

CSS(Cascading Styles Sheets,层叠样式表),是用于控制网页样式并允许将样式信息与网页内容分离的一种标记性语言。HTML 和 CSS 就是“内容”和“形式”的关系,由 HTML 组织网页内容的结构,而通过 CSS 决定页面的表现形式。

CSS 的主要用途包括两大方面:① 页面元素美化,由于 HTML 的主要功能是描述网页的结构,所以控制网页元素外观的能力很差,如无法精确调整文字大小、行距等,而且不能对多个网页元素进行统一的样式设置,只能一个一个元素地设置。使用 CSS 可实现对网页的外观进行更灵活、丰富的控制,使网页更美观;② 页面布局,利用 CSS 的盒子模型和相关属性可以将网页分栏分块,从而搭建出页面的版式。

3.1 CSS 基础



CSS 基础

CSS 样式表是由一系列样式规则组成,浏览器将这些规则应用到相应的元素上,CSS 语言实际上是一种描述 HTML 元素外观(样式)的语言。

3.1.1 CSS 的语法

CSS 代码包含很多条 CSS 样式规则。一条 CSS 样式规则由选择器(selector,也称选择符)和声明(declarations)组成,如图 3-1 所示。

网页一般由很多 HTML 元素组成,CSS 要将样式规则应用到特定的元素上,就必须先选中这些元素。选择器是为了选中网页中的特定元素,也就是告诉浏览器,这段 CSS 样式规则将应用到哪组(或哪个)元素上。

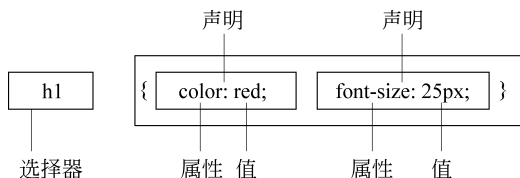


图 3-1 CSS 样式规则的组成(标记选择器)

选择器用来定义 CSS 规则的作用范围,它可以是一个标记名,表示将网页中所有该

标记的元素全部选中。图 3-1 中的 h1 就是一个标记选择器,它会将网页中所有<h1>标记的元素全部选中,选择器还可以是“.类名”或“#id 名”,它们的作用范围如图 3-2 所示。

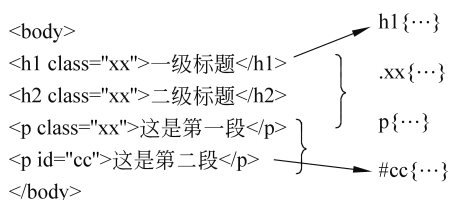


图 3-2 选择器的作用范围

而声明则用于定义选中元素的样式。介于大括号 {} 的所有内容都是声明,声明又分为属性(property)和值(value),图 3-1 中为<h1>标记的元素定义了两个属性,作用是使所有 h1 元素的文本变为红色、25 像素大小。

属性是 CSS 样式控制的核心,CSS 提供了丰富的样式属性,如颜色、大小、背景等,绝大多数 CSS 属性都是公共属性,任何 HTML 元素都可以使用。表 3-1 列出了一些常用的 CSS 属性。

表 3-1 一些常用的 CSS 属性

CSS 属性	含 义	举 例
font-size	字体大小	font-size:14px;
color	字体颜色(仅能设置字体的颜色)	color:red;
line-height	行高	line-height:160%;
text-decoration	文本修饰(如增、删下画线)	text-decoration:none;
text-indent	文本缩进	text-indent:2em;
background-color	背景颜色	background-color:#ffeeaa;

CSS 的属性和值之间用冒号隔开(注意 CSS 属性和值的写法与 HTML 属性的区别)。如果要设置多个属性,可以书写多条声明,每条声明之间用分号隔开。

对于属性值的书写,有以下规则:

- 如果属性的某个值是多个单词或者是中文,则值要用引号引起来,如 p { font-family: "sans serif" } ;
- 如果一个属性有多个值,则每个值之间要用空格隔开,如 a { padding: 6px 0 3px } ;
- 如果要为某个属性设置多个候选值,则每个值之间用逗号隔开,如 p { font-family: "Times New Roman", Times, serif } 。



引入 CSS 的方法

3.1.2 在 HTML 中引入 CSS 的方法

HTML 和 CSS 是两种作用不同的语言,它们同时对一个网页产生作用,必须通过一些方法,将 CSS 与 HTML 挂接在一起,才能正常工作。

在 HTML 中,引入 CSS 的方法有行内式、嵌入式、链接式和导入式 4 种。

1. 行内式

所有 HTML 标记都有一个通用的属性 style,行内式就是将元素的 CSS 规则作为 style 属性的属性值写在元素的标记内,例如:

```
<p style="color: red; line-height:160% ;" width="92% ">一段</p>
```

行内式引入的优点是：由于 CSS 规则就写在标记内，其作用对象就是该元素，所以无须书写 CSS 选择器。如果需要做测试或对个别元素设置 CSS 属性时，可以使用这种方式，只需要书写属性和值，但它没有体现出 CSS 统一设置许多元素样式的优势。

2. 嵌入式

嵌入式将页面中各种元素的 CSS 样式设置集中写在<style>和</style>之间，<style>标记是专用于引入嵌入式 CSS 的一个 HTML 标记，它只能放置在文档头部，即<style>...</style>只能放置在文档的<head>和</head>之间，例如：

```
<head>
<style>                <!--<style>标记用来嵌入 CSS 代码-->
h1{    color: red;
      font-size: 25px;    }
</style>
</head>
```

为单一的网页设置样式，嵌入式使用更方便，本书接下来的 CSS 代码一般都采用这种方式，但是，对于一个包含很多网页的网站来说，如果每个网页都以嵌入式的方式设置各自的样式，不仅麻烦，冗余代码多，而且网站中各个页面的风格不好统一。因此，对于一个网站来说，通常都是编写独立的 CSS 文件，使用以下两种方式之一，引入网站的所有 HTML 网页文档中。

3. 链接式和导入式

当样式需要应用于很多页面时，外部样式表（外部 CSS 文件）将是理想的选择。所谓外部样式表，就是将 CSS 代码保存成一个单独的文本文件，并将文件的后缀名命名为 .css。链接式和导入式的目的都是将外部 CSS 文件引入 HTML 文件中，其优点是可以让很多网页共享同一个 CSS 文件。

链接式是在网页头部通过<link>标记引入外部 CSS 文件，例如：

```
<link href="style1.css" rel="stylesheet" />
```

