

# HTML/CSS/JavaScript

## 目录

### 第1章 前后端

- 1.1 前端/后端工程师
- 1.2 前端技术
- 1.3 结构层/表现层/行为层

### 第2章 HTML5介绍

- 2.1 Hello World!
- 2.2 佛靠金装，人靠衣装——HTML和CSS的关系
- 2.3 标签语法
- 2.4 HTML5文档结构
- 2.5 小哥，做头吗？——head标签
- 2.6 你就是馋人家的身子！——body标签

### 第3章 语义化标签

- 3.1 语义化
- 3.2 开始我们的第一段对话吧——p标签
- 3.3 span标签
- 3.4 做个标题党——h1标签
- 3.5 div标签
- 3.6 写代码都不换行吗？——br标签
- 3.7 再加点空格呢？——特殊字符
- 3.8 再来个水平分割线——hr标签

### 第4章 列表、图片、链接

- 4.1 有序列表
- 4.2 无序列表
- 4.3 不发一张自拍吗？——img标签
- 4.4 百度一下，你就知道——a标签

### 第5章 表格、表单

- 5.1 表格标签
- 5.2 与用户交互——form标签
- 5.3 先来填用户名和密码——文本、密码输入框
- 5.4 数字、网址、邮箱输入框
- 5.5 文本域
- 5.6 label标签
- 5.7 单选框、复选框
- 5.8 下拉列表

### 第6章 CSS3介绍

- 6.1 给HTML打扮——CSS样式
- 6.2 既然那么好，那就引入CSS吧——内联式CSS
- 6.3 换个地方吧，行内太挤了——嵌入式CSS
- 6.4 还是把HTML和CSS分开吧——外部式CSS
- 6.5 总有个先来后到吧——三种链接方式的优先级

### 第7章 选择器

- 7.1 选一个标签——标签选择器
- 7.2 再选一个类——类选择器
- 7.3 取个唯一表示——ID选择器
- 7.4 捡了个儿子——子选择器
- 7.5 这么快就当爷爷了——后代选择器
- 7.6 我全都要——通配符选择器
- 7.7 给选择器分个组——分组选择器
- 7.8 伪装者——伪类选择器

7.9 别插队啊——选择器的优先级

7.10 为所欲为——选择器最高层级!important

## **第8章 字体、文本样式**

8.1 字体样式

8.2 上个色——color

8.3 文本样式

## **第9章 盒子模型与布局模型**

9.1 元素分类

9.2 盒子模型

9.3 布局模型

9.4 定位

## **第10章 JavaScript介绍**

# 第1章 前后端

## 1.1 前端/后端工程师

### 前端工程师 ( Front-End Engineer )

前端工程师互联网时代软件产品研发中不可缺少的一种专业研发角色。前端是一个相对比较新的行业，大约从2005年开始，正式的前端工程师角色被行业所认可。到了2010年，互联网开始全面进入移动时代，前端工程师的地位也越来越重要。

现在一些后端开发工作也可以由前端工程师来完成。最初所有的开发工作都是由后端工程师完成的，但是随着业务越来越繁杂、工作量过大，后端工程师们不堪重负，于是将可视化和部分交互功能的开发剥离出来，形成了前端开发。

前端工程师主要负责的工作就是使用 `HTML`、`CSS`、`JavaScript` 等专业技能和工具将产品UI设计稿实现成网站产品。涵盖用户PC端和移动端网页，处理视觉和交互问题。

广义上讲，所有用户终端产品，只要是与视觉和交互有关的部分都是前端工程师的专业领域。产品从前期开发到后期的维护、更新、升级都需要前端工程师来完成。

### 后端工程师 ( Back-End Engineer )

后端工程师主要负责服务器的数据逻辑和业务逻辑等，主要研究怎么把数据更好地传输给前端工程师。

如果一个人除了完成前端开发和后端开发工作以外，从产品设计到项目开发，再到后期运维都是同一个人，甚至可能还要负责UI、配动画、写文档等，那么就被称为全栈工程师 ( Full Stack Engineer )。



## 1.2 前端技术

---

### 应用领域

前端技术可以被应用在一系列领域中：

- 网页网站
- APP、微信小程序
- 移动端小游戏
- 炫酷的特效
- 大数据可视化
- VR虚拟现实

### 必备技能

前端工程师需要具备大量必要的技能，其中前端三大基础语言为HTML、CSS、JavaScript，除此之外还需要学习jQuery、网络、CSS3、H5、es6、webpack4.0、git、小程序、设计模式、VUE、VUEX、VUE源码、React、Node.js、Mongo DB数据库等内容。



## 1.3 结构层/表现层/行为层

### 结构层 / 表现层 / 行为层

- 结构层 HTML (HyperText Markup Language)：一个超文本标记语言，负责描绘出网页内容的架构。
- 表现层 CSS (Cascading Style Sheets)：层叠样式表，负责如何显示结构层的有关内容。
- 行为层 JavaScript：目前在Web上使用的最主要的客户端脚本语言，是Web脚本语言的一个标准，可以对结构层和表现层的内容随意进行更改。



### 代码注释

在HTML、CSS、JavaScript中代码的注释是不一样的。

- HTML

```
1 | <!-- 注释内容 -->
```

- CSS

```
1 | /* 注释内容 */
```

- JavaScript

```
1 | // 注释内容
2 | /* 注释内容 */
```

# 第2章 HTML5介绍

---

## 2.1 Hello World!

---

### Hello World!

问候一下世界，制作人生中的第一个HTML网页吧。

---

【代码】Hello World!

