HTML 中各元素间主要的层次关系包括后代、父子、相邻兄弟和一般兄弟等几种关系。这些关系被抽象为 HTML 的文档对象模型(Document Object Model,DOM)。下面首先通过一个示例来讲述什么是 DOM。

【例 3-7】 创建页面 3-7.html。

```
<body>
    <a class = "you">1</a>
    <a>2</a>
    <a>3</a>
    <ul>
        <li class = "you"><a>4</a></li>
        <li id = "my"><a>5</a></li>
        <li><a class = "you">6</a></li>
        <li><a>7</a></li>
    </ul>
    <h1>h1</h1>
    <p>p</p>
</body>
```

上述代码对应的 DOM 模型如图 3-4 所示。其中< a > 1、< a > 2、< a > 3 和< ul >是< body >的子元素,< a > 1、< a > 2、< a > 3 和< ul >互为兄弟元素,< li >是< ul >的子元素,< a > 4、< a > 5、< a > 6 是< li >的子元素。所有元素都是< body >的后代元素。

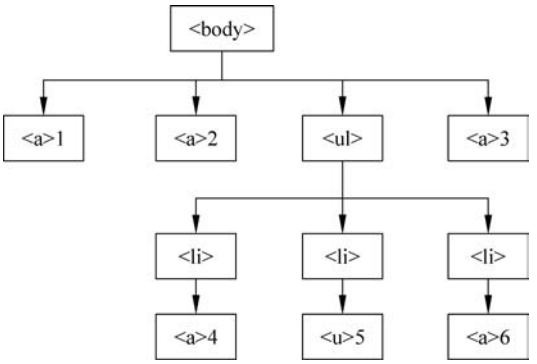

图 3-4　页面 3-7.html 的元素树形图

层次选择器通过 DOM 模型可以快速选择需要的元素,具体语法如表 3-1 所示。

表 3-1　层次选择器

选择器	类　　型	功 能 描 述
A B	后代选择器	选择元素 A 的所有与元素 B 匹配的后代元素
A>B	子选择器	选择元素 A 的所有与元素 B 匹配的子元素
A+B	相邻兄弟选择器	选择元素 A 后紧跟的与元素 B 匹配的元素
A~B	一般兄弟选择器	选择元素 A 后所有与元素 B 匹配的元素

2. 后代选择器

后代选择器的作用就是选择某元素的指定后代元素,例如给页面 3-7.html 添加如下样式后,浏览器会选择< body >标签内的所有< a >标签,并设置其背景为红色。

```
body a{
    background: red;
}
```

3. 子选择器

子选择器只能选择某元素的所有匹配子元素,例如给页面 3-7.html 添加如下样式:

```
body > a{
    background: red;
}
```

与后代选择器不同的是,此选择器仅选择了< body >标签内的所有子< a >标签(此例为< a > 1、< a > 2、< a > 3),并设置背景色为红色。而在< li >中的< a >标签(< a > 4、< a > 5、< a > 6、< a > 7)将不会被选择。

4. 相邻兄弟选择器

相邻兄弟选择器可以选择紧跟在另一个元素后面的元素,它有一个相同的父级元素。例如给页面 3-7.html 添加如下样式:

```
#my + li{
    color:red;
}
```


此选择器选择了 id 属性值为 my 所在标签后面的第一个< li >元素(此例为< a > 6 所在的< li >),并设置背景为红色。

5. 一般兄弟选择器

一般兄弟选择器选择某元素后面的所有兄弟元素,例如在页面 3-7. html 中添加如下样式:

```
#my～li{
    color:red;
}
```

此选择器选择了 id 属性值为 my 所在标签后面的所有< li >元素(此例为< a > 6 和< a > 7 所在的< li >),并设置背景为红色。

6. 并集选择器

并集选择器是指多个选择器的合并,各个选择器用“,”隔开。主要用来给多个选择器设置相同的样式,例如在页面 3-7. html 中添加如下样式:

```
#my,h1,p{
   background: yellow;
}
```

此选择器会选择页面中 id 属性为 my 的元素、所有< h1 >元素以及所有< p >元素,并设置各个元素的背景色为黄色。

7. 交集选择器

交集选择器也是指多个选择器的组合,但交集选择器需要同时满足其中每一个选择器的条件,并且各个选择器之间没有分隔符。例如在页面 3-7. html 中添加如下样式:

```
a.you{
    background: yellow;
}
```