而导入式是通过 CSS 规则中的@import 指令导入外部 CSS 文件，例如：

```
<style>@ import url ("style2.css");
</style>
```

链接式和导入式最大的区别在于，链接式使用 HTML 的标记引入外部 CSS 文件，而导入式则是用 CSS 的规则引入外部 CSS 文件，因此它们的语法不同。

此外，这两种方式的显示效果也略有不同。使用链接式时，会在装载页面主体部分之前装载 CSS 文件，这样显示出来的网页一开始就带有样式效果；而使用导入式时，则在整个页面装载完之后再载入 CSS 文件，如果页面文件比较大，则开始装载时会显示无样式的页面。从浏览者的感受来说，这是使用导入式的一个缺陷。

提示：在初学 CSS 或制作单个网页时，为了方便，可采取行内式或嵌入式方法引入

CSS。但若制作网站,则主要采用链接式引入外部 CSS 文件,以便能对网站内的所有页面统一设置风格,且能通过改变一个外部 CSS 文件,改变整个网站所有页面的样式。



选择器
的分类

3.1.3 选择器的分类

选择器是为了定义 CSS 规则的作用范围,为了能够灵活选中网页中的某个或某些元素。CSS 定义了很多种选择器,其中基本的 CSS 选择器有标记选择器、类选择器、id 选择器 3 种。

1. 标记选择器

标记是元素的固有特征,标记选择器用来声明哪种标记采用哪种 CSS 样式。因此,每个 HTML 标记名都可以作为相应的标记选择器的名称。标记选择器形式如图 3-1 所示,它将具有该标记名的所有元素全部选中。

【例 3-1】 标记选择器使用示例。

```
<style>
p{                               /* 标记选择器 */
    color:blue;    font-size:18px;    }
</style>
<p>选择器之标记选择器 1</p>      <!--蓝色,18 像素大-->
<p>选择器之标记选择器 2</p>      <!--蓝色,18 像素大-->
<p>选择器之标记选择器 3</p>      <!--蓝色,18 像素大-->
<h3>h3 则不适用</h3>
```

以上所有 p 元素都会应用 p 标记选择器定义的样式,h3 元素不会受到影响。

提示: 本书对代码采用了简略写法,书中 CSS 代码均采用嵌入式方式引入 HTML 文档中。因此,读者只要将代码中的<style>...</style>部分放置在文档的<head>和</head>之间,将其他 HTML 代码放置在<body>和</body>之间,就能还原成可运行的原始代码。

2. 类选择器

标记选择器一旦声明,页面中所有该标记的元素都会产生相应的变化。例如,当声明<p>标记为红色时,页面中所有的 p 元素都将显示为红色。如果希望其中某些 p 元素不是红色,而是蓝色,就需要将这些 p 元素自定义为一类,用类选择器选中它们;或者希望不同标记的元素属于同一类,应用同一样式,如希望某些 p 元素和 h3 元素都是蓝色,则可以将这些不同标记的元素定义为同一类。

也就是说,标记选择器根据元素的固有特征(标记名)分类,好比人可以根据固有特征“肤色”分为黄种人、黑种人和白种人,而类选择器是人为地对元素分类,比如人又可以分为教师、医生、公务员等这些社会自定义的类别。

要应用类选择器,首先应给相应的元素添加一个 HTML 属性: class,为元素定义类名。如果对不同的元素定义相同的类名,那么这些元素将被划分成同一类,例如:

```
<h3 class="test">将该元素划入 test 类</h3>
<p class="test">将该元素划入 test 类</p>
```

再根据类名定义类选择器来选中该类元素,类选择器以半角“.”开头,格式如下:

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

【例 3-2】 类选择器使用示例,运行效果如图 3-3 所示。

```
<style>
.one{                /* 类选择器.one */
    background-color: red;    }    /* 背景颜色为红色 */
.two{                /* 类选择器.two */
    font-size:12px;        }        /* 字体为 12 像素大 */
</style>
<p>无类名,作为对比</p>
<p class="one">应用.one 样式</p>
<p class="two">应用.two 样式</p>
<p class="one two">同时应用.one 和.two 样式</p>
<h3 class="two">h3 也应用.two 样式</h3>
```

其中两个 p 元素和 h3 元素被定义成同一类,而第四行通过 class="one two" 将同时应用两种类选择器的样式,得到红色 12 像素的大字体。对一个元素定义多个类名是允许的,就好像一个人既属于教师又属于作家一样。第一行的 p 元素因未定义类别名则不受影响,仅作为对比。

无类名,作为对比

应用.one 样式

应用.two 样式

同时应用.one 和.two 样式

h3也应用.two 样式

图 3-3 类选择器示例

3. id 选择器

id 选择器的使用方法与类选择器基本相同。不同之处在于,一个 id 选择器只能应用于一个元素,而类选择器可应用于多个元素。id 选择器以半角“#”开头,例如:

```
#one {color: blue; font-size:18px;}
```

要应用 id 选择器定义的样式,首先必须给元素添加 id 属性,如。

【例 3-3】 id 选择器使用示例,运行效果如图 3-4 所示。

```
<style>
#one{
    font-weight:bold;    }        /* 粗体 */
#two{
    font-size:24px;        /* 字体大小 */
    background-color:#ff9;    }    /* 背景颜色 */
</style>
<p id="one">ID 值为 one</p>        <!-- 第 1 行粗体 -->
<p id="two">ID 值为 two</p>        <!-- 第 2 行字体、背景颜色改变 -->
<p id="two">ID 值为 two</p>        <!-- 错误用法 -->
<p id="one two">ID 值为 one two</p>    <!-- 错误用法 -->
```

本例中,第 1 行应用了#one 定义的样式。而第 2 行和第 3 行将同一个 id 选择器应用到两个元素上,显然违反了一个 id 选择器只能应用在一个元素上的规定,但浏览器却也应用了 CSS 样式且没有报错。

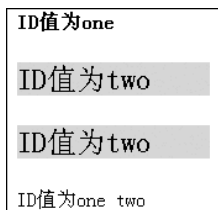


图 3-4 id 选择器示例

尽管如此,在编写 CSS 代码时,还是应该养成良好的编码习惯,一个 id 最多只能赋予一个 HTML 元素,因为每个元素定义的 id 不只是 CSS 可以调用,JavaScript 等脚本语言也可以调用,如果一个 HTML 文档中有两个相同 id 属性的元素,那么将导致 JavaScript 在查找 id 时出错(如 getElementById() 函数)。

