第 3 章

CHAPTER 3

CSS 样式设计

学习目标

- 熟练掌握 CSS 的样式规则，CSS 在 HTML5 页面中的应用和各种 CSS 选择器的使用，CSS 的层叠性和优先级。
- 熟练掌握各种 HTML5 App 开发常用的 CSS 属性和响应式布局设计。
- 掌握使用 Chrome 的“开发者工具”对 CSS 样式进行调试。

CSS 样式设计是 HTML5 App 开发和微信小程序开发中最重要的技术之一，有了它才真正实现了内容与外观的分离，通过它可以控制页面的布局、样式、动画等，并实现移动设备的适配。目前 CSS 也是各公司 HTML5 工程师必备的技能之一。本章针对 CSS 的语法规则、各种常用的 CSS 属性、CSS 在 Chrome 中的调试等重要内容作详细的讲解。

3.1 CSS 简介

CSS 即层叠样式表(Cascading Style Sheet)目前的最新版本是 CSS3。在页面制作时采用 CSS 技术，可以有效地对页面的布局、字体、颜色、背景，甚至动画效果实现精确控制。只要对相应的代码做一些简单的修改，就可以改变同一页面的外观。CSS 禅意花园(http://www.csszengarden.com/)是网站设计领域最著名的网站之一，网站提供了一张 HTML 页面，设计师们为它设计出成百上千个 CSS 样式文件，这张页面通过更换样式表呈现出各式各样、令人惊叹的效果，如图 3-1 所示，这两张页面的源码是一样的，只是样式表文件不同，这让人不禁感叹 CSS 的强大。

图 3-1 禅意花园的不同 CSS 设计

在页面中使用CSS技术,可以设计出更加整洁、漂亮的页面,它解决了内容与外观分离的问题。科学地编写CSS,还可以大大提高页面样式的复用性。

3.2 CSS核心基础

3.2.1 CSS样式规则

使用HTML时,需要遵守一定的规范,CSS也是如此。要想熟练地应用CSS进行页面样式设计,首先需要了解CSS的样式规则。CSS定义的语法格式如下:

```
selector {property1: value1;property2: value2;...}
```

其中,selector代表CSS选择器,property代表CSS属性,value代表的是CSS属性对应的值,图3-2是一个典型的CSS定义,它的作用是将h1标签内的文字颜色设置为红色,字体大小设为14像素。

图3-2 一个典型的CSS定义

由于各浏览器厂商对CSS3各属性的支持程度不一样,因此,有少数CSS3属性需要用厂商的前缀加以区分,通常把这些加上私有前缀的属性叫“私有属性”,以便于在不同的浏览器下更好地体验CSS3特性。表3-1列举了各主流浏览器的私有前缀,目前App和小程序都主要支持-webkit-前缀。

表3-1 主流浏览器私有属性

内 核	浏 览 器	私有前缀
Trident	IE8/IE9/IE10/IE11	-ms-
Webkit	Chrome/Safari	-webkit-
Gecko	Firefox	-moz-
Presto	Opera	-o-

3.2.2 CSS中的单位和颜色

1. 单位

在HTML5 App开发中常用的单位及说明如表3-2所示。

表3-2 CSS单位及说明

单位	描 述
%	百分比,以父元素的大小计算
em	通常1em=16px,2em=32px,当用于指定字体大小时,em单位是指父元素的字体大小
ex	相对于字符x的高度。此高度通常为字体尺寸的一半
px	像素,是屏幕上显示数据的最基本的点
rem	相对单位,相对html标签,常用于HTML5页面自适应

2. 颜色

在 HTML5 页面开发过程中经常涉及颜色设置，例如字体颜色、背景色等，颜色的设置可以使用表 3-3 中的方式。

表 3-3 CSS 颜色设置方式及说明

方 式	描 述
预定义颜色名	例如 red、black、blue 等
rgb(x,x,x)	红绿蓝值，例如 rgb(255,234,244)
rgba(x,x,x,a)	红绿蓝透明度值，例如 rgba(255,234,244,0.5)
#rrggbb	十六进制数，例如 #ff0000
HSL	色调(Hue)、饱和度(Saturation)、亮度(Lightness)三个颜色通道的改变以及它们相互之间的叠加来获得各种颜色，Hue 取值范围为 0～360，0(或 360)表示红色，120 表示绿色，240 表示蓝色，也可取其他数值来指定颜色。Saturation(饱和度)取值为：0～100.0%。Lightness(亮度)取值为：0～100.0%，例如 hsl(120,65%,75%)
HSLA	HSL 颜色值的扩展，带有一个 Alph 通道——它规定了对象的不透明度。例如 hsla(120,65%,75%,0.3)

3.2.3 在 HTML 文档中应用 CSS

要想使用 CSS 修饰页面，就需要在 HTML 页面中引入 CSS 样式表。引入 CSS 样式的常用方式有 3 种，具体如下。

1. 内联样式

内联样式是指通过 HTML 标签的 style 属性来设置标签的样式，示例如下：

```
<div style = "color:red;font - size: 14px;">HTML5 App 开发</div>
```

2. 内嵌样式

内嵌样式是指将在 HTML5 文档的 head 标签体中增加一个< style ></style >标签，将 CSS 设置集中在 style 的标签体中定义，基本的语法格式如下：

```
<head>
    <style>
        selector {property1: value1;property2: value2;…}
    </style>
</head>
```

3. 链接样式

链接样式是指将 CSS 样式定义在一个或多个以 CSS 为扩展名的外部样式文件中，通过 head 标签体中使用 link 标签将外部样式表文件链接到 HTML5 页面中，这也是页面样式复用经常会用到的方式，基本的语法格式如下：

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

在 HBuilderX 中书写时,只需键入快捷键“link”,在之后出现的提示框中选中,再按下 Enter 键,IDE 会自动补全 link 标签以及相应的属性。

3.3 CSS 选择器

要想将 CSS 样式应用于特定的 HTML 标签,首先需要找到该目标标签。在 CSS 中,执行这一任务的样式规则称为**选择器**。除了内联样式,内嵌样式和链接样式都需要设计选择器。下面具体介绍这些选择器。

3.3.1 基础选择器

1. 标签选择器

标签选择器指的是用 HTML 的标签名作为选择器,所有标签名都可以作为标签选择器使用,它用于为页面中某一种标签指定统一的 CSS 样式,但这也是缺点,不能实现同一种标签设计的差异化。它的语法示例为:

```
p{font-size:12px;color:red}
```

这就为页面中的所有段落的文字设计了样式:字体大小为 12 像素,颜色为红色。

2. id 选择器

id 选择器使用“#”进行标识,后面紧跟 HTML 标签的 id 属性值,一张页面中的 HTML 标签的 id 属性值是唯一的,所以这种选择器设计的样式只能针对 HTML 页面中某一个具体的标签。例如下面的样式:

```
#mydiv{font-size:12px;color:red}
```

页面启动后,该样式会自动应用到下面的标签元素上:

```
<div id="mydiv">
    HTML5 App 开发
</div>
```

3. 类选择器

类选择器使用“.”(英文点)进行标识,后面紧跟 HTML 标签的 class 属性值。它最大的优势是可以为具有相同 class 属性的 HTML 标签设置相同的样式。例如下面的样式:

```
.myclass{font-size:12px;color:red}
```

页面启动后，该样式会自动应用到 HTML 页面中 class 属性为 myclass 的所有 HTML 标签上，例如下面的 HTML 代码中，div 和 p 标签的 class 属性都是 myclass，它们内部的文字大小都是 12 像素，颜色为红色。

```
<div class = "myclass">HTML5 App 开发</div>
<p class = "myclass">HTML5 已经于 2014 年 10 月正式定稿</p>
```

4. 限定式选择器

限定式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，中间是没有空格的，例如下面的选择器：

```
div#mydiv{…}      //为 id 属性为"mydiv"的 div 标签设计样式
p.myclass{…}      //为 class 属性为"myclass"的 p 标签设计样式
```

5. 后代选择器

后代选择器是用来选择 HTML 标签元素的后代的，其写法是把父标签的选择器写在前面，后代标签的选择器写在后面，两者之间有一个空格。例如下面的选择器：

```
div p{…}          //为 div 标签中的 p 标签设计样式
div #mydiv{…}     //为 div 标签中的 id 属性为 mydiv 的子标签设计样式
p .myclass{…}     //为 p 标签中 class 属性为 myclass 的子标签设计样式
```

6. 并集选择器

并集选择器是各个选择器通过逗号连接而成的，任何形式的选择器都可以作为并列式选择器的一部分。如果某些选择器定义的样式完全或部分相同，就可以使用并列式选择器为它们定义相同的 CSS 样式，例如：

```
/* h1 标签，id 属性为"myspan"的 span 标签，class 属性为"myclass"的标签具有相同的属性 */
h1,span#myspan,.myclass{…}
```


选择器都是可以综合使用的，可以自由组合。

7. 通配符选择器

通配符选择器用 * 号表示，它是所有选择器中作用范围最广的，能匹配页面中的所有 HTML 标签元素。

3.3.2 其他选择器

3.3.1 节所讲的选择器基本上都能满足页面设计中的常规需求，HTML5 App 开发者必须重点掌握。对于一些特殊的设计需求，还可以使用表 3-4 中的选择器。

表 3-4 其他选择器

选 择 器	例 子	描 述
element > element	div > p	选择父元素为 div 标签的 p 标签(p 标签必须是 div 标签的直接子元素)
element+element	div+p	选择紧跟在 div 标签后面的 p 标签(不是内部)
element1～element2	p～ul	选择有相同的父元素中位于 p 元素之后的所有 ul 元素
[attribute]	input[name]	选择所有包含 name 属性的 input 标签
[attribute=value]	input[name="myname"]	选择 name 属性为“myname”的 input 标签
[attribute^=value]	input[name^="my"]	选择 name 属性以“my”开头的 input 标签
[attribute$=value]	input[name$="me"]	选择 name 属性以“me”结尾的 input 标签
[attribute*=value]	input[name*="na"]	选择 name 属性包含有“na”的 input 标签
:link	a:link	选择所有未被访问的超链接
:visited	a:visited	选择所有已被访问的超链接
:active	a:active	选择所有活动链接
:hover	div:hover	选择鼠标悬停的 div 标签
:focus	input:focus	选择所有获取焦点的 input 标签
:first-letter	p:first-letter	选择 p 段落中的首字母
:first-line	p:first-line	选择 p 段落中的首行
:first-child	p:first-child	选择属于父元素的第一个子元素的< p >标签
:last-child	p:last-child	选择属于父元素的最后一个子元素的< p >标签
:before	p:before{content: "测试";}	在每个< p >标签的内容之前插入文字“测试”
:after	p:after{content: "测试";}	在每个< p >标签的内容之后插入文字“测试”
:first-of-type	div p:first-of-type	选择 div 标签里面的第一个 p 标签
:last-of-type	div p:last-of-type	选择 div 标签里面的最后一个 p 标签
:nth-child(n)	li:nth-child(2)	选择属于其父元素的第 2 个 li 标签
:nth-last-child(n)	li:nth-last-child(2)	选择属于其父元素的倒数第 2 个 li 标签
:empty	div:empty	选择没有子元素的 div 标签
:not	li:not(:last-child)	选择除去最后一个 li 元素的其他所有 li 标签