此选择器中的 a. you 实际上是标签选择器 a 和类选择器. you 的组合,即选择页面中所有 class 属性值为 you 的< a >元素(此例为< a > 1 和< a > 4),并设置背景色为黄色。

3.4 CSS 样式设置

CSS 的样式设置主要包括文本样式、超链接和列表样式以及背景样式等。下面分别对这几类样式设置进行介绍。

3.4.1 文本样式

文字是网页中最重要的组成部分,通过文字可以传递多种信息。CSS 对网页文字的设置主要包括设置字体大小、类型、颜色、风格以及文本段落的对齐方式、行高、文字缩进等。

1. ＜span＞标签

在 HTML 中,< span >标签本身没有显示效果,通常用来与 CSS 结合来给段落中的文字进行样式设置。

【例 3-8】 利用< span >标签创建页面 3-8. html。

```
< head >
    < meta charset = "UTF - 8">
    < title > span 标签</title >
    < style type = "text/css">
        p{ font - size: 14px;}
        p . poem{font - size: 30px;}
        p ＃Li{font - size: 24px;font - weight: bold}
        p span{font - size: 20px; font - style: italic}
    </style >
</head >
< body >
    < p >锄禾日当午,汗滴禾< span class = "poem">下土</span >。</p >
    < p >霍元甲、< span id = "Li">陈真</span >和叶问。</p >
    < p >拳王< span >阿里</span >、泰森、霍利菲尔德</p >
</body >
```

例 3-8 所示代码显示效果如图 3-5 所示。

图 3-5　span 标签的应用

从页面效果可以看出,< span >标签可以为段落中的部分文字添加样式,并且不会像< p >标签一样独占一行,这种不独占一行的元素称为行内元素。另外,例 3-8 中分别对< span >标签内的字体进行了大小(font-size)、风格(font-style)、粗细(font-weight)等样式进行了设置。表 3-2 列出了几个常用的字体属性及其可选参数值。

除了字体属性外,CSS 对文本样式的控制还包括文本颜色、水平对齐方式、首行缩进、行高、文本装饰等属性设置,具体用法如表 3-3 所示。

表 3-2 常用字体属性设置

属 性 名	含 义	举 例
font-family	设置子类型	font-family:“宋体”
font-size	设置字体大小	font-size:15px
font-style	设置字体风格	font-style:italic
font-weight	设置字体粗细	font-weight:bold
font	在一个声明中设置所有字体属性	font:italic bold 30px “宋体”

表 3-3 常用文本样式设置

属 性 名	含 义	举 例
color	设置文本颜色	color:red
text-align	设置水平对齐方式	text-align:left
text-indent	设置首行文本的缩进	text-indent:18px
line-height	设置文本的行高	line-height:20px
text-decoration	设置文本的装饰	text-decoration:underline

更多的字体属性设置可参考 W3C 文档(https://www.w3school.com.cn/css/css_font.asp/)。

3.4.2 超链接

网页中的超链接有默认的伪类样式：超链接文字有下画线,单击超链接前文本颜色为蓝色,单击超链接后文本颜色为紫色,鼠标单击未释放时的超链接为红色。所谓伪类样式就是根据元素处于某种行为或者状态时的特征来修饰样式。基本语法为“标签名：伪类名{声明;}”。常用的超链接伪类有 4 种,如表 3-4 所示。

表 3-4 超链接伪类

伪 类 名 称	含 义	伪类样式举例
A:link	未单击访问的状态	A:link{color:yellow}
A:visited	单击访问后的状态	A:visited{color:blue}
A:hover	鼠标悬浮其上的状态	A:hover{color:red}
A:active	鼠标单击未释放的状态	A:active{color:green}

需要注意的是,对超链接伪类设置样式时要按照如下顺序进行：a:link-> a:visited-> a:hover-> a:active,如果先设置“a:hover”再设置“a:visited”,则“a:visited”将不起作用。

【例 3-9】 创建页面 3-9.html。

```
<head>
    <meta charset = "UTF - 8">
    <title>超链接样式</title>
    <style type = "text/css">
        a{text - decoration: none}
        a:link{color: #00509F}
        a:visited{color: yellowgreen}
        a:hover{
```

```
            text-decoration: underline;
            color: antiquewhite;
        }
        a:active{color:darkred;}
    </style>
</head>
<body>
    <a href="http://www.baidu.com" target="_blank">钢铁侠</a>
</body>
```