第 4 行在浏览器中没有任何 CSS 样式风格显示,这意味着 id 选择器不支持像 class 选择器那样的多个 id 名同时使用。因为每个元素和它的 id 是一一对应的关系,不能为一个元素指定多个 id,也不能将多个元素定义为一个 id。类似 id="one two" 这样的写法是错误的。

关于类名和 id 名是否区分大小写,CSS 大体上是不区分大小写的语言,但类名和 id 名是否区分大小写取决于标记语言是否区分大小写,如果使用 HTML 5 或 XHTML 文档类型声明,那么类名和 id 名是区分大小写的。另外,id 名或类名的第一个字符不能为数字。

3.1.4 CSS 文本修饰

文本的美化是网页美观的一个基本要求。通过 CSS 强大的文本修饰功能,可以对文本样式进行更加精细的控制,其功能远比 HTML 中的标记强大。

CSS 中控制文本样式的属性主要有 font-属性类和 text-属性类,再加上修改文本颜色的 color 属性和行高 line-height 属性,DW 中这些属性的设置是放在 CSS 规则定义面板的“类型”和“区块”中的。

【例 3-4】 利用 CSS 文本属性修饰文本,其显示效果如图 3-5 所示。

```
<style>
  h1 {font-size: 16px; text-align: center; letter-spacing: 0.3em;}
  p {font-size: 14px; line-height: 160% ; text-indent: 2em;margin:0;}
  .source {color: #999; font-size:13px; text-align: right;}
</style>
<h1>思考的权利</h1>
<p class="source">2022 年 5 月 11 日 灯下随笔</p>
<p>目前家长热衷于给孩子上各种各样的培训班、补习班。</p>
<p>在学习遇到难题时,不妨出去散步,刚才理解不了的问题往往散步时突然就想明白了。</p>
```

例 3-4 中用到的 CSS 属性说明如下。

- text-indent: 首行缩进,段前空两格就是用 text-indent: 2em 实现的。
- line-height: 行距,160%表示行距为字体高度的 1.6 倍。
- letter-spacing: 设置字符间的水平间距。
- text-align: 设置文本的水平对齐方式。
- color: 设置字体颜色(只能设置字体颜色,不能设置其他颜色)。

大多数 HTML 元素在浏览器中的默认行距是单倍行距,显得过窄。因此,制作网页时一般要对 CSS 文本属性进行调整,网页中常用的字体大小有 16px 和 18px,目前网页文

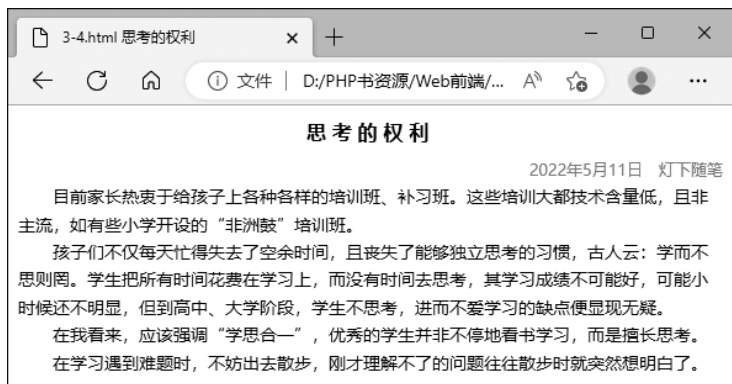


图 3-5 用 CSS 文本属性修饰文本

本样式设计的趋势是采用大字体、大行距。

如果要设置的字体属性过多,可以使用字体缩写属性:font,例如“font: 12px/1.6 Arial;”表示 12 像素字体大小、1.6 倍行距,但必须同时定义字体和字号才有效,因此这条规则中定义的字体“Arial”是不能省略的。

3.2 CSS 的特性



CSS 的特性

CSS 具有两大特性:层叠性和继承性。利用这两大特性可大大减少 CSS 代码的编写。

3.2.1 CSS 的层叠性

所谓层叠性,是指多个 CSS 选择器的作用范围发生了叠加,比如页面中某些元素同时被多个选择器选中(就好像同一案例适用于多个法律条文一样)。层叠性讨论的问题是:当有多个选择器都作用于同一元素时,CSS 该如何处理?

CSS 的处理原则是:

(1) 如果多个选择器定义的规则未发生冲突,则元素将应用所有选择器定义的样式。

【例 3-5】 选择器层叠不冲突时的样式,其显示效果如图 3-6 所示。

```
<style>
p{                                /* 标记选择器 */
    color:blue;    font-size:14px;}
.extra{                          /* 类别选择器 */
    font-weight: bold; text-decoration:underline; }
#bbl{                             /* id 选择器 */
    background-color:#FF9; }      /* 背景颜色为黄色 */
</style>
<p>标记选择器选中</p><p>标记选择器 2</p>
<p class="extra">标记选择器、类选择器均选中</p>
<p id="bbl" class="extra">标记选择器、类选择器和 id 选择器均选中</p>
```


标记选择器选中

标记选择器2

标记选择器、类选择器均选中

标记选择器、类选择器和id选择器均选中

图 3-6 选择器层叠不冲突时的样式

在代码中,所有 p 元素都被标记选择器 p 选中,同时第 3、4 个 p 元素又被类选择器.special 选中,第 4 个 p 元素还被 id 选择器 underline 选中,由于这些选择器定义的规则没有发生冲突,因此被多个选择器同时选中的第 3、4 个元素将应用多个选择器定义的样式。

(2) 如果多个选择器定义的规则发生了冲突,则 CSS 按选择器的优先级让元素应用优先级高的选择器定义的样式。CSS 规定选择器的优先级从高到低依次为

行内样式 > ID 选择器 > 类选择器 > 标记选择器

总的原则是:越特殊的样式,优先级越高。

【例 3-6】 选择器层叠冲突时的样式。

```
<style>
p{                                /* 标记选择器 */
    color:blue;                   /* 蓝色 */
    font-style: italic;           /* 斜体 */
}.green{                           /* 类选择器 */
    color:green;                  /* 绿色 */
}.purple{
    color:purple;                 /* 紫色 */
#red{                             /* id 选择器 */
    color:red;                    /* 红色 */
</style>

<p>这是第 1 行文本</p>          <!-- 蓝色,所有行都以斜体显示 -->
<p class="green">这是第 2 行文本</p>          <!-- 绿色 -->
<p class=" green" id="red">这是第 3 行文本</p>      <!-- 红色 -->
<p id="red" style="color:orange;">这是第 4 行文本</p>  <!-- 黄橙 -->
<p class="purple green">这是第 5 行文本</p>          <!-- 紫色 -->
```