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>Hello world!</title>
6 </head>
7 <body>
8     Hello world!
9 </body>
10 </html>
```

## 2.2 佛靠金装，人靠衣装——HTML和CSS的关系

### HTML和CSS的关系

先来看一下单纯的HTML标签长什么样。

```
1 <p>我是一个p标签</p>
```

我是一个p标签

再来看一下经过CSS修饰过后的HTML标签：

```
1 p {  
2   color: red;  
3   border: 1px solid red;  
4   width: 140px;  
5   height: 40px;  
6 }
```

我是一个p标签

CSS是用来修饰HTML样式的。虽然HTML本身是有一些默认样式，但如果想改变HTML标签的样式，就需要借助CSS。

HTML + CSS构成了网页的基本页面结构和样式。

【代码】为"Hello World!"添加样式

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4   <meta charset="UTF-8">  
5   <title>添加样式</title>  
6   <style type="text/css">  
7     p {  
8       font-size: 12px;  
9       color: #930;  
10      text-align: center;  
11    }  
12  </style>  
13 </head>  
14 <body>  
15   <p>Hello world!</p>  
16 </body>  
17 </html>
```

## 2.3 标签语法

### 标签语法

HTML中标签的语法有以下几点：

1. 标签（tag）是由英文尖括号 `<` 和 `>` 括起来的，如 `<html>` 就是一个标签。
2. HTML中的标签一般都是成对出现的，分开始标签和结束标签。结束标签比开始标签多了一个 `/`。例如：

```
1 <p></p>
2 <div></div>
3 <span></span>
```

3. 标签与标签之间是可以嵌套的，但先后顺序必须保持一致。如，`<div>` 里嵌套 `<p>`，那么 `</p>` 必须放在 `</div>` 的前面。

```
1 <div><p>Hello world!</p></div>
```

4. HTML标签不区分大小写，如 `<h1>` 和 `<H1>` 是一样的，但建议小写，因为大部分程序员都已小写为准。





## 2.4 HTML5文档结构

### HTML文档结构

#### 【代码】HTML文档结构

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>HTML文档结构</title>
6 </head>
7 <body>
8
9 </body>
10 </html>
```

- `<!DOCTYPE html>`：文档类型声明，表示该文件为 HTML5 文件。`<!DOCTYPE>` 声明必须是 HTML 文档的第一行，位于 `<html>` 标签之前。
- `<html></html>` 标签对：`<html>` 标签用来标识HTML文档的开始，`</html>` 标签位于HTML文档的最后面，用来标识HTML文档的结束。这两个标签对成对存在，中间的部分是文档的头部和主题。
- `<head></head>` 标签对：标签包含有关HTML文档的信息，可以包含一些辅助性标签。如 `<title></title>`、`<link />`、`<meta />`、`<style></style>`、`<script></script>` 等，浏览器除了会在标题栏显示 `<title>` 元素的内容外，不会向用户显示 head 元素内的其他任何内容。
- `<body></body>` 标签对：HTML文档的主体部分，在此标签中可以包含 `<p>`、`<h1>`、`<br>` 等众多标签。`<body>` 标签出现在 `</head>` 标签之后，且必须在闭标签 `</html>` 之前闭合。

## 2.5 小哥，做头吗？——head标签

---

### head标签

文档的头部描述了文档的各种属性和信息，包括文档的标题等，绝大多数文档头部包含的数据都不会真正作为内容显示给读者。

head 标签为双标签 `<head></head>`，表示头部标签，通常用来嵌套 `meta`、`title`、`style` 等标签。

- `<title>` 标签：在 `<title>` 和 `</title>` 标签之间的文字内容是网页的标题信息，它会出现浏览器的标题栏中。网页的title标签用于告诉用户和搜索引擎这个网页的主要内容是什么，搜索引擎可以通过网页标题，迅速的判断出网页的主题。每个网页的内容都是不同的，每个网页都应该有一个独一无二的title。
- `<meta charset="UTF-8">`：设置当前文件字符编码。
- `<style>` 标签：设置当前文件样式。

## 2.6 你就是馋人家的身子！——body标签

### body标签

在网页上要展示出来的页面内容一定要放在 `<body>` 标签中。

#### 【代码】body标签

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>body标签</title>
6 </head>
7 <body>
8     <!-- 标题标签 -->
9     <h1>HTML简介</h1>
10    <!-- 段落标签 -->
11    <p>HTML的全称为超文本标记语言，是一种标记语言。它包括一系列标签。通过这些标签可以将网络上的文
    档格式统一，使分散的Internet资源连接为一个逻辑整体。HTML文本是由HTML命令组成的描述性文本，HTML命
    令可以说明文字，图形、动画、声音、表格、链接等。</p>
12    <!-- 段落标签 -->
13    <p>超文本是一种组织信息的方式，它通过超级链接方法将文本中的文字、图表与其他信息媒体相关联。这些
    相互关联的信息媒体可能在同一文本中，也可能是其他文件，或是地理位置相距遥远的某台计算机上的文件。这种
    组织信息方式将分布在不同位置的信息资源用随机方式进行连接，为人们查找，检索信息提供方便。</p>
14 </body>
15 </html>
```

# 第3章 语义化标签

---

## 3.1 语义化

---

### 语义化 ( Semantic )

学习HTML标签需要注意标签的用途和标签在浏览器中的默认样式。

语义化，通俗的讲就是明白每个标签的用途，即在什么情况下使用此标签合理。比如，网页上文章的标题可以用标题标签、各个栏目的名称也可以使用标题标签、文章内容的段落就得放在段落标签中。