例 3-9 所示代码首先去掉超链接的默认下画线，设置超链接未访问前的颜色为＃00509F，访问后的颜色为 yellowgreen，鼠标悬停其上时出现下画线并改变颜色为 antiquewhite，鼠标单击未释放时颜色变为 darkred。

3.4.3 背景样式

背景在网页中无处不在，它能为整体页面带来丰富的视觉效果。在学习背景属性前，先介绍一个网页布局中最常用的双标签： 标签。与标签（行内元素）不同的是，一对 标签独占一行（块元素）。在实际开发中， 标签通常与 CSS 配合使用来对网页进行排版，制作出复杂多样的网页。另外，背景样式主要包括背景颜色（background-color）、背景图片（background-image）、背景图片大小（background-size）、背景图片重复方式（background-repeat）和背景图片位置。

【例 3-10】 创建页面 3-10.html。

```
<head>
    ...
<style type="text/css">
        #nav{
            width:200px;
            background-color: beige;
        }
        .title{
            background-color:#C00;
            background-image: url("down_arrow.jpg");
            background-size:15%;
            background-repeat:no-repeat;
            background-position: 170px 0px;
        }
    </style>
</head>
<body>
    <div id="nav">
        <h1 class="title">电影分类</h1>
        <ul>
            <li><a href="#">枪战片</a><a href="#">犯罪片</a></li>
            <li><a href="#">伦理片</a><a href="#">纪录片</a></li>
            <li><a href="#">爱情片</a><a href="#">武侠片</a></li>
        </ul>
    </div>
</body>
```

例 3-10 所示代码首先通过 ID 选择器设置整个 DIV 的宽度和背景色，然后通过类选择器设置标题的背景颜色、背景图片、背景大小、背景图片是否重复及背景位置等，显示效果如图 3-6 所示。除了可以分开设置背景图片的各个属性外，还可以使用 background 属性将多个属性综合起来声明，如例 3-10 中设置背景图片的代码可以改成如下形式：

```
.title{
    background: #C00 url("down_arrow.jpg") 170px 0px no-repeat;
    background-size:15%;
}
```

这样一来，background 属性将背景图片的颜色、路径、位置和重复属性放到一起声明，其效果与图 3-6 一致。

图 3-6　背景样式设置

从图 3-6 中可以看出，<a>标签也属于行内元素，不独占一行。更多背景设置可参考文档：https://www.w3school.com.cn/css/css_background.asp。

3.5　CSS 盒子模型

盒子模型是网页制作中的一个重要的知识点，它是使用 DIV＋CSS 进行网页布局的基础。先来看一个例子，图 3-7 展示的是由多个相框有序组成的一面相框墙，其中每一个相框都可以抽象成一个矩形，该矩形有一个边框（border），边框和相片的距离成为内边距（padding），每个相框之间的距离成为外边距（margin）。这种 padding-border-margin 的模型是一种极其通用的描述矩形对象布局形式的方法，我们称为“盒子”模型。

在网页设计中，不管多复杂的页面也是由一个个矩形区域合理地组织在一起形成的，因此也可以应用“盒子”模型来布局。在 CSS 中，一个独立的盒子模型由元素内容（element content）、边框（border）、内边距（padding）和外边距（margin）四部分组成，如图 3-8 所示。

其中元素内容位于最中间，是网页显示的内容，边框就是包着元素内容的外框，一般具有一定的高度和宽度。内边距是网页内容和边框之间的距离。外边距为不同盒子间的距离。使用 CSS 可以分别对盒子的边框、外边距和内边距进行样式控制。