由于类选择器的优先级比标记选择器的优先级高,而类选择器中定义的文字颜色规则和标记选择器中定义的发生冲突,因此被两个选择器都选中的第 2 行 p 元素将应用 .green 类选择器定义的样式,而忽略 p 选择器定义的规则,但 p 选择器定义的其他样式还是有效的。因此,第 2 行 p 元素显示为绿色斜体的文字;同理,第 3 行 p 元素将按优先级高低应用 ID 选择器的样式,显示为红色斜体;第 4 行 p 元素将应用行内样式,显示为橙色斜体;第 5 行 p 元素同时应用了两个类选择器 class="purple green",两个选择器的优先级相同,这时会以 CSS 代码中后出现的选择器(.purple)为准,显示为紫色斜体。

(3) !important 关键字。

!important 关键字用来强制提升某条声明的重要性。如果在不同选择器中定义的声明发生冲突,而且某条声明后带有 !important,则优先级规则为“!important > 行内样式 > ID 选择器 > 类选择器 > 标记选择器”。对于例 3-6,如果给 .green 选择器中的声明后添加 !important,则第三行和第五行文本都会变为绿色,在任何浏览器中都是这种效果。


```
.green{                                /* 类选择器 */
    color:green!important;    }    /* 通过!important 提升该选择器的优先级 */
```

如果在同一选择器中定义了两条相冲突的规则,那么 IE 6 总是以最后一条规则为准,不认 !important,而 Chrome/IE 7+以定义了 !important 的为准。

```
#box {
    color:red!important;            /* 除 IE 6 外的其他浏览器以这条规则为准 */
    color:blue;                    /* IE 6 总是以最后一条规则为准 */
```

!important 用法总结: ①若在同一选择器中定义的多条样式发生冲突,则 IE 6 会忽略样式后的 !important 关键字,总是以最后定义的那条样式为准;②如果在不同选择器中定义的样式发生冲突,那么所有浏览器都以 !important 样式的优先级为最高。

3.2.2 CSS 的继承性

CSS 的继承性是指如果子元素定义的样式没有和父元素定义的样式发生冲突,那么子元素将继承父元素的样式风格,并可以在父元素样式的基础上再添加新的样式,而子元素的样式风格不会影响父元素。

【例 3-7】 CSS 的继承性示例,显示效果如图 3-7 所示。

```
<style>
body {
    text-align:center; font-size:14px; text-decoration:underline; }
p {    background-color:#FF9; }    /* 背景为黄色 */
.write{    text-align: right; }
</style>
<h2>十二星座传说</h2><!--蓝色-->
<p><em>白羊座</em>的传说</p>
<p>天蝎座的传说</p>
<p class="write">作者: 莫某某</p>
```

本例中, body 标记选择器定义的文本居中, 14px 字体、带下画线等属性都被所有子元素(h2 和 p)所继承,因此前 3 行完全应用了 body 定义的样式,而且 p 元素还把它继承的样式传递给了子元素 em,但第四行的 p 元素由于通过“.write”类选择器重新定义了右对齐的样式,因此将覆盖父元素 body 的居中对齐,显示为右对齐。

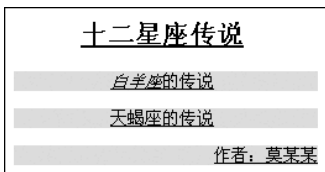


图 3-7 CSS 的继承性示例

由于浏览器对 h2 标题元素预定义了默认样式,该样式覆盖了 h2 元素继承的 body 标记选择器定义的 14px 字体样式,结果显示为 h2 元素的字体大小,粗体。可见,继承的样式比元素的浏览器默认样式的优先级还要低。如果要使 h2 元素显示为 14px 大小,需要对该元素直接定义字体大小,以覆盖浏览器默认样式。

CSS 的继承贯穿整个 CSS 设计的始终,每个标记都遵循 CSS 继承的概念。可以利用这种巧妙的继承关系,大大缩减代码的编写量,并提高可读性,尤其在页面内容很多且关系复杂的情况下。例如,如果网页中大部分文字的字体大小都是 14px,则可以对 body 元

素定义字体大小为 12 像素。这样,其他元素都会继承这一样式,就不需要对这么多的子元素分别定义样式了,有些特殊的地方如果字体大小要求是 18px,则可再利用类选择器或 id 选择器对它们单独定义。

实际上,HTML 文档是一个如图 3-8 所示的树状结构(称为文档对象模型(DOM)),因此 HTML 中的元素都存在继承关系,CSS 的继承性正是基于元素的这种继承关系。

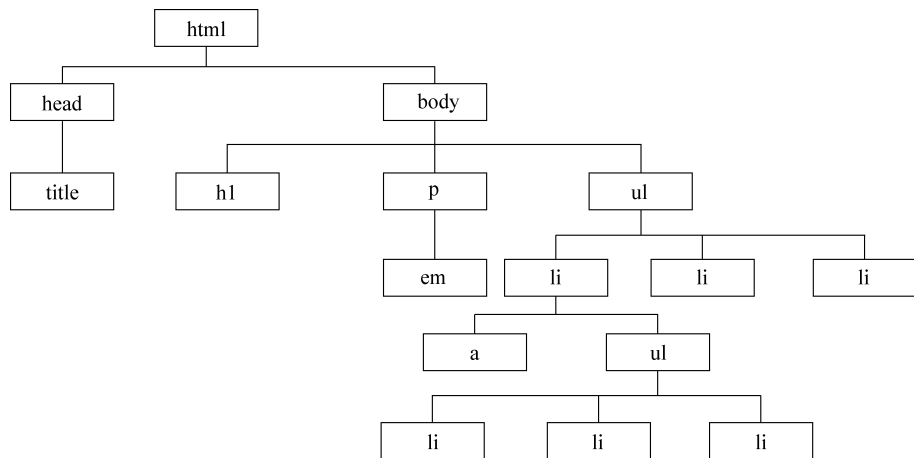


图 3-8 文档对象模型(DOM)图

需要指出,并不是所有的 CSS 属性都具有继承性,一般 CSS 的文本属性具有继承性,而其他属性(如背景属性、布局属性等)则不具有继承性。

具有继承性的属性大致有: color、font-(以 font 开头的属性)、text-indent、text-align、line-height、letter-spacing、border-collapse、opacity 等。无继承性的属性有 text-decoration,所有盒子属性(边框、边界、填充),布局属性(如 float)等。