语义化可以带来一些好处：

- 更容易被搜索引擎收录
- 更容易让屏幕阅读器读出网页内容



## 3.2 开始我们的第一段对话吧——p标签

### p标签

如果想在网页上显示文章，就需要使用

标签了，把文章的段落放到 `<p>` 标签中。

```
1 <p>段落文本</p>
```

注意一段文字一个 `<p>` 标签，如在一篇文章中有三段文字，就要分别放到三个 `<p>` 标签中。

#### 【代码】p标签

```
1 <!DOCTYPE HTML>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>p标签</title>
6 </head>
7 <body>
8     <p>所有主流浏览器都支持p标签。</p>
9     <p>p标签定义段落。</p>
10    <p>p元素会自动在其前后创建一些空白。浏览器会自动添加这些空间，您也可以在样式表中规定。</p>
11 </body>
12 </html>
```

`<p>` 标签的默认样式，在段前段后都会有空白，如果不喜欢这个空白，可以用 CSS 样式来删除或改变它。

## 3.3 span标签

### span标签

`<span>` 标签是没有语义的，它的作用就是为了设置单独的样式用的。

```
1 | <span>文本</span>
```

如果现在想把一段中某些字设置成蓝色，这种情况下就可以用到 `<span>` 标签了。

#### 【代码】span标签

```
1 | <!DOCTYPE HTML>
2 | <html lang="en">
3 | <head>
4 |     <meta charset="UTF-8">
5 |     <title>span标签</title>
6 |     <style type="text/css">
7 |         span {
8 |             color: blue;
9 |         }
10 |    </style>
11 | </head>
12 | <body>
13 |     <p>莫里亚蒂有份包裹指明要交给夏洛克，乔恩作为夏洛克指定继承人，自然包裹也就应交给乔恩。如何处置
    这个包裹，让乔恩左右为难。想来想去，还是趁着夜色赶到墓地，敲碎了夏洛克坟前的墓碑。这是乔恩与夏洛克商
    定的暗号，只有在<span>生命攸关</span>的时候才能使用。</p>
14 | </body>
15 | </html>
```

## 3.4 做个标题党——hx标签

### hx标签

文章的段落用 `<p>` 标签，那么文章的标题可以使用标题标签。标题标签一共有6个，`<h1>`、`<h2>`、`<h3>`、`<h4>`、`<h5>`、`<h6>` 分别为一级标题、二级标题、三级标题、四级标题、五级标题、六级标题，并且依据重要性递减，`<h1>` 是最高等级。

```
1 <h1>标题文本</h1>
2 <h2>标题文本</h2>
3 <h3>标题文本</h3>
4 <h4>标题文本</h4>
5 <h5>标题文本</h5>
6 <h6>标题文本</h6>
```

网页上的各个栏目标题也可使用标题标签。因为 `<h1>` 标签在网页中比较重要，一般 `<h1>` 标签被用在网站名称上。

标题标签的样式都会加粗，`<h1>` 标签字号最大，`<h2>` 标签字号相对 `<h1>` 要小，以此类推 `<h6>` 标签的字号最小。

#### 【代码】hx标签

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>h1-h6标题标签</title>
6 </head>
7 <body>
8     <h1>一级标签</h1>
9     <h2>二级标签</h2>
10    <h3>三级标签</h3>
11    <h4>四级标签</h4>
12    <h5>五级标签</h5>
13    <h6>六级标签</h6>
14 </body>
15 </html>
```

## 3.5 div标签

### div标签

在网页制作过程中，可以把一些独立的逻辑部分划分出来，放在一个<div>标签中，这个<div>标签的作用就相当于一个容器。

```
1 <div></div>
```

逻辑部分是页面上相互关联的一组元素，如网页中的独立的栏目版块，就是一个典型的逻辑部分。如下图所示，图中用红色边框标出的部分就是一个逻辑部分，就可以使用<div>标签作为容器。



### 【代码】div标签

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>div标签</title>
6 </head>
7 <body>
8     <div>
9         <h2>热门课程排行榜</h2>
10        <ol>
11            <li>前端开发面试心法</li>
12            <li>零基础学习HTML</li>
13            <li>JavaScript全攻略</li>
14        </ol>
15    </div>
16
17    <div>
18        <h2>最新课程排行</h2>
19        <ol>
20            <li>版本管理工具介绍-Git篇</li>
21            <li>Canvas绘图详解</li>
22            <li>QQ5.0侧滑菜单</li>
23        </ol>
24    </div>
25 </body>
```





## 3.6 写代码都不换行吗？——br标签

### br标签

在HTML代码中输入回车、空格都是没有作用的，在HTML中是忽略回车和空格的，输入再多的回车和空格也是现实不出来的。如果需要在HTML文本中输入回车换行，就必须使用 `<br />` 标签。在需要加回车换行的地方加入 `<br />`，`<br />` 标签的作用相当于Word文档中的回车。

`<br />` 标签是一个单标签，没有HTML内容的标签就是单标签。单标签只需要写一个开始标签，这样的标签有 `<br />`、`<hr />` 和 `<img />`。

#### 【代码】br标签

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>br标签</title>
6  </head>
7  <body>
8      <h2>《望庐山瀑布》</h2>
9      <p>唐·李白</p>
10     <p>
11         日照香炉生紫烟，<br/>
12         遥看瀑布挂前川。<br/>
13         飞流直下三千尺，<br/>
14         疑是银河落九天。
15     </p>
16 </body>
17 </html>
```



## 3.8 再来个水平分割线——hr标签

### hr标签

在信息展示时，有时会需要加一些用于分隔的横线，这样会使文章看起来整齐些。

`<hr />` 标签和 `<br />` 标签一样也是一个单标签，所以只有一个开始标签，没有结束标签。

`<hr />` 标签在浏览器中的默认样式线条比较粗，颜色为灰色。有些人觉得这种样式不美观，这些在样式可以通过CSS进行修改。

#### 【代码】分割两段文字

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>hr标签</title>
6  </head>
7  <body>
8      <h2>第1集</h2>
9      <p>深夜，一道黑影潜入贝克街221B。坐在暗处的夏洛克已等候多时，正等着贝尔纳多自投罗网。贝尔纳多的目标是壁炉架上的半身像，摔到地上，从碎片中取出一个用旧布包裹好的东西。打开旧布，露出发着幽光的波吉亚黑珍珠。</p>
10     <hr/>
11     <h2>第2集</h2>
12     <p>葛雷森警监仍昏迷不醒，需要持续输血。医生认为有内出血情况，需要开刀确定出血点。马库斯在警局内询问报案人特里梅因，想了解他发现警监时的详细情况，但并没有取得太多线索。马库斯一抬头，看到了乔恩。因夏洛克不能回美国，所以只有乔恩赶来参与调查。</p>
13 </body>
14 </html>
```