图 3-7　盒子模型举例

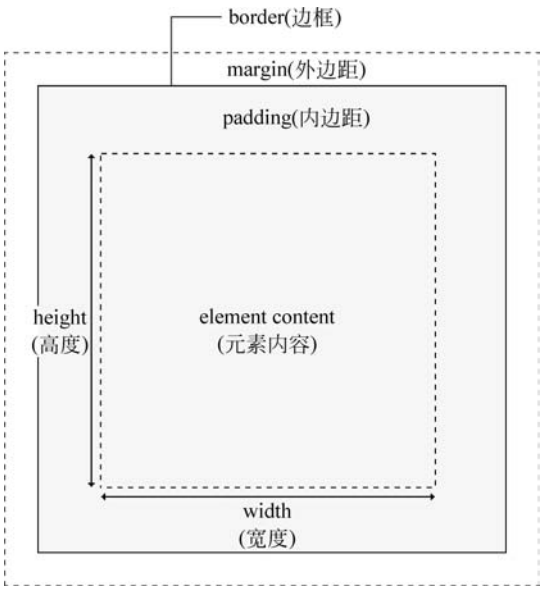

图 3-8　盒子模型

【例 3-11】 创建页面 3-11.html。

```
<head>
    <style type="text/css">
        .box{border:1px solid #3a6}
        form{background-color: antiquewhite}
        h2{background-color: aquamarine}
        #pwd{background-color: yellow}
    </style>
</head>
<body>
    <div class="box">
        <h2>用户登录</h2>
        <form>
            <div>姓名:<input type="text"></div>
            <div id="pwd">密码:<input type="password"></div>
            <div>
```

```
                <input type = "submit" value = "提交">
                <input type = "reset" value = "重置">
            </div >
        </form >
    </div >
</body >
```

例 3-11 所示代码显示效果如图 3-9 所示。

图 3-9　用户登录案例

图 3-9 中一共有 6 个盒子，分别为最外层的< div >标签（class 属性为 box）、< h2 >标签（块元素）、表单元素以及表单内的 3 个< div >标签。其中最外层< div >标签的边框粗细为 1px，类型为实线，颜色为 # 3a6；表单背景颜色为 antiquewhite；< h2 >标签背景颜色为 aquamarine；密码框所在< div >标签背景色为 yellow。从图 3-9 中可以看出，最外层< div >标签与< body >标签之间、< h2 >标签与最外层< div >标签和< form >标签之间都存在默认的 margin（如果以< body >标签为参照物，则< body >标签与最外层< div >标签的距离便是 padding）。由于 HTML 中很多标签都有默认的外边距并且不同浏览器中默认的 margin 值不同，因此在实际开发中通常统一设置默认的 margin 为 0，使得网页在不同浏览器中显示效果一致，只需添加如下 CSS 代码便可很容易实现。

```
* {margin: 0}
```

其中，* 表示匹配所有标签。另外，也可以根据需求个性化设置不同盒子间的外边距，例如，可以单独增加文本框所在盒子间的下外边距。

```
form div{
            margin - bottom: 10px;
}
```

类似地，也可以给盒子设置内边距，例如下述代码给< form >标签设置了内边距。

```
form{
    …
    padding: 20px 10px;
}
```

边框、内边距和外边距都可以分为上、下、左、右 4 个部分进行单独设置，读者可自行查阅 W3C 文档(https://www.w3school.com.cn/css/css_boxmodel.asp)。

3.6 CSS+DIV 网页布局

虽然网页基本上都包括头部、主体内容和尾部 3 个部分，但不同网页在布局上往往各不相同。网页默认的布局方式为标准文档流方式，所谓标准文档流是指网页根据块级元素或行内元素的特性按从上到下、从左到右的方式自然排列。标准文档流有如下几条重要的特性。

(1) 块级元素无论内容多少，都会独占一行。

(2) 块级元素可以设置高度和宽度。

(3) 行内元素不会独占一行。

(4) 行内元素的高度和宽度由内容撑开。

【例 3-12】 创建页面 3-12.html。

```
<head>
<style type="text/css">
  div{
     border: 1px solid blue;
     width: 100px;
     height: 100px;
  }
  span{
     border: 1px solid red;
     width: 100px;        /*高度宽度设定无效*/
     height: 100px;
  }
</style>
</head>
<body>
    <div>DIV 块级元素</div>
    <span>span 行内元素 1</span>
</body>
```

例 3-12 所示代码在网页中显示效果如图 3-10 所示。

从图 3-10 中可以看出，块级元素 独占一行并且所设定的高度和宽度生效，而行内元素不独占一行并且声明的高度、宽度无效。

如果按照标准文档流对网页进行布局，则大部分复杂效果将无法实现。因此，很多时候需要将元素脱离标准文档流或者改变元素的性质来达到复杂布局的需求。其中，元素的浮动和定位是最常用的两种方式。