3.2.3 选择器的组合

每个选择器都有它的作用范围,前面介绍的各种基本选择器,其作用范围都是一个单独的集合,如标记选择器的作用范围是具有该标记的所有元素的集合,类选择器的作用范围是自定义的一类元素的集合。如果希望对几种选择器的作用范围取交集、并集、子集以选中需要的元素,就要用到复合选择器了,它是通过对几种基本选择器的组合,实现更强、更方便的选择功能。

复合选择器就是两个或多个基本选择器,通过不同方式组合形成的选择器,主要有交集选择器、并集选择器和后代选择器。

1. 交集选择器

交集选择器由两个选择器直接连接构成,其结果是选中两者各自作用范围的交集。其中第一个必须是标记选择器,第二个必须是类选择器或 id 选择器。例如,h1.clas1 和 p#intro 这两个选择器之间不能有空格,格式如下。

```
h1.clas1 {color: green; font-size:24px;}
```

交集选择器将选中同时满足前后二者定义的元素,也就是前者定义的标记类型,并且指定了后者类名或 id 的元素。

【例 3-8】 交集选择器应用示例。

```
<style>
p {      color: blue;    }
.special {  color: green;  }
p.special {  color: red;   }
</style>
<p>普通段落文本</p>                                <!--蓝色-->
<h3>普通 h3 标题文本</h3>
<p class="special">指定了 special 类别的段落</p>      <!--红色-->
<h3 class="special">指定了 special 类别的 h3 标题</h3> <!--绿色-->
```

例 3-8 中,p 标记选择器选中了第一、三行文本;.special 类选择器选中了第三、四行文本,p.special 选择器选中了第三行文本,是两者的交集,用于对段落文本中的第三行进行特殊控制。第二行未被任何选择器选中,仅作对比。

2. 并集选择器

所谓并集选择器,其实就是对多个选择器进行集体声明,多个选择器之间用“,”隔开,其中每个选择器都可以是任意类型的选择器。如果某些选择器定义的样式完全相同,或者部分相同,就可以用并集选择器同时声明这些选择器完全相同或部分相同的样式。

【例 3-9】 并集选择器应用示例,其显示效果如图 3-9 所示。

```
<style>
* { text-align:center;}          /* 通配符选择器 */
h1,h2,h3,p {font-size: 14px;    background-color:#fcd;} /* 加背景色 */
h2.extra,#one {text-decoration: underline;}             /* 加下画线 */
</style>
<h1>h1 元素</h1>
<h2 class="extra">h2 元素</h2>
<h3>h3 元素</h3>
<h4 id="one">h4 元素</h4>
<p class="extra">段落 p 元素</p>
```

代码通过集体声明 h1、h2、h3、p 的样式,为选中的第 1、2、3、5 行的元素添加了背景色,然后对需要特殊设置的第 2、4 行添加了下画线。

上述代码中还使用了通配符选择器“*”,网页中的任何元素都会被通配符“*”选中。

3. 后代选择器

在 CSS 选择器中,还可以通过嵌套的方式,对内层的元素进行控制。例如,当标记被包含在<a>标记中时,就可以使用后代选择器 a b{...} 选中出现在 a 元素中的 b 元素。后代选择器的写法是把外层的标记写在前面,内层的标记写在后面,之间用空格隔开。



图 3-9 并集选择器示例

【例 3-10】 后代选择器应用示例,其显示效果如图 3-10 所示。

```
<style>
a { font-size: 16px; color: red;}
a b {background-color:#fcd;}
</style>
<b>这是 b 标记中的文字</b><br />
<a href="#">这是<b>a 标记中的 b<span>标记</span></b></a>
```

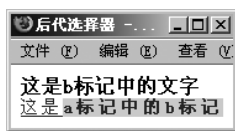


图 3-10 后代选择器示例

其中 a 元素被标记选择器 a 选中,显示为 16px 红色字体;而 a 元素中的 b 元素被后代选择器 a b 选中,背景色被定义为淡紫色;第一行的 b 元素未被任何选择器选中。

同其他 CSS 选择器一样,后代选择器定义的样式同样也能被其子元素继承。例如,在本例中,b 元素内又包含了 span 元素,那么 span 元素也将显示为淡紫色。这说明子元素(span)继承了父元素(a b)的颜色样式。

后代选择器的使用非常广泛,实际上不仅标记选择器可以用这种方式组合,类选择器和 ID 选择器也都可以进行嵌套,而且后代选择器还能进行多层嵌套。例如:

```
.special b { color : red } /* 应用了类 special 的元素里包含的<b> */
#menu li { padding : 0 6px ; } /* ID 为 menu 的元素里包含的<li> */
td.top .ban1 strong{ font-size : 16px ; } /* 多层嵌套,同样适用 */
#menu a:hover b /* ID 为 menu 的元素里的 a :hover 伪类里包含的<b> */
```

提示: 选择器的嵌套在 CSS 的编写中可大大减少对 class 或 id 的定义。因为在构建 HTML 框架时,通常只需给父元素定义 class 或 id,子元素能通过后代选择器选择的,则利用这种方式,而不需要再定义新的 class 或 id。

4. 复合选择器的优先级

复合选择器的优先级比组成它的单个选择器的优先级都要高。我们知道基本选择器的优先级是“ID 选择器 > 类选择器 > 标记选择器”,所以不妨设 ID 选择器的优先级权重是 100,类选择器的优先级权重是 10,标记选择器的优先级权重是 1,那么复合选择器的优先级就是组成它的各个选择器权重值的和。例如:

```
h1{color:red;} /* 权重=1 */
p em{color:blue;} /* 权重=2 */
.warning{color:yellow;} /* 权重=10 */
p.note em.dark{color:gray;} /* 权重=22 */
#main{color:black;} /* 权重=100 */
```

当权重值一样时,会采用“层叠原则”,一般后定义的会被应用。

【例 3-11】 复合选择器优先级计算的例子。

```
<style>
#aa ul li {color:red }
.aa {color:blue }
</style>
```

```
<div id="aa">
  <ul><li class="aa">
    CSS 常见问题之<em class="aa">复合选择器</em>的优先级
  </li>
</ul></div>
```