3.4 尺寸属性

为了控制各标签显示的大小,CSS 提供了一系列的尺寸属性,具体如表 3-5 所示。

表 3-5 尺寸属性

属 性	描 述
width	设置标签元素的宽度
height	设置标签元素的高度
max-width	设置标签元素的最大宽度。在内容没有达到 max-width 设定的值时,HTML 标签元素的宽度可以随内容自适应;一旦达到了,则宽度不再变化
min-width	设置标签元素的最小宽度。在内容没有达到 min-width 设定的值时,HTML 标签元素的宽度保持为 min-width 设定值;达到了,则宽度随内容自适应
max-height	设置标签元素的最大高度。在内容没有达到 max-height 设定的值时,HTML 标签元素的高度可以随内容自适应;一旦达到了,则高度不再变化
min-height	设置标签元素的最小高度。在内容没有达到 min-height 设定的值时,HTML 标签元素的高度保持为 min-height 设定值;达到了,则高度随内容自适应

3.5 文本样式属性

1. 普通文本样式

为了方便地控制页面中文字的各种属性，CSS 提供了一系列的样式属性，如表 3-6 所示。

表 3-6 文本样式属性

方 式	描 述
color	设置文字颜色
font-size	设置文字大小
font-weight	设置字体的粗细，默认 normal 标准体，bold 粗体，bolder 更粗，lighter 更细，100～900 整数(100 倍整数倍)，400 等于 normal，700 等于 bold
font-family	设置字体，可以同时指定多个，以逗号隔开，如果不支持第一个字体，则会尝试下一个，以此类推，若指定的字体没有安装时，会使用浏览器默认的字体。英文字体不需要加引号，中文字体需要，英文字体应位于中文字体之前，例如：body{font-family：Arial，“微软雅黑”}
font-style	设置为 italic 时使用斜体
letter-spacing	设置字符之间距离
word-spacing	设置英文单词之间距离
line-height	设置行间距
text-transform	capitalize 是首字母大写，uppercase 是全部大写，lowercase 是全部小写
text-decoration	underline 是下画线，overline 是上画线，line-through 是删除线
text-align	设置水平对齐方式，left 是左对齐，right 是右对齐，center 是居中
text-indent	设置首行缩进处理
text-shadow	为页面中的文本添加阴影效果
text-overflow	设置文本溢出时的处理，clip 为修剪，ellipsis 为用省略号“...”标示修剪文本

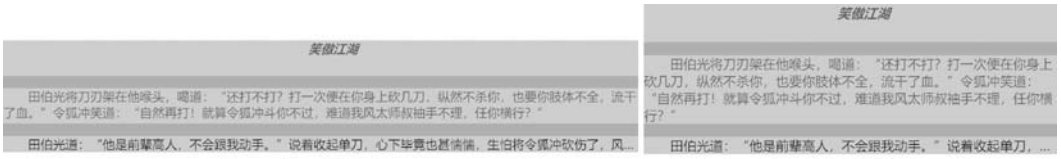
下面用一个例子示范 CSS 的尺寸属性、文本样式属性的使用。

【例 3-1】 CSS 尺寸属性、文本样式属性应用示范，代码如下：

```
<!-- 样式设计 -->
  <style>
      div,p {
           font-size: 14px;
           font-family: "微软雅黑";
       }
       .mypara,.mypara1{
              text-indent: 2em;
              background-color: #EC971F;
          }
          .mypara {
              color: #222222;
              min-height: 40px;
          }
          .mypara1 {
```

```
                white-space: nowrap;
                text-overflow: ellipsis;
                overflow: hidden;
                word-break: break-all;
            }
            #parent p:first-of-type {
                font-weight: bold;
                font-style: italic;
                text-align: center;
            }
    </style>
    <!-- HTML设计 -->
    <body>
        <p class="mypara">
            笑傲江湖
        </p>
        <p class="mypara">
    田伯光将刀刃架在他喉头,喝道:"还打不打?打一次便在你身上砍几刀,纵然不杀你,也要你肢体不全,流干了血."令狐冲笑道:"自然再打!就算令狐冲斗你不过,难道我风太师叔袖手不理,任你横行?"
        </p>
        <p class="mypara1">
    田伯光道:"他是前辈高人,不会跟我动手."说着收起单刀,心下毕竟也甚惴惴,生怕将令狐冲砍伤了,风清扬一怒出手,看来这人虽然老得很了,糟却半点不糟,神气内敛,眸子中英华隐隐,显然内功着实了得,剑术之高,那也不用说了,他也不必挥剑杀人,只须将自己逐下华山,那便糟糕之极了.
        </p>
    </body>
```

在 Chrome 中浏览该页面,效果如图 3-3 所示。



(a) 示例1　　(b) 示例2

图 3-3　CSS 尺寸和文本样式示例

例 3-1 示范了：

- 内联样式和内嵌样式的使用；
- 如何定义各种 CSS 选择器，包括 id 选择器、标签选择器、类选择器、并列选择器、后代选择器，还有 CSS3 比较有特色的第一个子标签选择器；
- 文本字体、颜色、粗细、斜体、居中的使用，中文段落首行缩进两格的使用；
- 尺寸中固定高度和宽度的设定，缩放浏览器窗口，可以看出 min-height 的使用。

页面中常见的单行实现省略号的效果，这里用到了一个 overflow 属性，控制标签中的内容超出了给定的宽度和高度自动隐藏(还可以自动生成滚动条)。

在 HBuilderX 中书写 CSS 属性时，根据不同的输入字符，HBuilderX 会有相应的智能提示，如图 3-4 所示，右边还有该属性相应的解释和示例。

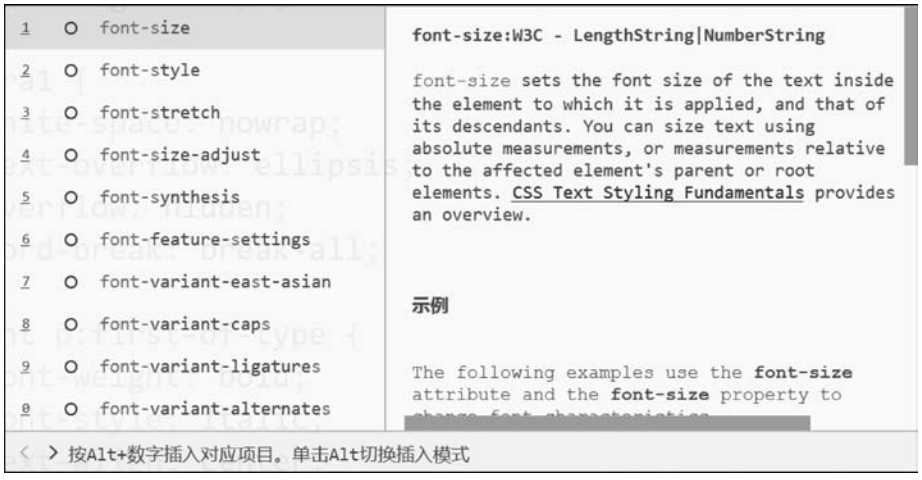


图 3-4 HBuilderX 中 CSS 的智能提示

2. 网络字体

在 CSS3 中提供了一个@font-face 规则，用来定义服务器字体，开发者可以使用任何喜欢的字体，而不用考虑客户端是否安装了相应字体。当用户在浏览该页面时，字体会自动下载到客户端(App 开发是将字体文件打包进安装包)。

@font-face 的基本语法格式为：

```
@font-face {
    font-family:"自定义字体名称";
    src: url("字体路径");
}
```

下面通过一个“毛泽东字体”的案例，来演示@font-face 的具体用法。

【例 3-2】 网络字体示例，代码如下：

```
<!-- 样式设计 -->
    @font-face {
     font-family:mzd;
     src: url("fonts/mzd.ttf");
    }
    #mypara{
     font-family: mzd;
     font-size: 0.9rem;
    }
    <!-- HTML 设计 -->
    <body>
        <p id="mypara">
            北国风光,千里冰封,万里雪飘.
        </p>
    </body>
```

页面在 Chrome 中浏览的效果如图 3-5 所示。

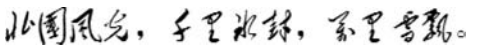


图 3-5　网络字体显示效果

网站有字库(http://www.youziku.com/)提供了中文网络字体的在线服务(只适用于网站开发，不适用于 App 和小程序开发)。

3.6　CSS 高级特性

3.6.1　继承性

所谓继承性是指书写 CSS 样式时，子标签会自动继承父标签的某些样式，例如文本的颜色和字号。利用这个特性，在设计页面时，不必在标签元素的每个子元素上再重复书写样式了，例如下面这段代码中，只是为父级元素设计了字体颜色，但它的子级元素会自动继承下来。

```
<div id = "parent" style = "color: red;">
    父及元素中的文字
    <div id = "child">
    子级元素中的文字
    </div>
</div>
```

恰当地使用继承这个特性可以简化代码，降低 CSS 样式的复杂性。例如字体、字号、颜色、行距就可以在 body 选择器中统一设置，然后通过继承影响页面中的所有文本。但是并非所有的 CSS 属性都可以被继承，例如下面的这些 CSS 属性就不具备继承性：

- 边框属性；
- 边距和填充属性；
- 背景属性；
- 定位属性；
- 尺寸属性。

3.6.2　CSS 层叠性和优先级

所谓层叠性是指对于同一个标签元素是可以设计多个 CSS 样式的，而 HTML 标签在页面上的最终显示效果是多种 CSS 样式的叠加结果，例如下面的设计，在页面上有这样一段 HTML 代码：

```
<p style = "color: red;" id = "mypara" class = "special">段落文本</p>
```

在这个 p 标签 style 属性中设计文本颜色为红色，又使用内嵌样式为它设计了如下样式：

```
<style>
    #mypara{
      font-size: 14px;
    }
    .special{
      font-family: "微软雅黑";
    }
</style>
```

最终在页面中浏览后，该段落中的文本样式是几种设计最终叠加的效果，颜色为红色，字体是“微软雅黑”，字体大小为 14px。

这里会存在问题：如果定义 CSS 样式时，出现多个相同的 CSS 属性，而不同的值应用在同一 HTML 标签元素上，它的最终效果又如何决定呢？这里需要对 CSS 样式的权重进行讲解。

CSS 权重指的是样式的优先级，有两条或多条样式作用于一个标签元素，权重高的那条样式会对标签元素起作用。当权重相同时，后写的样式会覆盖前面写的样式。

可以把样式的应用方式分为几个等级，按照等级来计算权重，如表 3-7 所示。

表 3-7 CSS 样式 4 个等级权值

选 择 器	权 重
通用选择器(*)、子选择器(>)、相邻选择器(+)、同胞选择器(~)	0
标签选择器、伪元素选择器	1
类选择器	10
id 选择器	100
style 属性	1000

对于由多个基础选择器构成的复合选择器(并集选择器除外)，其权重为这些基础选择器权重的叠加，例如下面这些 CSS 代码：

```
p span{…}                  /*权重为 1+1=2*/
P.blue{…}                  /*权重为 1+10=11*/
.blue div{…}               /*权重为 10+1=11*/
p.parent span{…}           /*权重为 1+10+1=12*/
p.parent .child{…}         /*权重为 1+10+10=21*/
#header span{…}            /*权重为 100+1=101*/
#header span.blue{…}       /*权重为 100+1+10*/
```

当 CSS 样式叠加时，页面将应用权重最高的样式，另外要注意一些特殊情况：

- 继承样式的权重为 0，也就是说子标签元素的样式会覆盖继承来的样式。
- 内联样式优先，也就是标签的 style 属性定义的样式，因为它的权重很高。
- 权重相同时，CSS 遵循的是就近原则，也就是说，后应用的样式优先级更大。
- CSS 定义了一个!important 语法，它的作用是赋予最大的优先级，也就是说，不管权重如何以及样式位置的远近，!important 都有最大的优先级，例如下面这个样式：

```
#mydiv{color:red!important;} /*不管其他样式如何设置,最终一定是红色文字*/
```