# 第4章 列表、图片、链接

## 4.1 有序列表

### 有序列表 ( Ordered List )

如果想在网页中展示有前后顺序的信息列表，如热度排行榜等，这类信息展示可以使用 `<ol>`-`<li>` 标签来实现有序列表。

```
1 <ol>
2   <li>信息</li>
3   <li>信息</li>
4 </ol>
```

### 排行榜

[更多 >](#)

-  小伙把颜料乱洒到白体恤上不但没弄脏，反而成...  
综合得分：111.4万
- 5分钟大腿内侧运动，练出大腿中缝，收紧松...
- 【全民健身日】每天五分钟，快乐爽歪歪！
- 你离标准美人脸差几步？如何正确了解自己...
- 帕梅拉 莱孚 - 10min腹部训练 脖颈背部友好...
- 【吊带大扫雷】丑哭了的吊带！吊带大户的...
- 提臀线/改善久坐臀腿顽固脂肪/骨盆前倾/大...
- 【宁静】呼吸也能瘦！静静子亲授 半个月收...
- 第一次蹦迪！周围一堆美女，我好土！
- 【爆笑】致命推荐！超性感小裙子试穿，穿...

### 【代码】有序列表

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>有序列表</title>
6 </head>
7 <body>
8   <h3>2020年7月编程语言排行榜</h3>
9   <ol>
10    <li>C</li>
11    <li>Java</li>
12    <li>Python</li>
13    <li>C++</li>
14    <li>C#</li>
15  </ol>
16 </body>
```

## 序号类型

`<ol>`-`<li>` 在网页中显示的默认样式一般为每项 `<li>` 前都自带一个序号，序号默认从 1 开始。通过修改 `<ol>` 标签的 `type` 属性，也能对序号进行修改。

`<ol>` 标签的 `type` 属性有 5 个值：

1. 默认为数字序号：`type="1"`。
2. 小写字母序号：`type="a"`。
3. 大写字母序号：`type="A"`。
4. 小写罗马数字序号：`type="i"`。
5. 大写罗马数字序号：`type="I"`。

`<ol>` 标签中设置 `start` 属性，可以指定开始序号。

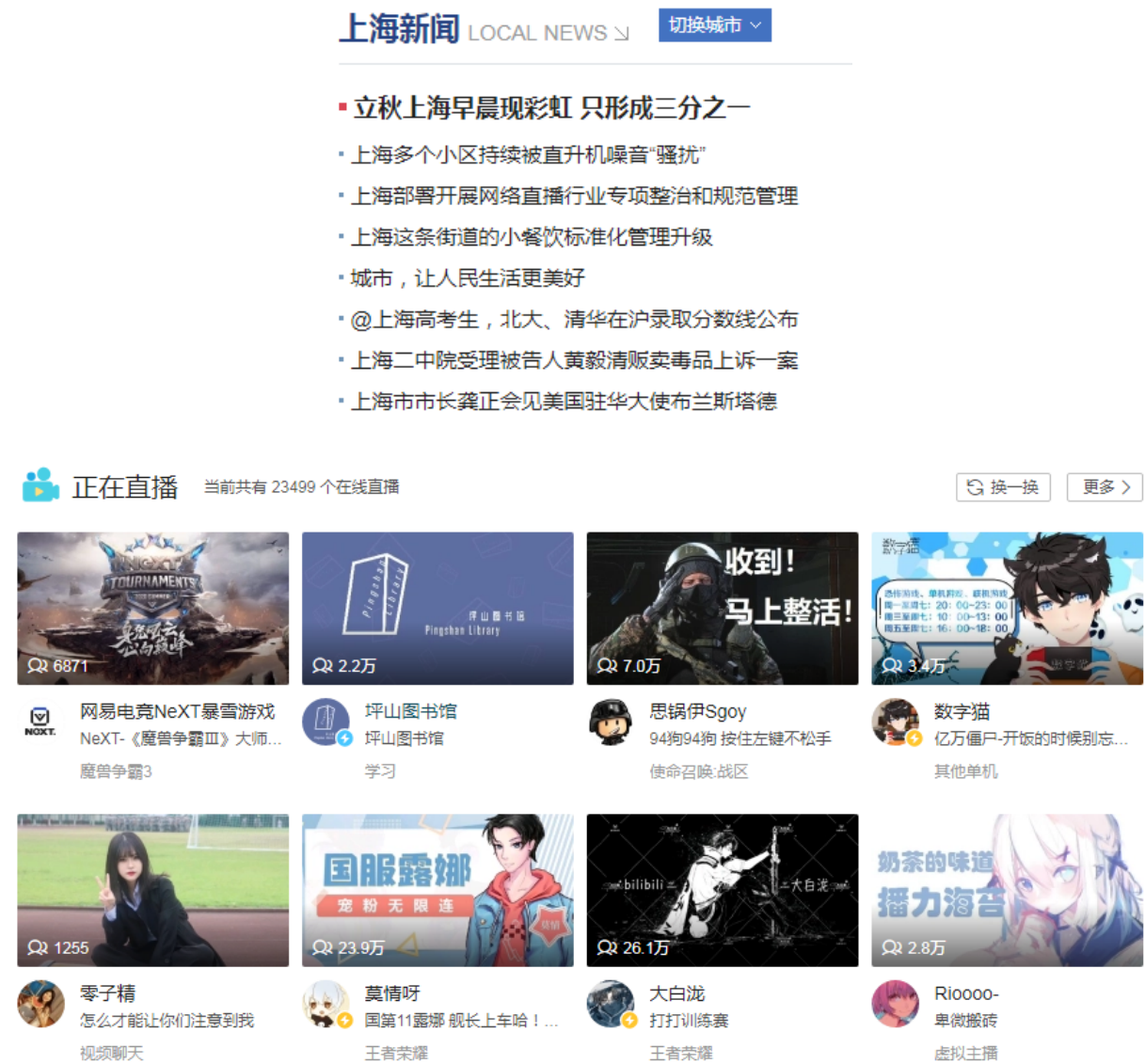
`<ol>` 标签中设置 `reversed="reversed"` 属性，可以将列表序号降序排序。

## 4.2 无序列表

### 无序列表 ( Unordered List )

网页上也有很多信息是无需按照先后次序排列的，如新闻列表、图片列表等，这类信息展示可以使用 `<ul>`-`<li>` 标签来实现无序列表。