对于标记中的内容,它同时被“#aa ul li”和“.aa”两个选择器选中,由于#aa ul li 的优先级为 102,而.aa 的优先级为 10,所以 li 中的内容将应用#aa ul li 定义的规则,文字为红色,如果希望文字为蓝色,可提高.aa 的特殊性,将其改写成“#aa ul li.aa”。

另外,代码中 em 元素内的文字颜色为蓝色,因为直接作用于 em 元素的选择器只有“.aa”,虽然 em 也会继承“#aa ul li”选择器的样式,但是继承的样式优先级最低,会被类选择器“.aa”定义的样式所覆盖。

综上所述,CSS 样式的优先级如图 3-11 所示。

其中,浏览器对标记预定义的样式是指对于某些 HTML 标记,浏览器预先对其定义了默认的 CSS 样式,如果用户没有重新定义样式,那么浏览器将按其定义的默认样式显示。常见的 HTML 标记在标准浏览器(如 Chrome)中默认样式如下。

```
body{ margin: 8px; line-height: 1.12em }
h1 { font-size: 2em; margin: .67em 0 }
h2 { font-size: 1.5em; margin: .75em 0 }
h3 { font-size: 1.17em; margin: .83em 0 }
h4, p,blockquote, ul,fieldset, form,ol, dl, dir,menu { margin: 1.12em 0 }
h5 { font-size: .83em; margin: 1.5em 0 }
h6 { font-size: .75em; margin: 1.67em 0 }
h1, h2, h3, h4,h5, h6, b,strong { font-weight: bolder }
```

有些元素的预定义(默认)的样式在不同的浏览器中区别很大,例如 ul、ol 和 dd 等列表元素,IE 中的默认样式是 ul,ol,dd { margin-left:40px; },而 Firefox 中的默认样式定义为 ul,ol,dd { padding-left:40px; }。因此,要清除列表的默认样式,一般可以如下设置:

```
ul, ol, dd {
  list-style-type:none;      /* 清除列表项目符号 */
  margin:0;                 /* 清除 IE 左缩进 */
  padding:0;                /* 清除非 IE 左缩进 */
}
```

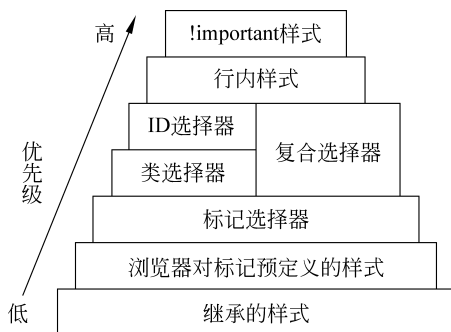


图 3-11 CSS 样式的优先级

3.3 CSS 高级选择器

为了能够更加灵活地选中网页中的各类元素,CSS 提供了各种高级选择器。

3.3.1 关系选择器

关系选择器用来选中指定元素的儿子、兄弟、后代等元素,它们能给 CSS 设计带来方便,而且对以后学习 jQuery 的选择器也是很有帮助的。

1. 子选择器

子选择器用于选中元素的直接后代(即儿子),它的定义符号是大于号(>)。

【例 3-12】 子选择器的应用示例。

```
body>p { color: green; }  
<body>  
  <p>这一段文字是绿色</p>  
  <div><p>这一段文字不是绿色</p></div>  
  <p>这一段文字是绿色</p>  
</body>
```

只有第一个和第三个段落的文字会变为绿色,因为它们是 body 元素的直接后代,所以被选中。而第二个 p 元素是 body 的间接后代,不会被选中,如果把 (body>p) 改为后代选择器 (body p),那么三个段落都会被选中。这就是子选择器和后代选择器的区别。后代选择器可选中任何后代。

2. 相邻选择器

相邻 (adjacent-sibling) 选择器的语法是“E+F”,用于选中元素 E 后面紧邻的一个兄弟(弟弟)F 元素(这两个元素具有共同的父元素,并且紧邻在一起)。

【例 3-13】 相邻选择器的应用示例。

```
h2+p { color: red; }  
<h2>下面哪些文字是红色的呢</h2>  
<p>第一段</p> <!--红色-->  
<p>第二段</p>  
<h2>下面有文字是红色的吗</h2>  
<div><p>第一段</p></div> <!--该 p 元素和 h2 不同级,不会被选中-->  
<p>第二段</p> <!--没有紧跟在 h2 后,不会被选中-->  
<h2>下面哪些文字是红色的呢</h2>  
这一段文字不是红色  
<p>第一段</p> <!--红色-->  
<p>第二段</p>
```

可见,共有两行被选中。其中,在最后一个 h2 元素后,尽管紧接的是一段文字,但那些文字不属于任何标记,因此紧随这些文字之后的第一个 p 元素也会被选中。

如果希望紧跟在 h2 后面的任何元素都变成红色,则可使用如下方法,那么第二个 h2 后的 div 元素也会被选中。

```
h2+* { color: red; }
```

3. 兄弟选择器

兄弟选择器的语法是“E~F”,用于选中元素 E 后面的所有兄弟 F 元素。

【例 3-14】 兄弟选择器的应用示例。

```
h2~p { color: red; }
<h2>下面哪些文字是红色的呢</h2>
<p>第一段</p>                <!--红色-->
<p>第二段</p>                <!--红色-->
<h2>下面有文字是红色的吗</h2>
<div><p>第一段</p></div>      <!--该 p 元素和 h2 不同级,不会被选中-->
<p>第二段</p>                <!--红色-->
<h2>下面哪些文字是红色的呢</h2>
这一段文字不是红色
<p>第一段</p>                <!--红色-->
<p>第二段</p>                <!--红色-->
```

可见,兄弟选择器选中了所有的弟弟元素,其选择范围比相邻选择器更广。

3.3.2 属性选择器

引入属性选择器后,CSS 变得更加复杂、准确、功能强大。属性选择器主要有 3 种形式,分别是:匹配属性、匹配属性和值、匹配部分属性值。属性选择器的定义方式是将属性和值写在“[]”内,“[]”前面可以加标记名、类名等基本选择器。

1. 匹配属性 E[att]

属性选择器选中具有某个指定属性的元素,例如:

```
a[name] {color:purple; }           /* 选中具有 name 属性的 a 元素 */
img[border] {border-color:gray;}   /* 选中具有 border 属性的 img 元素 */
[ special ] {color:red;}           /* 选中具有 special 属性的任何元素 */
```