- 对两个权值进行比较的时候,是从高到低逐级将等级位上的权重值进行比较的,换句话说,低级别的权重不会进位成高级别的权重,例如,11 个 class 选择器的权重不会超过 1 个 id 选择器。

3.6.3 Chrome 调试 CSS

下面使用 Chrome 浏览页面,利用它的页面调试工具,对页面 HTML 和 CSS 进行调试,进一步掌握 CSS 的使用。

图 3-6 “检查”菜单

1. 页面 CSS 样式查看及调试

在 Chrome 中浏览例 3-1 的页面,把鼠标移到第二段文字上面后,单击鼠标右键,在弹出的菜单中选择“检查”(或按 Ctrl+Shift+I 键),如图 3-6 所示,打开 Chrome 的“开发者工具”,如图 3-7 所示。

在这个工具中可以看到,界面的 Elements 选项卡中显示出了页面的 HTML 源代码,并且鼠标选择“检查”的位置对应的 HTML 标签代码背景会以灰色显示,在图 3-7 中的右侧,从下到上显示了第二个< p >标签的 CSS 样式层叠过程,CSS 样式显示有删除线,如图 3-8 所示,表示有同样的样式设置进行了覆盖。如果 CSS 样式书写是有问题的,界面上也会有明显的提示,如图 3-9 所示。

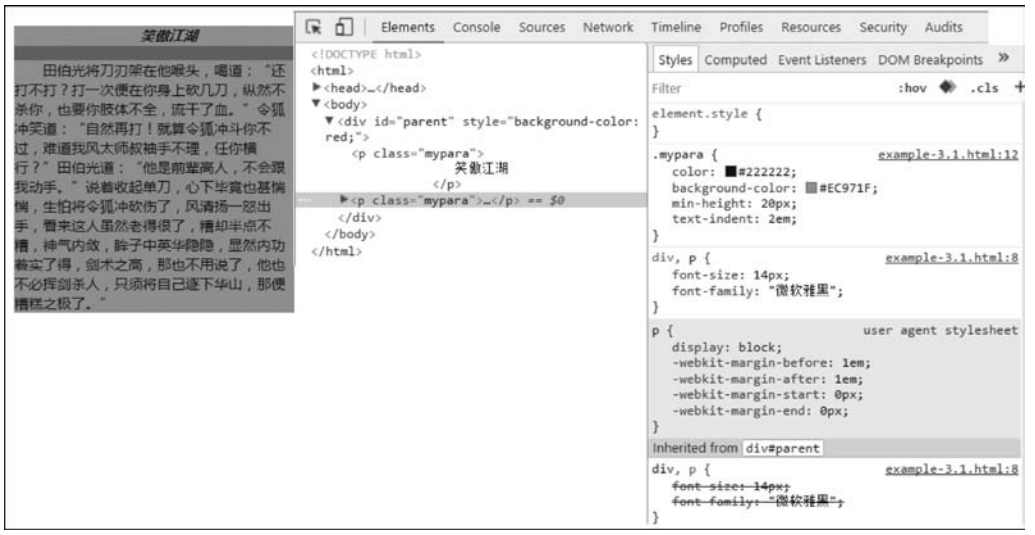


图 3-7 Chrome“开发者工具”界面

使用这样的工具,对于标签元素最终叠加出现的效果的过程一目了然,出现了问题也便于分析(例如选择器权重不够,设计的 CSS 样式应用不上或样式书写错误)。单击如图 3-10 所示的“即时 inspect 按钮”高亮后,鼠标在 HTML 页面上划过不同的 HTML 标签时,Elements 选项卡中的 HTML 代码会自动跟随切换,单击鼠标时,和前面的 CSS 观察效果一样,自动显示样式的层叠过程。

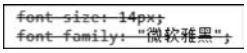


图 3-8 CSS 样式被覆盖

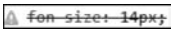


图 3-9 CSS 样式不能被识别

图 3-10 即时 inspect 按钮

在实际的开发中，经常需要不断修改 CSS 属性值，以达到最佳的视觉效果。

Chrome 提供了“即改即所得”的修改功能，如图 3-11 所示，鼠标放在某段样式设计上时，它内部设计的 CSS 属性前面都会出现复选框，单击鼠标左键，进入编辑状态，可以直接添加新的 CSS 属性，页面立刻会自动应用新的 CSS 样式。

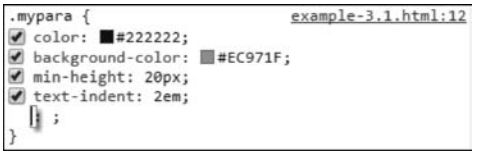


图 3-11 样式可编辑效果

也可以试着取消选中某个 CSS 样式，如图 3-12 所示，取消 text-indent 属性后，该属性被打上了删除线，页面中段落首行空两格的效果也自动消失了。

还可以用鼠标单击某个 CSS 属性的值，使其可修改，如图 3-13 所示，页面会根据修改的值立即作出响应。

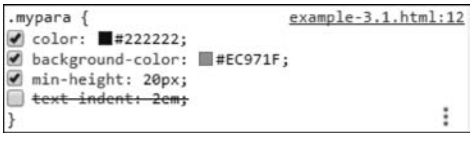


图 3-12 取消某个样式效果

图 3-13 修改某个样式效果

“即改即所得”调试效果中对 CSS 的设置和修改不会自动保存，可以记下对应的文件和行号，调试好之后在相应文件及位置进行修改再保存。

2. 模拟不同的移动设备查看页面

之所以在 HTML5 App 开发中推荐使用 Chrome 浏览器，除了它具有强大的页面调试功能，还有一项功能非常好用，也是目前移动应用开发中经常需要考虑的设备适配问题。由于移动设备的分辨率变化较大，可以使用 Chrome 的模拟功能，进行不同设备分辨率下的查看。具体方法是：打开 Chrome 的“开发者工具”后，单击左上角的“移动设备切换”按钮，如图 3-14 所示，进入移动设备查看页面。效果如图 3-15 所示。

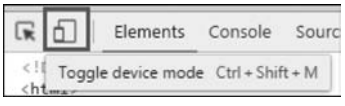


图 3-14 移动设备切换按钮

Chrome 已经内置了一些移动设备的分辨率，例如 iPhone 5、iPhone 6、iPhone 6 plus，如图 3-16 所示，你可以选择不同的设备查看页面效果。如果没有相应的设备或分辨率，还可以单击 Edit 选项进行添加。

把鼠标放在模拟器页面上时，鼠标会变成一个指尖大小的圆圈，同时触摸事件会像在手机设备上那样被触发。

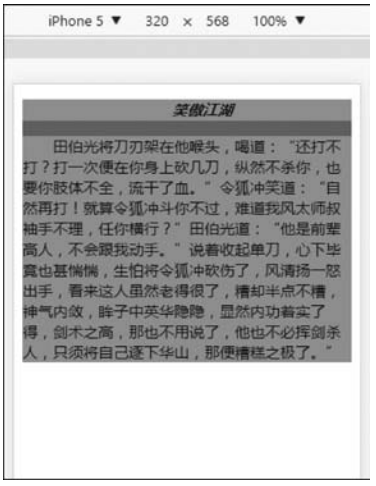


图 3-15　移动设备查看页面

图 3-16　移动设备切换菜单

3.7　背景属性

页面能通过背景图像给人留下深刻的印象，所以合理控制背景颜色和图像至关重要。本节将介绍 CSS 控制背景的一些样式属性。

3.7.1　设置背景颜色

在900万开发者的支持下，DCloud公司创造了一个小奇迹，取得中国开发者服务团队历史上未曾达到的成绩。目前HBuilder与sublime、webstorm、vscode并驾齐驱为前端界四大开发工具。

图 3-17　十六进制颜色设置

在前面的示例中已经使用过 background-color 属性来设置 HTML 标签元素的背景色，其颜色和文本颜色的取值是类似的，可以使用表 3-3 中的所有设置。例如采用十六进制的设置方式，效果如图 3-17 所示。

```
background-color: #f0ad4e;
```

在设置背景色时，还可以采用 rgba 方式来控制背景的透明度，例如：

```
background-color: rgba(240,173,78,0.5);
```

其中，前 3 个参数是颜色"#f0ad4e"的 R(红)、G(绿)、B(蓝)值，最后一个参数是透明度 Alpha 参数，它的值是 0.0(完全透明)～1.0(完全不透明)。设置的效果如图 3-18 所示。

和 RGBA 颜色设置类似，还有一个 CSS 属性 opacity 也可以用来设置透明度，它的属性值也是介于 0～1 的浮点数，和 RGBA 不同的是，它可以使任何元素呈现透明效果，它的语法如下，效果如图 3-19 所示。

```
opacity: 0.5;
```

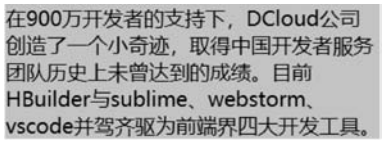


图 3-18　RGB 颜色设置

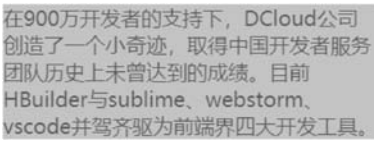


图 3-19　opacity 透明度设置

3.7.2　设置背景图片

1. 简单设置

准备一张背景图，如图 3-20 所示，使用 background 为一个 div 标签元素设置背景图像的 CSS 样式(div 标签大小任意)。

```
background: url(imgs/bk.jpg);
```

url()中代表的是图片所在的路径，可以使用相对路径(是当前 CSS 样式相对图片的路径，不是页面相对图片的路径)，也可以使用网络路径。

在 HBuilder 中，输入 url 后，会有路径的自动提示和图片预览，非常方便，如图 3-21 所示。设置背景图片时，是可以使用多张图片的，例如下面的语法：

```
background: url(img_flwr.gif), url(paper.gif);
```

图 3-20　背景图像素材

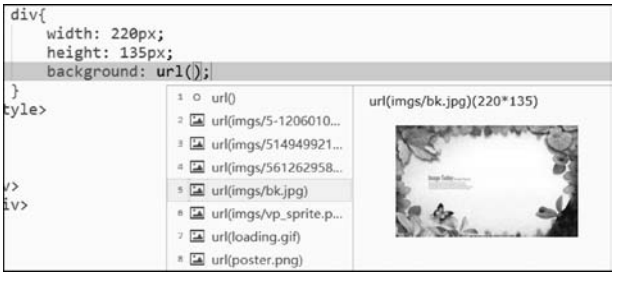


图 3-21　HBuilderX 图像路径提示

在 Chrome 中浏览后，会发现背景图片自动沿水平和垂直两个方向平铺，充满整个 div，效果如图 3-22 所示。

2. 平铺控制

如果不想让其平铺，一种方法是让容器 div 的尺寸和图片的大小一致，另一种方法就是使用下面的语法进行控制，水平平铺和垂直平铺效果如图 3-23 和图 3-24 所示。

```
background: url(imgs/bk.jpg) no-repeat;        //不允许平铺
background: url(imgs/bk.jpg) repeat-x;         //水平平铺
background: url(imgs/bk.jpg) repeat-y;         //垂直平铺
```