```
1 <ul>
2   <li>信息</li>
3   <li>信息</li>
4 </ul>
```



`<ul>`-`<li>` 标签在网页中显示的默认样式一般为每项 `<li>` 前都自带一个圆点。通过修改 `<ul>` 标签的 `type` 属性，也能对前面的符号进行修改。

`<ul>` 标签的 `type` 属性有2个值：

1. 默认为实心小圆点：`type="disc"`。
2. 实心正方形：`type="square"`。

### 【代码】无序列表

```
1 <!DOCTYPE html>
```

```

2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>无序列表</title>
6 </head>
7 <body>
8     <h3>前端三剑客</h3>
9     <ul>
10         <li>HTML</li>
11         <li>CSS</li>
12         <li>JS</li>
13     </ul>
14 </body>
15 </html>

```

淘宝网的导航栏就是利用 `<ul>`-`<li>` 的父子结构进行实现的。

The screenshot shows the Taobao.com website. The navigation bar is a horizontal list of categories: 主题市场, 天猫, 聚划算, 天猫超市, 淘抢购, 电器城, 司法拍卖, 淘宝心选, 兴农. The HTML code on the right shows the structure of the navigation bar, with a red circle highlighting the '主题市场' (Main Market) link.

## 【代码】淘宝导航栏

```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>淘宝</title>
6     <style type="text/css">
7         ul {
8             list-style: none; /* 无序列表前面不带点 */
9         }
10
11         li {
12             float: left;
13             margin: 20px;
14             padding: 5px;
15             color: #000;
16         }
17
18         li:hover {
19             background-color: #f40; /* 淘宝红 */
20             color: #fff;
21             border-radius: 15px;

```



```
22         cursor: pointer;
23     }
24 </style>
25 </head>
26 <body>
27     <ul>
28         <li>天猫</li>
29         <li>聚划算</li>
30         <li>天猫超市</li>
31     </ul>
32 </body>
33 </html>
```

---

## 4.3 不发一张自拍吗？——img标签

### img标签

在网页的制作中为使网页炫丽美观，肯定是缺少不了图片，<img> 标签用于插入图片。

```
1 
```

- src 属性用于标识图像的资源地址，资源地址可以来源于网上的URL，也可以来源于本地。
- alt 属性指定图像的描述性文本，当图像下载不成功时，可看到该属性指定的文本。
- title 属性提供当鼠标停留在图片时显示的文本。

### 【代码】img标签

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>img标签</title>
6 </head>
7 <body>
8     
11 </body>
12 </html>
```



```
12 <a href="#div2">跳转到div2</a>
13 </body>
14 </html>
```

`<a>` 标签作为锚点的应用场景也很多，比如目录跳转。

#### 概览

变量 (Variables)  
混合 (Mixins)  
嵌套 (Nesting)  
运算 (Operations)  
转义 (Escaping)  
函数 (Functions)  
命名空间和访问符  
映射 (Maps)  
作用域 (Scope)  
注释 (Comments)  
导入 (Importing)

## 概览

Less (Leaner Style Sheets 的缩写) 是一门向后兼容的 CSS 扩展语言。这里呈现的是 Less 的官方文档 (中文版)，包含了 Less 语言以及利用 JavaScript 开发的用于将 Less 样式转换成 CSS 样式的 Less.js 工具。

因为 Less 和 CSS 非常像，因此很容易学习。而且 Less 仅对 CSS 语言增加了少许方便的扩展，这就是 Less 如此易学的原因之一。

- 有关 Less 语言特性的详细文档，请参阅 [Less 语言特性](#) 章节
- 有关 Less 内置函数的列表，请参阅 [Less 函数手册](#) 章节
- 有关详细的使用说明，请参阅 [Less.js 用法](#) 章节
- 有关第三方工具的详细信息，请参阅 [工具](#) 章节