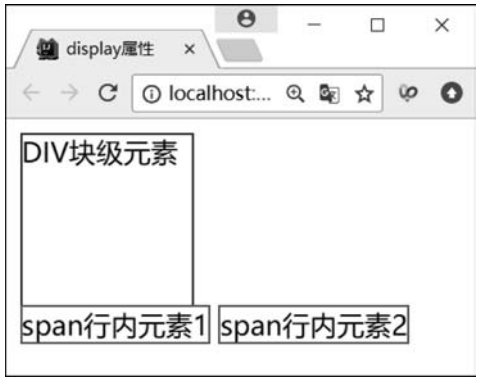

图 3-10 标准文档流

3.6.1 浮动

浮动是指将块级元素排列在一行并且支持宽度和高度设定的方法。要实现浮动需要在CSS中设置float属性,默认值为none。如果将float属性值设置为left或right,元素就会向其父元素的左侧或者右侧浮动。

【例3-13】 创建页面3-13.html。

```
<head>
<style type="text/css">
        div{
            border: 1px solid blue;
            width: 100px;
            height: 100px;
            float: left;
        }
    </style>
</head>
<body>
    <div id="d1">div1</div>
    <div id="d2">div2</div>
</body>
```

例3-13所示代码显示效果如图3-11所示。

从图3-11中可以看出,div1和div2脱离了标准文档流向左浮动,从而使得它们能排列在同一行,并且原来的宽度和高度有效。另外,例3-13所示代码中2个 标签都设置了左浮动,如果只有div1设置了浮动,则只有div1会脱离标准文档流并且原先所在的位置会被div2所占据。需要注意的是,div2的文字内容会环绕浮动块显示而不会被覆盖。例如,将例3-13中的CSS代码修改如下,则显示的效果如图3-12所示。

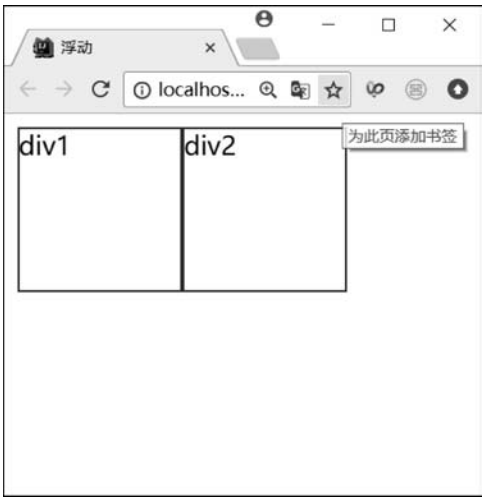

图3-11 左浮动

图3-12 单个浮动的影响

```
div{
   border: 1px solid blue;
   width: 100px;
   height: 100px;
}
#d1{
   float: left;
}
```

修改后的效果说明了元素的浮动会影响其他元素的位置。若要使得标准文档流中的元素不受其他浮动元素的影响，则需要用到 clear 属性来清除浮动。例如在页面 3-13.html 中添加如下代码。则 div2 会保留在原来的位置而不受 div1 浮动的影响。

```
#d2{
   clear:left;
}
```

clear 属性的值可以为 left、right 和 both，在实际开发中为了方便起见，通常不管上一个元素是左浮动还是右浮动都统一设置为 both。另外，还可以使用元素的 display 属性来进行行内元素和块级元素的互换，从而满足不同的需求。

3.6.2 定位

要设计复杂的页面效果，仅靠浮动是不够的，还需要对元素进行定位。CSS 中使用 position 属性来对元素进行定位。position 属性有 4 个值，分别代表着不同的定位类型。

(1) static：默认值，没有定位。元素按照标准文档流进行布局。

(2) relative：相对定位，盒子的位置以标准文档流为基准，然后相对于原本所在的位置偏移指定的距离。相对定位后的盒子仍在标准文档流中，其后的盒子仍以标准文档流方式对待。

(3) absolute：绝对定位，以它最近的一个已经定位的祖先元素为基准进行定位。如果没有祖先元素被定位，则以< body >标签为基准(浏览器左上角)。绝对定位的盒子会从标准文档流中脱离，并且对其后的其他盒子的定位没有影响。

(4) fixed：与 absolute 类似，不同的是以浏览器窗口为基准进行定位。

【例 3-14】 创建页面 3-14.html。

```
< style type = "text/css">
        div{
            padding: 5px;
            font - size: 10px;
        }
        #outer{
            margin: 10px;
            border: 1px #555 solid;
        }
        #in1{
            background - color: yellowgreen;
        }
        #in2{
```

```
            background-color: lightseagreen;
        }
        #in3{
            background-colr: papayawhip;
        }
    </style>
</head>
<body>
    <div id="outer">
        <div id="in1">div1</div>
        <div id="in2">div2</div>
        <div id="in3">div3</div>
    </div>
</body>
```