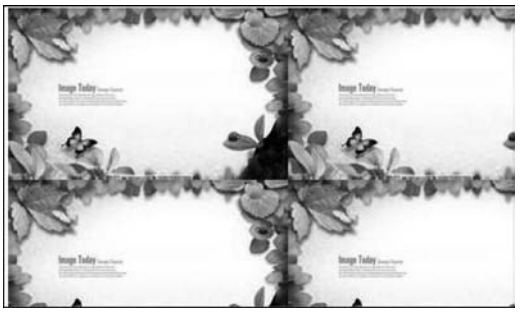


图 3-22　背景图像自动平铺

图 3-23　背景图像水平平铺

图 3-24　背景图像垂直平铺

3. 设置背景图片位置

使用背景图片时，默认情况下图片会出现在容器的左上角，如图 3-25 所示，若需要出现在其他位置，就需要使用 background-position 属性进行控制。它的值用于控制水平方向的有 left、center、right；垂直方向有 top、center、bottom。例如，想让它在右下角（见图 3-26），可以用语法：

```
background - position: right bottom;
```

图 3-25　背景图像默认在左上角

图 3-26　背景图像控制在右下角

同时使用 background 和 background-color 属性时，background-color 要写在 background 之后，或者把背景色直接写在 background 中。

4. CSS Sprite 使用

CSS Sprite 又称为“CSS 精灵”，是一种网页图片应用处理方式。它允许将一个页面涉及的所有零星图片都包含到一张大图中去，这样一来，当访问该页面时，载入的图片就不会像以前那样一幅一幅地慢慢显示出来了，它可以显著提高图片加载的效率。例如下面的图片素材，如图 3-27 所示，所有的图标都在一张图片上。那如何控制其显示哪一部分图片呢？这得借助 background-position 属性，下面用一个例子来演示 CSS Sprite 技巧。

图 3-27 CSS Sprite 图片

【例 3-3】 CSS Sprite 技巧应用示例，代码如下：

```
    <style>
        #div1,#div2,#div3{
            width: 40px;
            height: 44px;
            background: url(imgs/spritetest.png) no-repeat;
        }
        #div2{
            background-position: -40px 0;
        }
        #div3{
            background-position: -80px 0;
        }
    </style>
</head>
<body>
    <div id="div1"></div>
    <div id="div2"></div>
    <div id="div3"></div>
</body>
```

例 3-3 使用了 3 个 div 分别作为三个图标的容器，容器的大小控制为图标的大小，它们使用的都是同一张图片作为背景，只是显示的图片部分不同。这里需要使用 background-position 进行相应的坐标位置控制。坐标的计算方法是将容器和背景的图片左上角重合，考虑图片如何移动，让其显示在容器中。在 Chrome 中浏览的效果如图 3-28 所示。

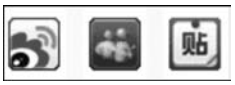


图 3-28 CSS Sprite 应用

5. 背景图片的适配

在 HTML5 App 开发中会涉及背景图片的适配问题，各种移动设备屏幕尺寸不一样，而制作不同大小的图片又不现实，这时候 CSS 属性 background-size 就可以发挥重要作用了，它的语法形式为：

```
backgournd-size:属性值 1 属性值 2;
```

在上面的语法中,background-size 可以设置一个或两个属性值来定义背景图像的宽高,其中属性值 1 必选,属性值 2 可选。属性值可以是像素、百分比、cover 或 contain,具体解释如表 3-8 所示。

表 3-8 background-size 属性值

属性值	说 明
像素值	设置背景图片的像素高度和宽度,第一个是宽度,第二个是高度。若只设一个值,则第二个值默认为 auto
百分比	以父元素的宽度和高度百分比设置背景图片的高度和宽度。只设一个值,则第二个值默认为 auto
cover	保持图像的纵横比并将图像缩放成将完全覆盖背景定位区域的最小大小
contain	保持图像的纵横比并将图像缩放成将适合背景定位区域的最大大小

下面通过一个简单的例子来进行演示。

【例 3-4】 background-size 实现背景图片的适配,代码如下:

```
<head>
    <meta charset="UTF-8">
    <title>background-size 实现背景图片的适配</title>
    <meta name="viewport" content="width=device-width,
initial-scale=1,maximum-scale=1,user-scalable=no">
    <style>
        html,body{
            margin: 0;
            padding: 0;
            width: 100%;
            height: 100%;
        }
        #mydiv{
            width: 100%;
            height: 100%;
            background: url(imgs/mobilebk.jpg) no-repeat;
            background-size: 100% 100%;
        }
    </style>
</head>
<body>
    <div id="mydiv">
    </div>
</body>
```

在 Chrome 中浏览后,打开它的“开发者工具”,切换到移动设备模式,切换不同的设备,可以看到它的效果,在不同的手机分辨率下都实现了背景的自适应,如图 3-29 所示。

如果把“background-size: 100%”修改成“background-size: contain”,则有可能出现空白区域这种情况(因为要保持纵横比),如图 3-30 所示。cover 正好相反,如果图片的比例和

容器相差很大，某些部分可能会截取掉，不会显示。

图 3-29 background-size 的适配性

图 3-30 contain 的空白情况

3.8 边框属性

使用 CSS 边框属性可以创建出效果出色的边框，并且可以应用于任何元素。它的语法形式为：

```
border: 1px solid black;
```

其中，1px 代表边框粗细，solid 代表线，图 3-31 中还有其他设置的示例，black 表示线条颜色为黑色。

对应于四条边框，还可以使用 border-top、border-bottom、border-left、border-right 进行分别控制。

我们一般使用 border:none;来消除边框。

在页面设计中，经常需要设置圆角边框，可以使用 border-radius 完成圆角化效果，它的语法形式为：

```
border-radius:左上角圆角半径 右上角圆角半径 右下角圆角半径 左下角圆角半径
```

如果 4 个值相同，可以只用一个参数，例如下面的代码，其显示效果如图 3-32 所示。

图 3-31 不同的边框

图 3-32 圆角效果

```
border: 2px solid black;
border-radius: 5px;
Border-radius:50%;        //当宽度和高度一致时,可以生成一个圆
```

3.9 CSS 动画效果

CSS3 的出现使得网页上可以增加不少动画元素,让页面变得更加生动有趣,并且更易于交互。过去这些实现是必须依赖于 Flash 或 JavaScript。CSS3 动画效果属性主要分为三类——过渡、变换、动画,在 App 和小程序开发中,如果为了保证兼容性,可以考虑加上前缀"-webkit-",下面分别进行讲解。

3.9.1 过渡

CSS 中的过渡是指 HTML 标签元素从一种样式逐渐变化到另一种样式。要实现这个效果,必须要考虑两方面内容:变换样式使用的 CSS 属性和样式变化的时长。设置过渡要用 CSS 属性 transition,它的语法形式如下:

```
transition: 属性名称 过渡时间 速度曲线 过渡延迟时间
```

其中前两个参数是必选的,后两个参数是可选的,例如下面这段 CSS 代码:

```
div{
    width: 100px;
    height: 100px;
    background-color: yellow;
    /*设置宽度和高度变化时间是2秒*/
    transition: width 2s,height 2s;
}
div:hover{
    width: 200px;
    height: 200px;
}
```

上面这段代码为一个 div 设计了一个高度和宽度变化的过渡,div:hover 决定了当鼠标悬停在 div 上时,它的宽度和高度自动增加一倍,但这个变化过程是在 2s 内缓慢完成的,实现了动画的过程。

如果所有属性变换的时间一样长,过渡属性还可以使用:

```
transition: all 2s;
```

至于速度曲线,它的属性值及说明见表 3-9 所示。

表 3-9 速度曲线取值及说明

属 性 值	说 明
linear	规定以相同速度开始至结束的过渡效果,匀速
ease	慢速开始,然后变快,然后慢速结束的过渡效果
ease-in	慢速开始(淡入)的过渡效果
ease-out	慢速结束(淡出)的过渡效果

3.9.2 2D 及 3D 变换

CSS 中的变换属性 transform 可以动态控制 HTML 标签元素,让其在页面中进行移动、缩放、倾斜、旋转,或结合过渡和动画属性产生一些新的动画效果。transform 属性既可以实现 2D 变换,也可以实现 3D 变换。

1. 2D 变换

transform 属性的 2D 变换分为平移、旋转、缩放、倾斜,这些变换操作都是以 HTML 标签元素的中心点为基准进行的,结合过渡可以作出不同的动画效果,下面是它们的语法及示例(其中虚线表示变换之前,实线表示变换之后)。

- 平移:需要使用 translate 方法,指定中心点 X 坐标和 Y 坐标的偏移量,值可以使用负数,表示反方向移动元素,元素平移后继续保留它原有的位置,它的语法形式为:

```
transform: translate(50px,100px);
//水平向右偏移 50 像素,竖直向下偏移 100 像素
```

效果如图 3-33 所示(注:虚线都表示 HTML 标签元素的原来状态)。

- 旋转:需要使用 rotate 方法,角度单位用 deg,默认为顺时针,值可以使用负数,表示逆时针旋转,它的语法形式如下,效果如图 3-34 所示。

```
transform: rotate(30deg);
//绕中心点顺时针旋转 30 度
```

- 缩放:需要使用 scale 方法,指定宽度和高度的缩放比例,比例可以是小数,语法形式如下,效果如图 3-35 所示。

```
transform: scale(0.5,0.5)
   //宽度和高度缩小一半
```

- 倾斜:需要使用 skew 方法,指定相对于 X 轴和 Y 轴的倾斜角度,语法形式如下,效果如图 3-36 所示。

```
transform: skew(10deg,10deg)
   //相对于 X 轴和 Y 轴都倾斜 10 度
```

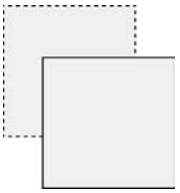


图 3-33　translate 效果

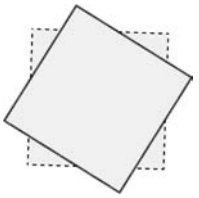


图 3-34　rotate 效果

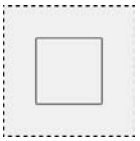


图 3-35　scale 效果

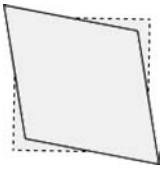


图 3-36　skew 效果

2. 3D 变换

transform 属性的 3D 变换主要是让元素分别绕 X 轴、Y 轴进行旋转。下面是它们的语法及示例。

- X 轴旋转：需要使用 rotateX 方法，指定旋转角度，单位 deg，默认为顺时针，值可以使用负数，表示逆时针旋转，语法形式如下，效果如图 3-37 所示。

```
transform: rotateX(120deg);
//绕 X 轴顺时针旋转 120 度
```

- Y 轴旋转：使用方法与 rotateX 类似，语法形式如下，效果如图 3-38 所示。

```
transform: rotateY(120deg);
//绕 Y 轴顺时针旋转 120 度
```


图 3-37　rotateX 效果

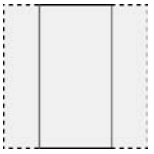


图 3-38　rotateY 效果


有时在 2D 及 3D 变换中需要同时使用两种效果，可以在 transform 属性中同时调用两种方法，类似下面的方法：

```
transform: translate(200px,0) rotateZ(280deg);
```

3.9.3　动画控制

CSS 除了支持渐变、过渡和变换特效，可以实现更强大的动画效果，它提供了一个动画控制属性 animation，可以用来设置更复杂的动画效果，例如控制动画次数、逆向动画、动画播放和暂停等。