这些情况下,只要给定属性在元素中出现(无论属性值是什么),便会匹配该属性选择器,还可给元素自定义一个它没有的属性名,如,那么这个 img 元素会被[special]属性选择器选中,这时属性选择器就相当于类选择器或 ID 选择器的作用了。

2. 匹配属性和值 E[att="val"]

属性选择器也可根据元素具有的属性和值匹配,例如:

```
a[href="http://www.hynu.cn"] {color:yellow; }
                                                    /* 选中指向 www.hynu.cn 的链接 */
input[type="submit"] {background:purple; }      /* 选中表单中的提交按钮 */
img[alt="Sony Logo"][class="pic"] {margin:20px;} /* 同时匹配两个属性和值 */
```

这样,用属性选择器就能很容易地选中某个特定的元素,而不用为这个特定的元素定

义一个 id 或类,再用 id 或类选择器匹配它了。

3. 匹配部分属性值

CSS 提供了 5 种匹配部分属性值的属性选择器,如表 3-2 所示。

表 3-2 匹配部分属性值的属性选择器

选 择 器	功 能
E[att~="val"]	选择 att 属性值为一用空格分隔的词列表,其中一个词是 val 的 E 元素
E[att*="val"]	选择 att 属性值中包含字符串 val 的 E 元素
E[attl="val"]	选择 att 属性值以“val-”开头或属性值为 val 的 E 元素
E[att^="val"]	选择 att 属性值为以 val 开头的 E 元素
E[att\$="val"]	选择 att 属性值为以 val 结尾的 E 元素

其中,E[att~="val"]能匹配属性值列表中的单个属性值,由于对一个元素可指定多个类名,因此匹配单个属性值的选择器就可以选中具有某个类名的元素,这才是它的主要用途。

【例 3-15】 属性选择器的应用示例。

```
h2[class~="two"] { color: red; }          /* 匹配具有类名为 two 的 h2 元素 */
h2[class*="wo"] { font-style: italic; }
h2[class$="ree"] { text-decoration: underline; }
h2[class^="on"] { background-color: #fcc; }
p[data|= "a"] { color: green; font-weight: bold; }
<h2 class="one two three">文字是红色</h2>
<p data="a-test">这一段文字是绿色</p>
```



:hover 伪类
选择器

3.3.3 伪类选择器

伪类(pseudo-class)是用来表示动态事件、状态改变或者在文档中以其他方法不能轻易实现的情况——例如用户的光标悬停或单击某元素。总的来说,伪类可以对目标元素出现某种特殊的状态应用样式。这种状态可以是光标停留在某个元素上,或者是访问一个超链接。伪类允许设计者自由指定元素在一种状态下的外观。

1. 常见的伪类选择器

常见的伪类有 4 个,分别是:link(链接)、:visited(已访问的链接)、:hover(光标悬停状态)和:active(激活状态)。其中前面两个称为**链接伪类**,只能应用于链接(a)元素,后两种称为**动态伪类**,理论上可以应用于任何元素。其他的一些伪类如:focus,表示获得焦点时的状态,一般用在表单元素上。

伪类选择器前面必须是标记名(或类名、id 名等选择器名),后面是以“:”开头的伪类名,如图 3-12 所示。

图 3-12 中的伪类选择器的作用是定义所有 a 元素在光标悬停(hover)状态下的样式。

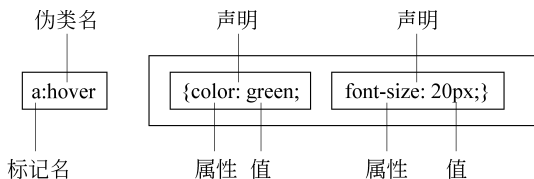


图 3-12 伪类选择器

2. 制作动态超链接

在 HTML 中,超链接默认都是统一的蓝色带下画线,被单击过的超链接则为紫色带下画线,这种传统的超链接样式看上去过于呆板。

在 CSS 中,去掉文本下画线的方法是设置 `text-decoration: none`;添加下画线则是 `text-decoration: underline`; `text-decoration` 属性的其他值还有 `line-through` (中画线), `overline` (上画线)等。

为了让网页中的超链接具有动态效果,例如,超链接初始时没有下画线,而当光标经过超链接上时,超链接会变色并添加下画线等,以提示用户可以单击。CSS 伪类选择器可实现这种动态超链接效果。

因为伪类可以描述超链接元素在各种状态下的样式,所以通过定义 `a` 元素的各种伪类具有不同的样式,就能制作出千变万化的动态超链接效果。具体来说,`a` 元素可定义的伪类有 4 种,用来描述链接的 4 种状态,如表 3-3 所示。

表 3-3 超链接标记的 4 个伪类

伪 类	作 用
<code>a:link</code>	超链接的普通样式风格,即正常浏览状态时的样式
<code>a:visited</code>	被单击过的超链接的样式风格
<code>a:hover</code>	光标悬停在超链接上时的样式风格
<code>a:active</code>	当前激活(在鼠标单击与释放之间发生)的样式风格

只要分别定义上述 4 种状态(或其中几种)的样式代码,就能实现动态超链接效果,如图 3-13 所示。

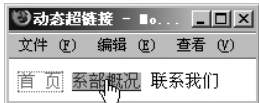


图 3-13 动态超链接

【例 3-16】 动态超链接效果,其显示效果如图 3-13 所示。

```
<style>
a {font-size: 14px; text-decoration: none;} /* 设置链接的默认状态 */
a:link {color: #666;}
a:visited {color: #000;} /* 单击后的样式 */
a:hover {color: #900; text-decoration: underline; background: #9CF;}
a:active {color: #FF3399;} </style>
<a href="#">首页</a><a href="#">系部概况</a><a href="#">联系我们</a>
```

例 3-16 中分别定义了链接在四种不同的状态下具有不同的颜色,在光标悬停时还将添加下画线并改变背景颜色。需要注意的是:

- 链接伪类选择器的书写应遵循 LVHA 的顺序,即 CSS 代码中 4 个选择器出现的顺序应为 `a:link`→`a:visited`→`a:hover`→`a:active`,若违反这种顺序,某些样式可能不起作用。
- 各种 `a` 的伪类选择器将继承 `a` 标记选择器定义的样式。
- `a:link` 选择器只能选中具有 `href` 属性的 `a` 标记,而 `a` 选择器能选中所有 `a` 标记,包括用作锚点的 `a` 标记。