Less 到底为 CSS 添加了什么功能？以下就是这些新加功能的概览。

## 变量 (Variables)

无需多说，看代码一目了然：

```
@width: 10px;
@height: @width + 10px;
```

## 协议限定符

`<a>` 标签还能作为协议限定符，在点击链接时会强制执行JavaScript代码。

### 【代码】协议限定符

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>访问限定符</title>
6 </head>
7 <body>
8   <a href="JavaScript: while(1){alert('嘿嘿');}">点我</a>
9 </body>
10 </html>
```

# 第5章 表格、表单

## 5.1 表格标签

### table标签

有时候需要在网页中添加一些表格数据，如企业员工通讯录等。

企业员工通讯录 → 标题标签

企业员工通讯录 → 表格标题 (和文章标题区分)

姓名	电话	电子邮件	职务
张三	18278900988	zhangsan@163.com	研发工程师
王二	16589012689	wanger@163.com	研发经理
李四	17230019065	lisi@163.com	研发工程师

→ 表格的行

表格的单元格 (行和列都是由单元格构成的)

表格的列

创建表格主要包含5个元素：

1. `<caption>` 标签：定义表格的标题。
2. `<table>` 标签：整个表格以 `<table>` 标记开始、`</table>` 标记结束，`<table>` 标签在没有添加 `border` 属性之前，在浏览器中显示是没有表格线的。
3. `<tr>` 标签：表格的一行，有几对 `<tr>` 表格就有几行，`<tr>` 标签里只能放 `<th>` 或者 `<td>` 标签。
4. `<th>` 标签：表格头部的一个单元格，会加粗居中显示。
5. `<td>` 标签：表格的一个单元格，有几对 `<td>` 一行中就有几列。

### 【代码】企业员工通讯录

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>企业员工通讯录</title>
6 </head>
7 <body>
8     <h3>企业员工通讯录</h3>
9     <hr/>
10
11     <!-- 表格标签 border属性代表给表格加上边框 -->
12     <table border="1">
13         <!-- 表格标题 -->
14         <caption>企业员工通讯录</caption>
15         <!-- tr代表一行 -->
16         <tr>
17             <!-- th代表表格头部的一个单元格 -->
18             <th>姓名</th>
19             <th>电话</th>
20             <th>电子邮件</th>
21             <th>职务</th>
22         </tr>
23         <tr>
```

```
24      <!-- td代表一个单元格 -->
25      <td>张三</td>
26      <td>18278900988</td>
27      <td>zhangsan@163.com</td>
28      <td>研发工程师</td>
29  </tr>
30  <tr>
31      <td>王二</td>
32      <td>16589012689</td>
33      <td>wanger@163.com</td>
34      <td>研发经理</td>
35  </tr>
36  <tr>
37      <td>李四</td>
38      <td>17230019065</td>
39      <td>lisi@163.com</td>
40      <td>研发工程师</td>
41  </tr>
42 </table>
43 </body>
44 </html>
```

## 5.2 与用户交互——form标签

### form标签

使用HTML表单（form）可以实现网站与用户的交互，表单可以把浏览者输入的数据传送到服务器端，这样服务器端程序就可以处理表单传过来的数据。

```
1 | <form method="传送方式" action="服务器文件">表单内容</form>
```

`<form>` 标签是成对出现的，以 `<form>` 开始、`</form>` 结束。`method` 属性表示数据传送的方式，包括 `get` 和 `post` 两种方式，`get` 和 `post` 的区别属于后端程序员需要考虑的问题，完全取决于后端人员要求什么方式进行传输。`action` 属性表示浏览者输入的数据被传送到地方，比如一个PHP页面。

所有的表单控件（文本框、文本域、按钮、单选框、复选框等）都必须放在 `<form>` 和 `</form>` 标签之间，否则用户输入的信息无法提交到服务器上。

## 5.3 先来填用户名和密码——文本、密码输入框

### 文本、密码输入框

当用户需要在表单中输入字母、数字等内容时，就需要使用到文本输入框，文本框也可转化为密码输入框。

```
1 <form>
2   <input type="text/password" name="名称" value="文本" />
3 </form>
```

- `type="text"`：输入框为文本输入框。
- `type="password"`：输入框为密码输入框。
- `name` 属性：为文本框命名，以备后台程序使用。
- `value` 属性：为文本输入框设置默认值，一般起提示作用。

### 给点提示呗——placeholder属性

有时候需要提示用户输入框需要输入的内容，这时候就需要使用 `<input>` 标签中的占位符 `placeholder` 属性。

账号：	<input type="text" value="请输入账号"/>
密码：	<input type="password" value="请输入密码"/>

`placeholder` 属性的值可以根据情况合理填写，当输入框输入内容时，占位符内容消失，当输入框无内容时，占位符内容显示。注意，占位符内容不是输入框真正的内容。

### 重置按钮

当用户需要重置表单信息到初始状态时，比如输入“账号”或“密码”有误，就可以使用重置按钮使输入框恢复到初始状态。通过设置 `<input>` 标签中 `type` 属性为 `reset` 即可实现重置按钮。

### 提交按钮

当用户需要提交表单信息到服务器时，需要用到提交按钮。通过设置 `<input>` 标签中 `type` 属性为 `submit` 即可实现提交按钮。表单信息会被发送给后端，在数据库对比账户和密码信息。

#### 【代码】登录功能

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4   <meta charset="UTF-8">
5   <title>登录</title>
6 </head>
7 <body>
8   <form action="get/post">
9     账号: <input type="text" placeholder="请输入账号" name="username">
10    <br/>
11    密码: <input type="password" placeholder="请输入密码" name="password">
12    <br/>
13    <input type="submit">
```