要使用 animation 动画控制属性，首先要学会定义动画规则，动画规则的定义需要使用 @keyframes 规则，用它来定义动画中的关键帧(表示动画过程中的一个状态)，它的语法形式有两种，一种是只设置起始和终止的动画帧，另一种是使用百分比来细化动画帧(百分比

可自由定义),语法形式如下:

```
/*只设置起始和终止动画帧*/
@keyframes 动画规则名
{
    from {/*CSS属性设置*/}
     to {/*CSS属性设置*/}
}
/*以百分比方式设定帧*/
@keyframes 动画规则名
{
     0% {/*CSS属性设置*/}
     25% {/*CSS属性设置*/}
     50% {/*CSS属性设置*/}
     100% {/*CSS属性设置*/}
}
```

定义好动画帧后,就可以使用 animation 属性进行动画控制了,它的语法形式如下:

```
animation:
    name duration timing-function delay iteration-count direction;
```

- name:用@keyframes 已定义好的动画规则名。
- duration:动画花费时间。
- timing-function:动画速度曲线,取值见表 3-9。
- delay:动画延迟时间。
- iteration-count:动画播放次数。
- direction:动画逆向播放,默认值为 normal,alternate 表示动画轮流反向播放。

下面用一个常见的 CD 播放旋转效果来演示动画控制属性的使用。

【例 3-5】 动画控制的应用示例。

(1) 新建 HTML5 页面,输入代码如下:

```
    <style>
        /*样式略,请参看本书配套代码*/
    </style>

<body>
    <div id="mscPlrCtn">
        <div id="mscPlr" class="rotateCD"></div>
        <div id="mscPlrMask">
        </div>
    </div>
</body>
```

(2) 在 Chrome 中浏览,效果如图 3-39 所示。

图 3-39　初始效果

(3) 在 style 标签中添加如下动画规则和动画控制代码：

```
@keyframes CDR{
    from{
        transform: rotate(0deg);
    }
    to{
        transform: rotate(360deg);
       }
}
.rotateCD{
    animation: CDR 3s;
}
```

(4) 刷新页面后，发现 CD 会自动顺时针旋转一圈后停下。

(5) 修改选择器为".rotateCD"样式，加个动画次数，刷新页面后会发现动画自动播放了两次，CD 旋转两圈，代码如下：

```
animation: CDR 3s 2;
```

(6) 再次修改，添加逆向播放，刷新页面后发现 CD 顺时针旋转一圈，逆时针旋转一圈，代码如下：

```
animation: CDR 3s 2 alternate;
```

(7) 要想保持 CD 一直匀速旋转，需要修改如下：

```
animation: CDR 3s infinite linear;
```

动画控制中还有个很有用的属性 animation-play-state，它有两个值——paused 与 running，与 JavaScript 结合使用，可以控制动画的播放和暂停。

动画可以极大地提高界面的用户体验，但是完全基于 CSS3 的动画属性，要制作出精美的动画效果并非易事。animate.css 是一个来自国外的 CSS3 跨浏览器动画库，它预设了抖动(shake)、闪烁(flash)、弹跳(bounce)、翻转(flip)、旋转(rotateIn/rotateOut)、淡入淡出(fadeIn/

fadeOut)等多达 60 多种动画效果,几乎包含了所有常见的动画效果,可以从 https://animate.style/中下载。用户只需在加入页面后,在需要动画效果的标签元素上使用内置的 class 属性,就可实现各种动感的效果,例如下面的代码就可以实现一个文字摇晃的效果:

```
<h1 class="animate__animated animate__swing">Animate 动画库</h1>
```

3.10 其他常用的 CSS 属性

在 HTML5 App 开发中,我们还有可能会用到一些 CSS 属性,这里进行简单介绍。

- overflow:当容器的内容超过容器自身的大小时,内容溢出的显示方式,语法如下:

```
overflow: visible|hidden|auto|scroll;
```

其中,visible 表示内容不会裁剪,但会呈现在元素之外;hidden 表示溢出的内容自动隐藏;auto 表示自适应,在需要时再产生滚动条;scroll 表示始终显示滚动条,将溢出内容裁剪掉。

- list-style:设置列表项的样式,常用它消除列表项前的小黑点,语法如下:

```
list-style:none;
```

- a:link,a:visited,a:hover,a:active:分别用于设置超链接标签在被访问前、已被访问过、鼠标悬停、在被用户激活(在鼠标单击与释放之间发生的事件)时的样式,定义这几个样式时,请一定按 link、visited、hover、active 顺序定义,语法如下:

```
a:link{ text-decoration: none; /* 去除下划线 */ color: #404040;}
a:visited{ color: #00BFFF;}
a:hover{ color: red; }
a:active{ color: green; }
```

- border-collapse:设置表格显示方式,语法如下:

```
table{ border-collapse:separate|collapse };
```

其中,separate 是默认方式,也就是 table 加了 border 属性后显示的效果。对 table 标签设置 border-collapse 的效果如图 3-40 所示,使用 collapse 值后,就像 Excel 中的表格样式了。

学号	姓名	英语	数学	语文
13123401	张三	88	85	87
13123402	李四	86	95	92
13123403	王五	92	90	86
13123404	赵六	72	68	61
13123405	苟七	82	78	71

图 3-40 border-collapse 设置为 collapse

- -webkit-tap-highlight-color：用于设定元素在移动设备(如 Android、iOS)上被触发单击事件时，响应的背景框的颜色。一般设置为透明色，防止元素被单击时出现背景色。

```
-webkit-tap-highlight-color:transparent;
```

- -webkit-user-select：也是 App 开发中常用的，用于设置是否允许用户选中文本，它的设置一般为：

```
-webkit-user-select:none|text;
```

其中，none 表示不允许选中文本，text 表示可以选择。

3.11 CSS 盒子模型

CSS 中的盒子模型是页面布局的基础，只有掌握了盒子模型和各种规律及特征，才能更好地控制页面中各元素的呈现效果。所谓盒子模型(如图 3-41 所示)就是把 HTML 标签元素看作一个矩形的盒子或容器，那页面布局也就是把盒子一个个摆放或套装在不同盒子中。每个盒子都由元素的内容、内填充(padding)、边框(border)和外边距(margin)组成。宽度关系如下所示。

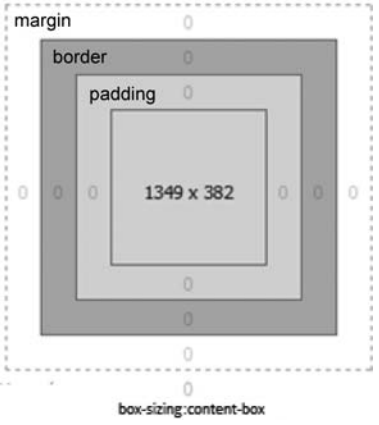


图 3-41 CSS 盒子模型示意

盒子的总宽度＝内容宽度＋左右填充＋左右外边距＋左右边框宽度

盒子的总高度＝内容高度＋上下填充＋上下外边距＋上下边框宽度

3.11.1 内填充属性

内填充属性指的是 HTML 标签元素内容与边框之间的距离。使用 CSS 属性 padding 进行设置的代码形式为：

```
padding: 上填充距离 右填充距离 下填充距离 左填充距离
```

它的参数有 4 个，除了第 1 个参数，其他 3 个是可选参数，例如对 p 标签使用 padding 属性进行设置如下，CSS 内填充效果如图 3-42 所示。

```
padding:25px 50px 75px 100px;
//上、右、下、左填充距离分别为 25px 50px 75px 100px
```

下面是 padding 的另外一些用法：

```
padding:25px 50px 75px;
//上填充距离为 25px、左右填充距离为 50px,下填充距离为 75px
padding:25px 50px;
//上下填充距离为 25px、左右填充距离为 50px
padding:25px;
//上下左右填充距离均为 25px
```

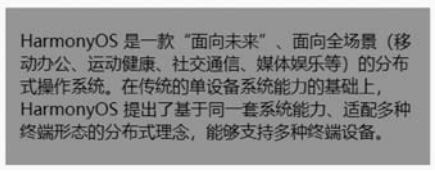


图 3-42 CSS 内填充效果

也可以使用 padding-top、padding-bottom、padding-left、padding-right 这 4 个属性分别单独设置。

3.11.2 外边距属性

从盒子模型的定义看出，页面上的内容是由很多盒子排列而成的，要想拉开盒子与盒子之间的距离，合理布局页面，就需要为盒子设置外边距。CSS 中使用的是 margin 属性，它的语法和 padding 类似，参数也是 1～4 个。

```
margin: 上外边距 右外边距 下外边距 左外边距
```

同样可以用 margin-top、margin-bottom、margin-left、margin-right 这 4 个属性进行分别单独设置。

在水平方向，两个盒子之间的距离是左边盒子的右边距与右边盒子的左边距之和，如图 3-43 所示，两个 img 标签元素之间的水平外边距离为 20px。margin 和 padding 的值还可以设置为负数，如图 3-44 所示，设为负值后，两个 img 标签元素会出现重叠效果。

图 3-43 水平边距相加

图 3-44 margin 为负值为叠加

但是在垂直方向就不一样了，当两个垂直相邻的块级元素的上下两个边距相遇时，外边距会产生重叠现象，且重叠后的外边距等于其中较大者，如图 3-45 所示。

在 Chrome 中，当打开“开发者工具”查看某个标签元素时，它的边距和填充会自动显示在工具的窗口中，如图 3-46 所示。

页面设计中，我们经常使用下面的代码来实现内容水平居中，0 表示上下外边距为 0，auto 表示左右则根据宽度自适应相同值。

```
margin:0 auto;
```

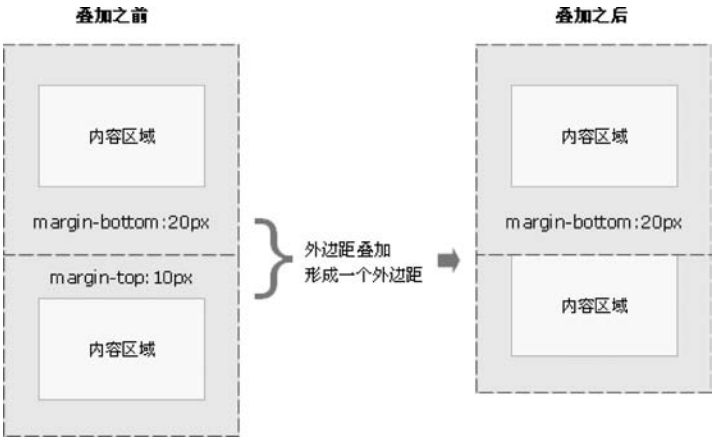


图 3-45　上下边距叠加问题

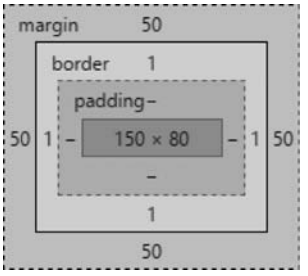


图 3-46　Chrome 边距和填充查看

由于浏览器自带的样式包括了自带的内填充和外边距，所以在 HTML5 App 开发中常用 *{margin:0;padding:0;}去除边距和填充。

3.11.3　box-sizing 属性

当一个盒子的总宽度确定之后，要想再添加边框或内填充，往往需要重新计算 width 属性值，这样才能保证盒子宽度不变，修改非常麻烦。使用 box-sizing 属性可以解决这个问题，它用于定义盒子的宽度值和高度值是否包含元素的内填充和边框，语法格式如下：

```
box-sizing:content-box|border-box;
```

content-box 表示内填充和边框不包括在宽度和高度之内，这是默认值，而 border-box 则表示内填充和边框包括在宽度和高度之内。效果如图 3-47 所示，当指定 div 的宽高为 200×120 后，使用 content-box，div 的宽度和高度明显变大，而使用 border-box 不会改变。