3. 制作动态图片边框效果

在 CSS 2.0+规范中,任何元素都支持动态伪类,像 `li:hover`、`img:hover`、`div:hover` 和 `p:hover` 这些伪类都是合法的,并且使用伪类还能控制元素的后代或兄弟元素的动态效果。例如 `li:hover ul`(控制悬停时后代元素的样式)、`img:hover~b`(控制悬停时兄弟元素的样式)。

【例 3-17】 动态图片边框效果,如图 3-14 所示。

```
<style>
a{ display:inline-block; text-align:center; width:164px; text-decoration:
none;}
a img{ display:inline-block; padding:6px; border:1px solid transparent;}
a img:hover{border:1px solid #009;}           /* 改变边框颜色 */
a img:hover~b{ color:#F00; text-decoration:underline;}      </style>
<a href="#"><b>沙漠中的石柱</b></a>
```



沙漠中的石柱



沙漠中的石柱

图 3-14 动态图片边框效果(左为默认状态,右为光标经过时状态)

4. :first-child 伪类

`:first-child` 伪类选择器用于匹配它的父元素的第一个子元素,也就是说,这个元素是它父元素的第一个儿子,而不管它的父元素是哪个。

【例 3-18】 `:first-child` 伪类应用示例。

```
p:first-child{font-weight: bold;}
<body>
<p>这一段文字是粗体</p>           <!--第 1 行,被选中-->
<h2>下面哪些文字是粗体的呢</h2>
<p>这一段文字不是粗体</p>
```

```
<h2>下面哪些文字是粗体的呢</h2>
<div><p>这一段文字是粗体</p>           <!--第5行,被选中-->
<p>这一段文字不是粗体</p></div>
<div>下面哪些文字是粗体的呢
这一段文字不是
<p>这一段文字是粗体</p>           <!--第9行,被选中-->
<p>这一段文字不是</p></div>
</body>
```

这段文字共有3行会以粗体显示。第一行p是其父元素body的第一个儿子,被选中;第5行p是父元素div的第一个儿子,被选中,第九行p也是父元素div的第一个儿子,也被选中,尽管它前面还有一些文字,但那不是元素。

5. :focus 伪类

:focus 用于定义元素获得焦点时的样式。例如,对于一个表单来说,当光标移动到某个文本框内时(通常是单击了该文本框或使用Tab键切换到了这个文本框上),这个input元素就获得了焦点。因此,可以通过input:focus伪类选中它,改变它的背景色,使它突出显示,代码如下。

```
input:focus { background: yellow; }
```

对于不支持:focus伪类的老式浏览器,要模拟这种效果,只能使用两个事件结合JavaScript代码来模拟,它们是onfocus(获得焦点)和onblur(失去焦点)事件。

3.3.4 使用过渡属性配合动态伪类效果

CSS的动态伪类能为网页添加一些动态效果,但动态伪类没有中间状态,当一个属性的值发生变化时,这种改变是突然发生的。比如一个元素的宽度是100px,当光标悬停在上面时,宽度改变为250px,这两种状态之间并没有平滑的变化,而是在这两种状态之间发生了跳跃。为此,CSS 3引入了过渡(Transitions)模块,提供了改变这种变换方式的选项。在CSS中,过渡就是让一个属性在两种状态之间平滑改变的动画。

为了让过渡发生,必须满足4个条件:一个初始状态、一个终止状态、过渡特征(如过渡时间和过渡的属性值等)和触发器(如光标悬停)。

【例3-19】 transition 过渡属性应用示例。

```
div{ width:100px;                /* 初始状态 */
    height:40px; background:#fcc; line-height:40px;
    transition:width 0.5s;}      /* 过渡特征 */
div:hover{width:300px;}         /* 终止状态,触发器为hover */
<div>演示过渡效果</div>
```

该例产生的动画效果是:当光标滑到div元素上时,div元素的宽度会从100像素逐渐伸展到300像素,动画持续时间为0.5s。

1. 过渡属性详解

CSS 3的过渡使用transition属性定义。transition属性的基本语法如下。

```
transition: transition-property transition-duration transition-timing-  
function transition-delay;
```

transition 属性实际上是 4 个属性的简写,其各属性值的含义如下。

(1) transition-property: 指定元素的某个属性上会有动画效果。例如:

```
transition-property:width;           /* 在 width 属性上应用过渡 */  
transition-property:font-size;       /* 在 font-size 属性上应用过渡 */  
transition-property:all;             /* 在所有属性上应用过渡 */
```

(2) transition-duration 属性指定过渡从开始到结束的持续时间。其属性值是一个时间值(如 0.5s),单位是 s(秒)或者 ms(毫秒)。默认值是 0s,如果提供一个负值给该属性,则也会被解释为 0s。例如:

```
transition-duration: 2s;              /* 过渡持续时间为 2s */
```

注意: transition-duration 是创建过渡的唯一必要的属性,如果其他过渡属性都省略,但声明了 transition-duration,过渡也会发生,反过来则不行。

(3) transition-timing-function 属性:能够让动画在过渡持续期间在速度上有变化,对动画的节奏进行控制。该属性的取值有两种不同的类型:①关键字;②cubic-bezier 函数。

关键字的取值有以下 5 种。

- ease: 默认值,平稳开始和结束,即动画开始和结束时比较慢,中途比较快。
- linear: 动态效果一直都匀速进行。
- ease-in: 平稳开始,动画效果开始时比较慢,然后不断加速,直到结束。
- ease-out: 平稳结束,动画效果开始时比较快,然后不断减速,直到结束。
- ease-in-out: 类似于 ease 值,但加速、减速没有那么明显。

(4) transition-delay 属性用于延迟一段时间后再开启过渡效果,例如,希望在过渡开始前设置 0.5s 的延迟,可以使用下列代码:

```
transition-delay: 0.5s;
```

transition-delay 属性也可设置负值,考虑一个 3s 的过渡,但延迟了-1s。当该过渡被触发时,过渡会立即开始,但看上去就像已经过去了 1s 一样,也就是说,整个过渡会从中途开始。

2. transition 属性的简写

transition 属性是 4 个过渡属性的简写形式,该属性可接 2~4 个值,因为过渡的速度变化和过渡前的延迟两个属性是可以省略的。但是,不管怎么写,transition-delay 值必须写在 transition-duration 值的后面。下面是 transition 缩写的几个示例。

```
transition: color .5s ease-in .1s;    /* 作用属性持续时间速度变化延迟 */  
transition: color .5s .1s;           /* 作用属性持续时间延迟 */  
transition: color .5s;               /* 作用属性持续时间 */
```