显示效果如图 3-13 所示。

使用相对定位将 div1 定位的位置相对于原始位置向右上方移动 20px，修改例 3-14 相应的 CSS 代码如下：

```
#in1{
            background-color: yellowgreen;
            position: relative;
            top: -20px;
            left: 20px;
        }
```

定位后的效果如图 3-14 所示。

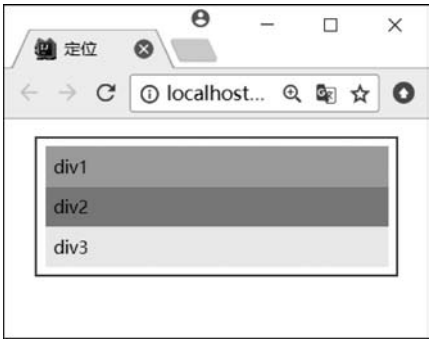

图 3-13　定位前效果

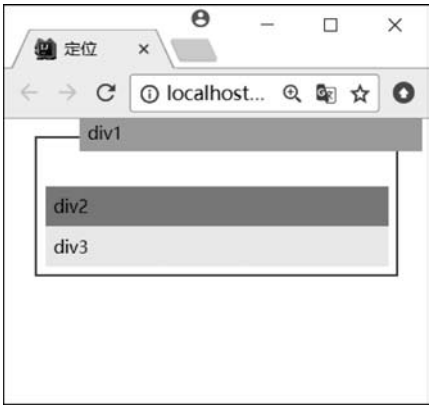

图 3-14　相对定位效果

left 和 top 可以取正值也可以取负值，left 取正值元素向右移动，top 取正值元素向下移动。这是因为网页的坐标原点在页面的左上角，纵坐标向下为正方向，横坐标向右为正方向。从图 3-14 中可以看出 div1 相对于起始位置偏移了一定的距离，并且其他元素不受影响，说明相对定位并没有脱离标准文档流。

再来看看绝对定位的例子，在 3-14.html 页面中添加以下代码：

```
#in2{
```

```
    background-color: lightseagreen;
    position:absolute;
    top: 0px;
    left: 0px;
}
```

#in2 样式中，由于包含 div2 的父元素没有被定位，因此 div2 将以浏览器左上角为基准，偏移指定距离，偏移后该元素脱离标准文档流，如图 3-15 所示。如果在外层< div >标签上添加如下样式，则显示效果如图 3-16 所示，此时将以离它最近的已经定位的祖先元素为基准进行定位。

```
#outer{
   …
   position: relative;
}
```

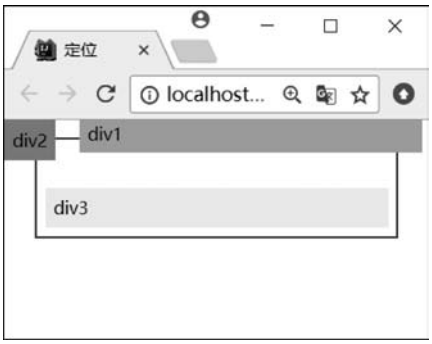

图 3-15　绝对定位效果 1

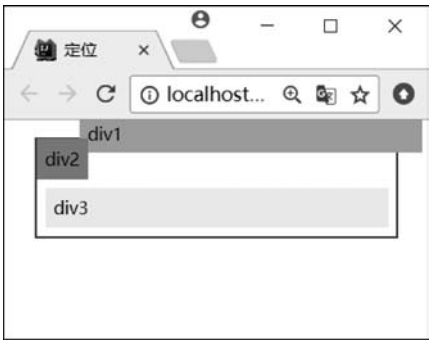

图 3-16　绝对定位效果 2

小　　结

本章主要讲解了如何创建样式表来控制页面的样式和布局，主要包括使用 CSS 来添加背景、格式化文本以及格式化边框，并定义元素的填充和边距以及浮动和定位。学习 CSS 的重点是理解并掌握 CSS 选择器的使用及盒子模型，尽管现阶段的实际开发中大多使用框架进行网页样式设计，但了解底层的 CSS 设计还是有一定必要的。