图 3-47　不同 box-sizing 设置的效果

3.12　浮动和定位

在默认情况下，页面中的 HTML 元素会按从上到下或从左到右的顺序将一个个盒子

罗列出来，如果按这种方式对页面排版，页面会非常单调和混乱。为了让页面版面更丰富合理，在CSS中可以对HTML标签元素实现浮动和定位。

3.12.1 浮动

1. 设置浮动

浮动属性作为CSS重要属性，在页面布局中至关重要，所谓浮动是指浮动的HTML标签元素会脱离标准文档流的控制，移动到父元素中指定位置的过程。浮动在页面开发中灵活应用，常常可以得到意想不到的效果。

可以通过float属性来设置浮动，其语法形式为：

```
float:left|right;
```

下面通过一个例子来学习float的用法。

【例 3-6】 浮动的应用示例。

(1) 实现HTML页面设计。

```
<!-- 样式设计 -->
<style>
  .container img { float:left;}
</style>
<!-- 标签设计 -->
<div class = "container">
  <img src = "imgs/hehua.jpg" class = "left_img"/>
  <p>水陆草木之花，可爱者甚蕃.晋陶渊明独爱菊.自李唐来，世人甚爱牡丹.予独爱莲之出淤泥而不染，濯清涟而不妖，中通外直，不蔓不枝，香远益清，亭亭净植，可远观而不可亵玩焉.予谓菊，花之隐逸者也；牡丹，花之富贵者也；莲，花之君子者也.噫！菊之爱，陶后鲜有闻.莲之爱，同予者何人？牡丹之爱，宜乎众矣！
  </p>
</div>
```

(2) 使用Chrome浏览效果，如图3-48所示。

(3) 在页面修改成右浮动，代码如下，再使用Chrome浏览效果，如图3-49所示。

```
.container img { float:right;}
```

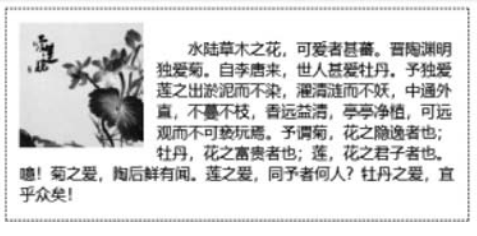


图3-48 左浮动效果

图3-49 右浮动效果

除了用来实现文字环绕效果,有时我们还会用它来实现列表的行排列,例如网易首页的栏目设计,如图 3-50 所示,就采用了 ul、li 标签实现,但 li 标签默认是按列进行排列的,所以对 li 需要采用左浮动,将栏目实现行排列。

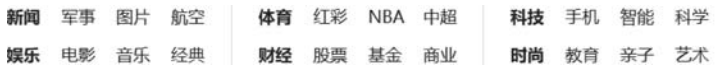


图 3-50 网易首页左浮动效果

当浮动标签元素的容器不足以容纳全部标签时,标签元素会自动换行浮动,但有可能会被其他浮动元素“卡住”,出现如图 3-51 所示的效果。

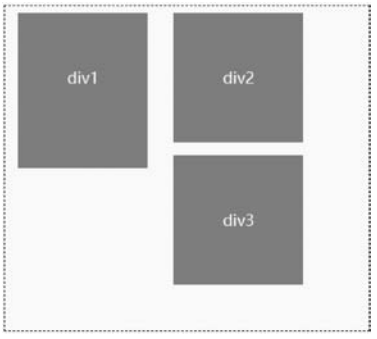


图 3-51 浮动中被卡住

2. 清除浮动

浮动有时会给页面的布局带来一定的麻烦,如图 3-52 中的左例,当使用左浮动实现文字环绕后,后续的 p 标签会自动进行填位,如果要让它单独占一行,实现布局,则要使用 CSS 中的 clear 属性,它定义了 HTML 元素的哪边(左边或右边或两边)不允许出现浮动元素。它的设置语法形式为:

```
clear: left|right|both;
```

图 3-52 清除浮动效果

3.12.2 定位

浮动布局虽然灵活,但却无法对 HTML 标签元素实现精确控制。在 CSS 中,还可以使用定位属性来进行精确定位。

1. 定位方式和偏移量

在CSS中，使用position属性设置HTML标签元素的定位方式。它的语法格式如下：

```
position: static|relative|absolute|fixed|sticky;
```

在上面的语法中，position属性的常用值有5个，分别表示不同的定位模式。定位模式仅用于定义HTML标签元素以哪种方式定位，并不能确定它的具体位置，必须通过偏移属性top、bottom、left、right的组合来精确定义元素的位置。下面我们依次来看这几种不同的定位方式。

- static：静态定位，默认值，即各HTML标签元素在HTML文档流中的默认位置。
- relative：相对定位，相对于HTML标签元素本来该在HTML文档流中显示的位置，但移动以后，它原本所占空间仍然保留，如图3-53所示，当对div1实现相对定位后，div2并不会自动向上，使用代码如下：

```
<div style="position: relative;left:60px;top:55px;">div1</div>
```

- absolute：绝对定位，依据最近已经实现过定位（相对、绝对、固定）的父元素进行定位，若所有父元素都没有定位，则根据html元素（浏览器窗口）定位，它所占空间不会保留，如图3-54所示，当div1的容器实现相对定位后，而对div1再实现绝对定位，div2会自动填补，使用代码如下：

```
<div style="position: relative;">
  <div style="position: absolute;left:60px;top:55px;">
      div1
  </div>
</div>
```

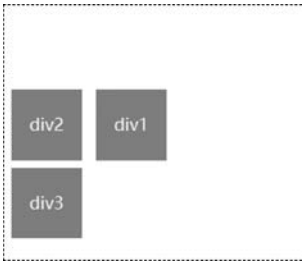


图3-53 相对定位效果

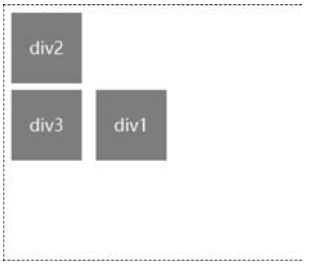


图3-54 绝对定位效果

- fixed：固定定位，以浏览器窗口作为参照物。不管浏览器滚动条如何滚动或浏览器窗口大小如何变化，始终显示在固定的位置，如图3-55所示，当缩小浏览器窗口或上下拉动滚动条时，广告信息始终能固定在浏览器的右下角，使用的代码如下：

```
#mydiv{ position: fixed; right: 0; bottom: 0;}
```

- sticky：黏性定位，可以看作是 relative 和 fixed 的结合体，当元素在屏幕内，表现为 relative，就要滚出屏幕的时候，表现为 fixed。必须指定 top、bottom、left、right 四个值中的一个才能生效。如图 3-56 所示，当滚动条下拉到“悬停菜单”位置时，这个“悬停菜单”自动会居于页面顶部，使用的代码如下：

```
#mydiv{ position: sticky; top: 0;}
```

图 3-55　固定定位效果

图 3-56　黏性定位效果

只是使用定位方式 position，没有定义任何偏移属性的值，偏移属性使用默认值 0。

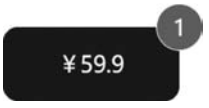


图 3-57　购物车按钮

【例 3-7】 使用相对定位和绝对定位完成购物车按钮，效果如图 3-57 所示。其中，按钮作为父容器采用了相对定位，而数量作为子元素使用绝对定位，核心代码如下所示：

```
<!-- 样式代码设计 -->
.cart{
    position: relative;
}
.quantity{
    position: absolute;
    right: -16px;
    top: -16px;
}

<!-- HTML 代码设计 -->
<div class = "cart">
    <span>¥59.9</span>
    <div class = "quantity">1</div>
</div>
```

2. z-index

当对多个元素同时设置定位时，定位元素之间有可能会出现重叠。在 CSS 中，要想调整重叠定位的顺序，可以使用 z-index 属性控制，z-index 是整数，默认值为 0，取值越大，定位元素在层叠中越在上面。如图 3-58 所示，通过控制 z-index 的值，可以控制这 3 个 div 的层叠顺序。

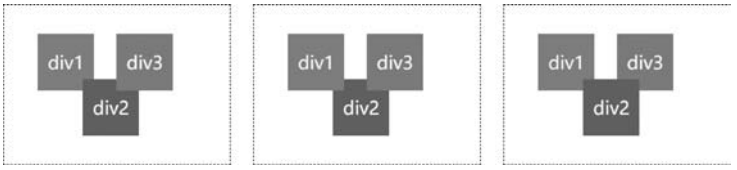


图 3-58 z-index 控制层叠顺序

3.12.3 块元素与行内元素

HTML 标签元素可以分为块元素和行内元素两种类型。块元素在页面中以区域块的形式出现，它会独自占据一行或多行，可以对其设置宽度、高度、对齐等属性，常用于页面布局和结构的搭建。常见的块元素标签有 h1～h6、div、p、ul、li 等。行内元素不一样，它不需要在新的一行出现，也不强迫其他元素在新的一行显示。一个行内元素通常都会和它前后的行内元素显示在同一行中，不占独立的区域。但一般不能设置宽度、高度、对齐等属性。常见的行内元素标签有 span、a 等。块元素和行内元素之间是可以互相转换的，效果如图 3-59 所示。主要使用 CSS 的 display 属性，它的语法形式为：

```
display: inline|block|inline-block|none;
```

- inline：将标签显示为行内元素；
- block：将标签显示为块元素；
- inline-block：将标签显示为行内块元素，可以对其设置宽高和对齐等属性，但是不会独占一行；
- none：标签元素被隐藏，不在页面上显示，也不占用页面空间。

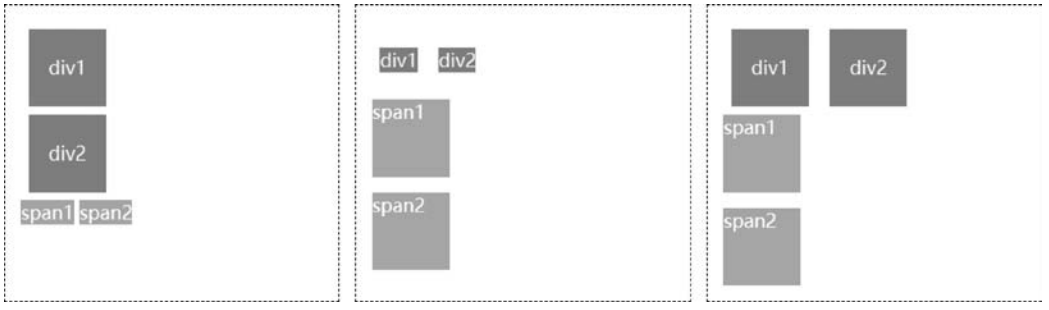


图 3-59 块元素与行内元素互相转换

3.13 响应式布局

响应式布局是指一个网站或 App 的页面能够兼容多个终端,而不是为每个终端做一个特定的版本,这个概念是为解决移动互联网浏览而诞生的。过去需要开发多套界面,通过检测判断当前访问的设备是 PC 端、平板或手机,从而请求返回不同的页面,而响应式布局开发一套界面,针对不同客户端做代码处理,来展现不同的布局和内容。下面介绍几种常用的响应式布局方案,这些方案可以结合使用。