```
14     <input type="reset">
15   </form>
16 </body>
17 </html>
```

---

## 5.4 数字、网址、邮箱输入框

### 数字输入框

将 `<input>` 标签的 `type` 属性设置为 `number`，则表示该输入框的类型为数字。数字框只能输入数字，输入其它字符无效。数字框最右侧会有一个加减符号，可以调整输入数字的大小，不同浏览器表现不一致。

### 网址输入框

将 `<input>` 标签的 `type` 属性设置为 `url`，则表示该输入框的类型为网址。网址输入框的必须以 `http://` 或 `https://` 开头，且后面必须有内容，否则表单提交的时候会报错误提示。

### 邮箱输入框

将 `<input>` 标签的 `type` 属性设置为 `email`，则表示该输入框的类型为邮箱。邮箱输入框的值必须包含 `@`，并且之后必须有内容，否则会报错误提示。

#### 【代码】提交个人信息

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>个人信息</title>
6  </head>
7  <body>
8      <form action="get/post">
9          姓名: <input type="text" name="name"><br/>
10         年龄: <input type="number" name="age"><br/>
11         主页: <input type="url" name="webpage"><br/>
12         邮箱: <input type="email" name="email"><br/>
13         <input type="submit">
14         <input type="reset">
15     </form>
16 </body>
17 </html>
```

## 5.5 文本域

### 文本域

当用户需要在表单中输入大段文字时，就需要用到文本输入域。`<textarea>` 标签中也可以设置 `placeholder` 属性来显示提示信息。

```
1 <textarea rows="行数" cols="列数">文本</textarea>
```

- `rows` 属性：多行输入域的行数，可以用CSS样式的height代替。
- `cols` 属性：多行输入域的列数，可以用CSS样式的width代替。

### 【代码】文本域

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>文本域</title>
6 </head>
7 <body>
8     <form action="get/post">
9         <textarea cols="30" rows="10" placeholder="请输入..."></textarea>
10    </form>
11 </body>
12 </html>
```

## 5.6 label标签

### label标签

`<label>` 标签不会向用户呈现任何特殊特效，它的作用是为鼠标用户改进了可用性。如果在 `<label>` 标签内点击文本，就会触发此控件。也就是说，当用户单击选中该 `<label>` 标签时，浏览器就会自动将焦点转到和标签相关的表单控件上。

```
1 | <label for="控件id名称">
```

注意，`<label>` 标签的 `for` 属性的值必须要与相关空间的 `id` 属性值相同。

#### 【代码】label标签

```
1 | <!DOCTYPE html>
2 | <html lang="en">
3 | <head>
4 |     <meta charset="UTF-8">
5 |     <title>label标签</title>
6 | </head>
7 | <body>
8 |     <form action="get/post">
9 |         <label for="username">用户名: </label>
10 |         <input type="text" id="username">
11 |     </form>
12 | </body>
13 | </html>
```

## 5.7 单选框、复选框

### 单选框、复选框

在使用表单设计调查表时，为了减少用户的操作，使用选择框是一个好主意。HTML中有两种选择框，即单选框和复选框，两者的区别是单选框中的选项用户只能选择一项，而复选框中用户可以任意选择多项，甚至全选。

```
1 <input type="radio/checkbox" name="名称" value="值" checked="checked">
```

- `type="radio"`：单选框。
- `type="checkbox"`：复选框。
- `name` 属性：为控件命名，具有相同名称的选择框属于同一组。
- `checked` 属性：设置默认选中的值。

在网页开发中，往往需要优化用户体验。一个好的产品有3个特点：解决刚需的问题、用户体验、用户黏性（产品定位）。用户体验就是养成用户的懒习惯，减少用户操作，能不让用户操作就不让用户操作，节省用户时间，用户不动手就会觉得方便。例如一个选择性别单选框，概率上看，假设有100个人，50男50女，使用默认选项，可以节省一半人的操作。

#### 【代码】单选框/复选框

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>单选框/复选框</title>
6 </head>
7 <body>
8     <form action="get/post">
9         1. 您是否喜欢运动?
10        <input type="radio" name="sport" checked="checked">是
11        <input type="radio" name="sport">否
12        2. 您喜欢的运动是?
13        <input type="checkbox" name="sport_item">跑步
14        <input type="checkbox" name="sport_item">篮球
15        <input type="checkbox" name="sport_item">羽毛球
16    </form>
17 </body>
18 </html>
```

## 5.8 下拉列表

### 下拉列表

下拉列表在网页中也常会用到，它可以有效的节省网页空间。下拉列表既可以单选也可以多选。

```
1 <select>
2     <option value="值">选项</option>
3     <option value="值">选项</option>
4 </select>
```

- `value` 属性：向服务器提交的值，而双标签中间的内容才是显示的值。
- `selected` 属性：设置 `selected="selected"` 表示该选项被默认选中。

### 【代码】下拉列表

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>下拉列表</title>
6 </head>
7 <body>
8     <!DOCTYPE html>
9 <html lang="en">
10 <head>
11     <form action="get/post">
12         <select>
13             <option value="male" selected="selected">男</option>
14             <option value="female">女</option>
15         </select>
16     </form>
17 </body>
18 </html>
```

# 第6章 CSS3介绍

## 6.1 给HTML打扮——CSS样式

### 层叠样式表 ( CSS, Cascading Style Sheets )

CSS主要用于定义HTML内容在浏览器内的显示样式，如文字大小、颜色、字体加粗等。

使用CSS样式的一个好处是通过定义某个样式，可以让不同网页位置的问题有着统一的字体、字号或颜色等。

CSS样式由选择器和声明组成，而声明又由属性和值组成。

```
1 selector {  
2     attribute: value;  
3 }
```

选择器也称选择符，用于指明网页中要应用样式规则的元素。声明的内容写在大括号 {} 内，属性和值之前用冒号 : 分隔。当有多条声明时，中间可以用分号 ; 分隔。

最后一条声明可以没有分号，但是为了后续修改方便，一般也加上分号。为了使样式更加容易阅读，可以将每条声明单独成行。

#### 【代码】修改字体大小和颜色

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>修改字体大小和颜色</title>  
6     <style type="text/css">  
7         p {  
8             font-size: 20px;  
9             color: red;  
10        }  
11    </style>  
12 </head>  
13 <body>  
14     <p>修改字体大小和颜色</p>  
15 </body>  
16 </html>
```