3.13.1 viewport

先来看一个简单的例子,代码如下,在 Chrome 中使用“移动设备模式”,模拟 iPhone6 的显示效果如图 3-60 所示,会发现一个奇特的现象,iPhone 屏幕的宽度是 375 像素,而 div 的宽度设置也是 375 像素,但是并没有铺满整个屏幕。

```
<div style="width:375px;height:200px;background-color:gray;"></div>
```

这是由于屏幕的 DPR(device pixel ratio,设备像素比)不同导致的,DPR=物理像素/逻辑像素,这个倍率叫作“缩放因子”,它是移动端响应式的关键因素。物理像素是设备出厂自带的硬件像素,而逻辑像素就是我们在 CSS 中写的 px,一个逻辑像素可以代表一个或多个物理像素。DPR 是厂商给屏幕设置的一个固定值,出厂时就确定了,它的大小不会随着程序的设置而改变,例如 iPhone 6/7/8 的 DRP 是 2,而 iPhoneX 的 DPR 是 3。iPhone6 的物理分辨率为 750×1334,逻辑分辨率为 375×667,也就是在 iPhone6 上 1 个逻辑像素对应 2 个物理像素。如图 3-61 所示,假如我们设置了一个元素的 CSS 样式,也就是设计了逻辑像素,在不同的 DPR 设备上物理像素的渲染方式是不同的。

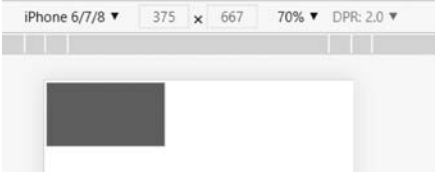


图 3-60 div 的展示效果

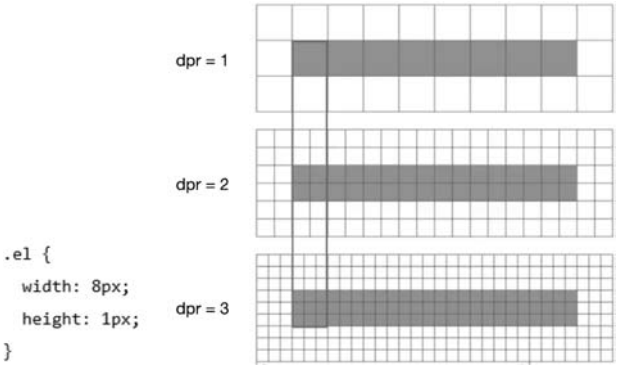


图 3-61 不同 dpr 渲染的效果

viewport 是屏幕后的一张画布,浏览器会先把页面内容绘制到画布上,然后通过屏幕窗口呈现出来。画布的宽度可大可小,默认情况下大多数设备的 viewport 宽度都是 980 像素,当画布的宽度大于屏幕宽度时,画布上的内容就无法通过屏幕全部展示出来,用户可以通过屏幕手势来拖动画布查看被遮挡的部分。可以通过在< head >标签中增加 meta 标签

来设置 viewport 属性，设置代码如下：

```
<meta name="viewport" content="width=device-width,initial-scale=1,
minimum-scale=1,maximum-scale=1,user-scalable=no" />
```

各参数的具体含义如下：

- width：控制 viewport 的大小为 device-width，也就是设备的逻辑像素宽度。
- initial-scale：页面初始缩放程度，一个浮点值，是页面大小的一个乘数。
- maximum-scale：页面最大缩放比例。
- minimum-scale：页面最小缩放比例。
- user-scalable：用户是否能改变页面缩放程度，默认值是 yes。

对于 HTML5 页面，如果没有设置 viewport，当画布宽度大于浏览器可视窗口宽度的时候，浏览器会自动对画布进行缩放，以适配浏览器可视窗口宽度。如图 3-62 所示，这是同一个页面添加 viewport 设置前后，在 iPhone6 上的对比效果。可以看出，前者整个页面宽度动态缩小以适配手机可视窗口宽度，文字严重缩小，后者字体按所设置大小正常显示。

图 3-62 添加 viewport 设置前后效果对比

3.13.2 百分比布局

HTML 中布局元素的宽度不再写成绝对值，而是以容器的宽度作为作为标准，换算成所占百分比，这样无论屏幕宽度大小如何变换，布局结构不会发生变化，如图 3-63 所示的效果，图片容器 div 的宽度为手机屏幕宽度的 44%，图片宽度也为容器宽度的 60%，无论设备宽度如何变化，都能保持两列的布局效果，示例代码如下：

图 3-63 百分比布局效果

```
.comic { margin: 10px 4%; width: 44%; }
.comic img { width: 60%; }
```

3.13.3 vw/vh 和 calc

vw、vh 是 CSS3 新增的视窗单位，同时也是相对单位，它是相对视口(viewport)大小的百分比(浏览器实际显示内容的区域，不包括工具栏、地址栏、书签栏)。vw 表示相对于视口的宽度，vh 表示相对于视口的高度，如图 3-64 所示，这种方法将 viewport 的大小宽度设置为 100vw，高度设置为 100vh，如果设置有 css 样式 width：100vw；height：100vh，则表示该元素自动全屏化。它与百分比的区别很明显，百分比是相对于父元素的，而 vw 和 vh 永远都是相对 viewport 的。

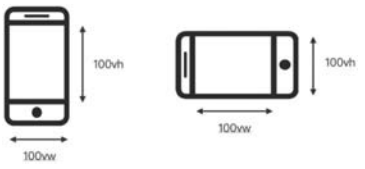


图 3-64 vw 和 vh 单位

CSS3 的 calc() 函数允许我们在 CSS 属性值中执行动态的数学计算操作，它支持“+”“-”“*”“/”运算，例如下面这个表达式，它表示该元素的宽度总是比父级容器的宽度小 50px。

```
.foo { width: calc(100% - 50px); }
```


使用 calc()函数时，表达式中若有“+”和“-”，运算符前后必须要有空格。

3.13.4 Media Queries

Media Queries 又称为媒体查询方法，它能在不同的条件下使用不同的样式，使页面在不同终端设备下达到不同的渲染效果。这里具体说一下它的使用方法，在内嵌样式或链接样式中使用它，其语法形式为：

```
@media 媒体类型 and (媒体特性){CSS 样式设置列表}
```

常用的媒体特性见表 3-10。

表 3-10 常用的媒体特性

媒体特性	说明
aspect-ratio	设备中的页面可见区域宽度与高度的比率
device-aspect-ratio	设备的屏幕可见宽度与高度的比率
device-height	设备的屏幕可见高度
device-width	设备的屏幕可见宽度
max-device-aspect-ratio | min-device-aspect-ratio	输出设备屏幕可见宽度与高度的最大(小)比率
max-device-height | min-device-height	输出设备的屏幕可见的最大(小)高度
max-device-width | min-device-width	输出设备的屏幕最大(小)可见宽度
max-resolution | max-resolution	设备的最大(小)分辨率

使用 Media Queries 必须要使用“@media”开头。媒体类型一般都使用 screen，表示用于电脑屏幕、平板、智能手机。媒体特性的书写方式和样式的书写方式非常相似，主要分为两部分，第一部分指的是媒体特性，第二部分为媒体特性所指定的值。但与 CSS 属性不同的是，媒体特性通过 min/max 来表示大于等于或小于作为逻辑判断，而不是使用小于号(<)和大于号(>)来判断。多个媒体特性使用时使用关键字“and”连接，例如下面在 iPhone6 中采用 Media Queries 写法，可以实现横屏和竖屏时，body 呈现不同背景色。

```
@media only screen and (min-device-width: 375px) and (max-device-width: 667px) and
(orientation: portrait) {
      /* iPhone 6 竖屏 */
      body{ background-color: #22BE81; }
}

@media only screen and (min-device-width: 375px) and (max-device-width: 667px) and
(orientation : landscape) {
       /* iPhone 6 横屏 */
      body{ background-color: lightcoral; }
}
```

only 用来指定某种特定的媒体类型，可以用来排除不支持媒体查询的浏览器。其实 only 很多时候是用来对那些不支持 Media Queries 的设备隐藏样式表的。

3.13.5 rem 布局

所谓 rem 布局，是指以 html 标签元素定义 font-size 为基础，例如在 iPhone6 中，将 html 标签的字体大小设置为 100px：

```
html{ font-size:100px; }
```

那么将有 1rem=100px，1.1rem=110px，所有的大小、边距等都以 rem 作为单位，CSS 设置时需要换算成 rem 设置，例如：

```
#div1{ margin-top:0.05rem; } //上外边距为 5px,使用 0.05rem
```

这样结合 Media Queries 或 JavaScript 技术，将 html 中的 font-size 根据不同的移动设备进行动态设置，就可以达到适配的要求。

HBuilderX 中已内置了 px 自动转 rem 提示功能，需要自行配置，如图 3-65 所示，可以打开菜单“工具”，选择“设置”，再切换到“编辑器设置”进行配置，设置完成后，在编辑器中输入 px 单位后，会进行 px 转 rem 的自动提示。

设计师一般会把 iPhone6(750px)作为设计稿，设计稿中的元素也都是基于 750px 进行标注的(这里的 px 指的是物理像素)。开发人员拿到设计稿后，根据 iPhone6 的 DPR 把标

图 3-65　HBuilderX 中 px 转 rem 配置和效果

注中的元素大小换算成 CSS 中的大小(逻辑像素),如设计稿中按钮的宽度标注为 40px,则 CSS 中应该写成 40/2＝20px,然后再根据屏幕的逻辑宽度进行同步缩放(如 rem/vw 方案),实现向上或向下适配所有设备。

3.13.6　Flex 布局

传统的布局解决方案主要是基于盒子模型(Box Model),它对于那些特殊布局非常不方便,例如,垂直居中就不容易实现。Flex 布局是目前推荐的最佳解决方案,可以简便、完整、响应式地实现各种页面布局。目前,它已经得到了所有浏览器的支持,特别是在微信小程序中推荐使用这种布局方式。Flex 布局的结构如图 3-66 所示。

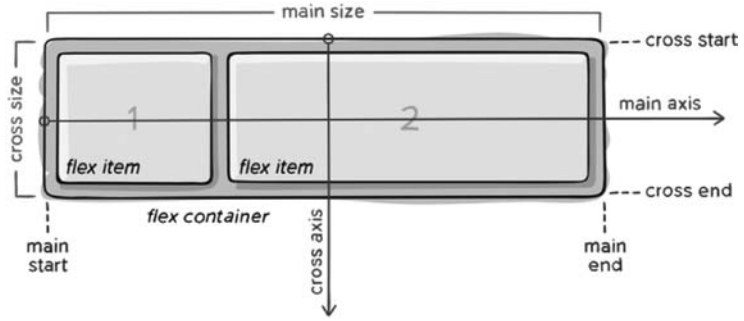


图 3-66　Flex 布局结构

采用 Flex 布局的元素称为 Flex 容器(flex container),简称“容器”。它的所有子元素自动成为容器成员,称为 Flex 项目(flex item),简称“项目”。容器默认存在两根轴:水平的主轴(main axis)和垂直的交叉轴(cross axis)。主轴的开始位置(与边框的交叉点)叫作 main start,结束位置叫作 main end;交叉轴的开始位置叫作 cross start,结束位置叫作 cross end。项目默认沿主轴排列。单个项目占据的主轴空间叫作 main size,占据的交叉轴空间叫作 cross size。

和弹性布局相关的 CSS 样式共有 13 个,简称“十三太保”属性,其中,和容器相关的有 7 个属性,和项目相关的有 6 个属性,但 IE 以及低端安卓机(版本在 4.3 以下)是不兼容的,为保证兼容性,最好加-webkit-前缀。下面对这些属性分别进行讲解。

1. 容器相关属性

- display:要想实现弹性布局,必须对容器进行 CSS 属性指定。任何一个元素(包括行内元素)都可以指定为弹性布局,代码如下:

```
display: flex;
display: inline-flex; /* 针对行内元素 */
```

容器设置为弹性布局后，项目的 float、clear 等属性会自动失效。

- flex-direction：row | row-revers | column | column-reverse，决定主轴的方向(项目的排列方向)，图 3-67 示意了这个属性取不同值后的效果(箭头表示排列方向)。
- flex-wrap：nowrap | wrap | wrap-reverse，弹性布局中，所有项目默认排在轴线上(主轴或交叉轴)，当容器的宽度和高度不足以容纳所有项目时，该属性定义如何换行，图 3-68 示意了这个属性的效果。

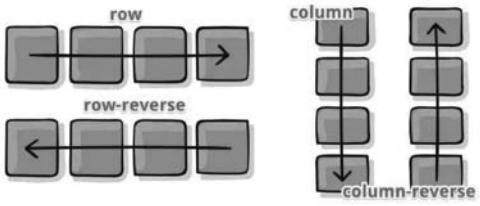


图 3-67 flex-direction 用法示意

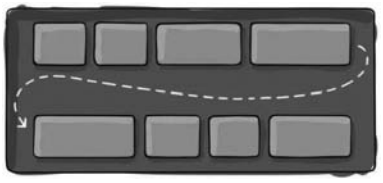


图 3-68 flex-wrap 用法示意

- flex-flow：是 flex-direction 属性和 flex-wrap 属性的简写形式，默认值为 row nowrap。
- justify-content：flex-start | flex-end | center | space-between | space-around | space-evenly，定义项目在主轴上的对齐方式，图 3-69 示意了这个属性的用法。

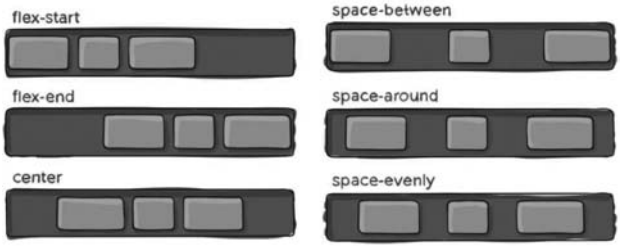


图 3-69 justify-content 用法示意

- align-items：flex-start | flex-end | center | baseline| stretch，定义一行项目在交叉轴上的对齐方式，图 3-70 示意了这个属性的效果。

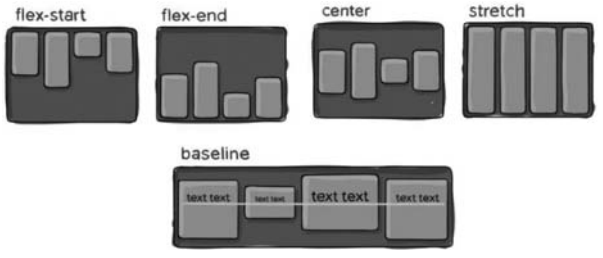


图 3-70 align-items 用法示意

- align-content：flex-start | flex-end | center | space-between | space-around | stretch，定义了多行项目在交叉轴上的对齐方式，图 3-71 示意了这个属性的效果。

2. 项目相关属性

- order：定义项目的排列顺序，值越小，排列越靠前，默认值为 0，图 3-72 示意了这个属性的效果。

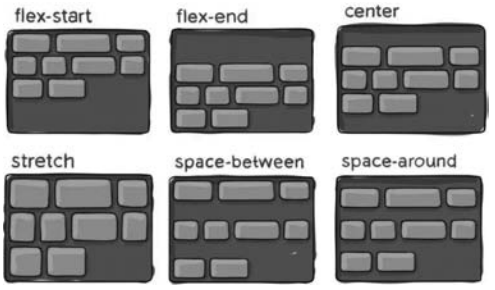


图 3-71　align-content 用法示意

图 3-72　order 用法示意

- flex-grow：定义当容器过大时，项目的放大比例，默认值为 0，图 3-73 示意了这个属性的效果。

图 3-73　flex-grow 用法示意

- flex-shrink：定义当容器过小时，项目的缩小比例，默认为 1，负值无效。当空间不足，都将等比例缩小，如果某个项目的该属性设置为 0，其他项目为 1，空间不足时，它不参与缩小。
- flex-basis：定义了在分配多余空间前，项目所占主轴空间。它和 width 有一定的关系，大多数情况和 width 表现一致，如果两者同时使用，则 flex-basis 优先级会高些，浏览器根据这个属性值，计算主轴剩余空间，默认为 auto(即项目的实际大小)。
- flex：是 flex-grow、flex-shrink 和 flex-basis 的简写，默认值为 0 1 auto，后两个属性可选，该属性有两个快捷值：auto (1 1 auto) 和 none (0 0 auto)。
- align-self：auto | flex-start | flex-end | center | baseline | stretch，设置单个项目在交叉轴上的对齐方式，图 3-74 示意了它的效果。

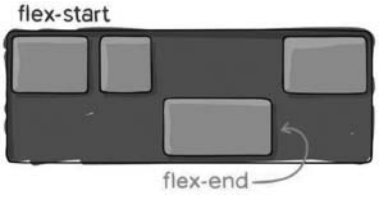


图 3-74　align-self 用法示意

网址 http://flexboxfroggy.com/提供了一个有趣的小游戏，这是一个引导式学习 Flex 布局的游戏，如图 3-75 所示，使用 Flex 布局相关的 CSS 属性让青蛙跳到荷叶上就算过关。游戏里面几乎包含了所有常用的 Flex 布局属性，可以使用它来练习 Flex 布局。

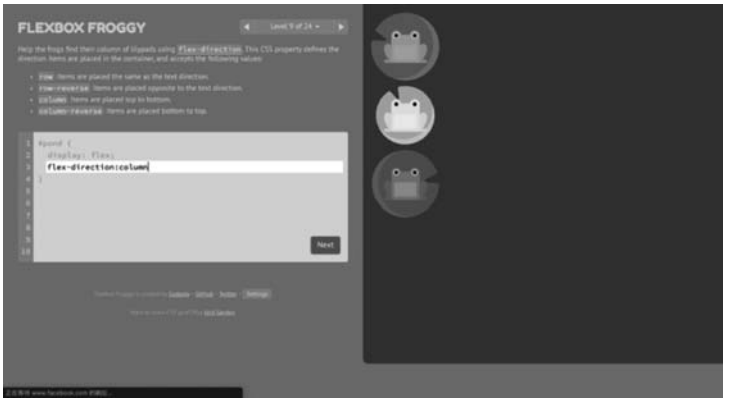


图 3-75 FlexBox Froggy 效果

3.14 实战演练

3.14.1 注册界面样式美化

【例 3-8】 注册界面样式美化。

例 2-8 完成了一个简单的表单注册页面,但是界面看起来比较简陋。在这里,我们利用学过的 CSS 属性,对界面的表单进行了样式美化。

由于 CSS 代码较多,就不在此处赘述了,请参考本书的配套源代码,结合 Chrome 的"开发者工具"学习,在此基础上可以设计出更漂亮的注册界面。本例的运行效果如图 3-76 所示。

图 3-76 注册界面美化效果

3.14.2 仿美团首页设计

【例 3-9】 仿美团首页设计的实现,这个页面实现了响应式布局,大量使用了 Flex 布局,另外还用到了 rem 布局和 Media Queries,效果如图 3-77 所示。请用手机扫描二维码,结合本书的配套源代码,参看本例的讲解。

图 3-77 仿美团首页效果

小结

本章主要学习了 CSS 样式设计页面中的使用,讲解了 CSS 样式规则,颜色和单位,如何在页面中应用 CSS,CSS 的各种选择器,CSS 的一些基本属性的使用,CSS 层叠性和优先级。另外和页面布局相关的盒子模型、浮动与定位、响应式布局等都做了详细讲解。限于篇幅,本章无法对所有的 CSS 属性都作具体的讲解,请自行参考其他的书籍和资料。

习题

一、选择题

1. 下面(　　)项是 CSS 的正确语法。

A. body:color=black　　B. {body:color:black;}

C. body{color:red;}　　D. body{color=black;}

2. 以下用于给页面所有 h1 标签添加背景色的是(　　)。

A. h1{background-color:"red";}　　B. #h1{background-color:"red";}

C. h1 all{background-color:"red";}　　D. .h1{background-color:"red";}

3. 下面(　　)CSS属性设置可以设置HTML元素的内填充左、上、右、下分别是10px、20px、30px、40px。

A. padding:10px 20px 30px 40px　　B. padding:20px 30px 40px 10px

C. padding:30px 40px 10px 20px　　D. padding:40px 10px 20px 30px

4. 页面上的div标签设计和为它设计的样式如下,这个div的背景色是(　　)。

```
<style>
    div{
        width: 100px;
        height: 100px;
        background-color: yellow;
    }
    #div1{
        background-color: blue!important;
    }
    .fordiv{
        background-color: green;
    }
</style>

<div style="background-color: red;" id="div1" class="fordiv"></div>
```

A. blue　　B. yellow　　C. green　　D. red

5. 下列(　　)样式设置后,行内元素可以定义宽度和高度。

A. display:inline;　　B. display:none;

C. display:block;　　D. display:inherit;

6. 要实现一张扑克牌在桌面上的翻牌动作,可以使用CSS变换中的(　　)方法。

A. rotate()　　B. rotateX()　　C. rotateY()　　D. skew()

7. a:hover表示超链接在鼠标(　　)的状态。

A. 按下去　　B. 经过　　C. 悬停　　D. 访问后

8. 在Webkit核心的浏览器中,设置transform属性需要添加私有前缀(　　)。

A. -ms-　　B. -o-　　C. -webkit-　　D. -moz-

二、判断题

1. 相对定位中,HTML标签元素原来所占有的空间会被保留。(　　)

2. 内填充和外边距的值不能为负值。(　　)

3. div是行内元素,span是块级元素。(　　)

4. z-index可用于设置HTML元素的重叠定位顺序,值越大的越在上面。(　　)

5. 过渡属性transition不能设置动画逆向。(　　)

6. 父级元素的position属性可以被子元素继承。(　　)

7. 在Flex布局中,align-content用于单行元素,align-items用于多行元素。(　　)

8. 使用CSS3的calc()函数时,表达式中的“+”和“-”前后必须要有空格。(　　)

三、填空题

1. 在页面中使用 CSS 有 3 种方式,分别是_____、_____、_____。

2. CSS 的 id 选择器要在定义的前面使用符号_____,class 选择器使用符号_____。

3. HTML 标签元素的绝对定位是相对于_____实现定位的。

4. 向左浮动可以使用 CSS 属性_____,元素两边都不允许有浮动,应该书写 CSS 属性为_____。

5. 为 HTML 标签元素设置边框圆角需要使用 CSS 属性_____。

6. 使用网络字体应该使用 CSS 规则_____。

7. 为 HTML 标签元素设置弹性布局需要使用 CSS 属性_____。

四、简答题

1. CSS 的层叠性是如何体现的?什么是优先级?

2. 简述 CSS Sprite 的原理及使用技巧。

3. 实现响应式布局的方法有哪些?

五、编程题

完成一个纯 CSS 实现的下拉菜单效果,如图 3-78 所示。

图 3-78 CSS 下拉菜单效果