## 6.2 既然那么好，那就引入CSS吧——内联式CSS

### 内联式CSS

内联式CSS样式，也称行内样式，就是把CSS代码直接写在现有的HTML标签中。注意CSS样式必须写在元素的开始标签里，不能在结束标签里。

```
1 | <开始标签 style="属性: 值;">文本</结束标签>
```

CSS样式必须写在 style 属性的双引号中，如果有多条CSS样式代码可以设置可以写在一起，中间用;隔开。

#### 【代码】内联式CSS样式

```
1 | <!DOCTYPE html>
2 | <html lang="en">
3 | <head>
4 |     <meta charset="UTF-8">
5 |     <title>内联式CSS</title>
6 | </head>
7 | <body>
8 |     <p style="color: red; font-size: 20px;">内联式CSS</p>
9 | </body>
10| </html>
```



## 6.3 换个地方吧，行内太挤了——嵌入式CSS

### 嵌入式CSS

嵌入式CSS样式，也称页面级CSS样式，就是把CSS样式代码写在 `<style type="text/css"></style>` 标签之间。

```
1 <style type="text/css">
2     selector {
3         attribute: value;
4     }
5 </style>
```

嵌入式CSS样式一般放在 `<head></head>` 标签之间。

#### 【代码】嵌入式CSS样式

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>嵌入式CSS</title>
6     <style type="text/css">
7         span {
8             color: red;
9         }
10    </style>
11 </head>
12 <body>
13     <p>既然那人知道利用<span>莫里亚蒂</span>的名头，说不定<span>莫里亚蒂</span>也会知道那人。
    抱着种想法，<span>乔恩</span>找到<span>艾洛莉</span>。</p>
14 </body>
15 </html>
```

## 6.4 还是把HTML和CSS分开吧——外部式CSS

### 外部式CSS

外部式CSS样式，也称外联式CSS样式，就是把CSS样式代码写一个单独的外部文件中，这个CSS样式文件以 .css 为扩展名。在 <head> 标签内使用 <link> 标签将CSS外部样式文件链接到HTML文件内。

```
1 <link href="CSS样式文件名" rel="stylesheet" type="text/css" />
```

CSS样式文件名以有意义的英文命名，<link> 标签中 rel="stylesheet" type="text/css" 属性是固定写法，不需要修改。

#### 【代码】外部式CSS样式

- external\_css.html

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>外部式CSS</title>
6     <link type="text/css" rel="stylesheet" href="external_css.css">
7 </head>
8 <body>
9     <div></div>
10 </body>
11 </html>
```

- external\_css.css

```
1 div {
2     width: 100px;
3     height: 100px;
4     background-color: blue;
5 }
```

## 6.5 总有个先来后到吧——三种链接方式的优先级

### CSS引入方式优先级

如果有一种情况：对于同一个元素同时使用了三种方法设置CSS样式，那么哪种方式真正有效呢？

三种CSS引入方式是有优先级的：`内联式` > `嵌入式` > `外部式`。但是 `嵌入式` > `外部式` 有一个前提，那就是嵌入式CSS样式的位置一定在外部式的后面。在实际开发中也会将 `<link>` 标签写在 `<style>` 标签的前面。

总的来说，优先级遵循 `就近原则`，离被设置元素越近优先级别越高。

但是以上总结的优先级有一个前提，那就是内联式、嵌入式、外部式样式表中CSS样式是在相同权值的情况下。那权值是什么呢？



# 第7章 选择器

## 7.1 选一个标签——标签选择器

### 选择器 ( Selector )

CSS样式由选择器和声明组成，而声明又由属性和值组成。

```
1 selector {  
2     attribute: value;  
3 }
```

在 {} 之前的部分就是选择器，选择器指明了 {} 中的样式的作用对象，也就是这些样式作用于网页中的哪些元素。

### 标签选择器

标签选择器其实就是HTML代码中的标签。

```
1 tag_selector {  
2     attribute: value;  
3 }
```

### 【代码】 标签选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>标签选择器</title>  
6     <style type="text/css">  
7         h2 {  
8             color: green;  
9             font-size: 30px;  
10        }  
11    </style>  
12 </head>  
13 <body>  
14     <h2>CSS特点</h2>  
15     <p>CSS为HTML标记语言提供了一种样式描述，定义了其中元素的显示方式。CSS在web设计领域是一个突破。利用它可以实现修改一个小的样式更新与之相关的所有页面元素。</p>  
16 </body>  
17 </html>
```

## 7.2 再选一个类——类选择器

### 类选择器

类选择器在CSS样式中是最常用的。

```
1 .class_selector {  
2     attribute: value;  
3 }
```

类选择器使用 `.` 开头，后加类选择器的名称，CSS样式代码会被作用到属于该类的HTML标签中。在标签中使用 `class` 属性为标签设置一个类。

类选择器与标签是多对多的关系，即类选择器名称可以多个标签共用，一个元素可以用多个class，一个class值可以对应多个元素。多个class值之间使用空格分隔。

#### 【代码】类选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>类选择器</title>  
6     <style type="text/css">  
7         .title {  
8             color: green;  
9         }  
10    </style>  
11 </head>  
12 <body>  
13     <h3 class="title">丰富的样式定义</h3>  
14     <p>CSS提供了丰富的文档样式外观，以及设置文本和背景属性的能力；允许为任何元素创建边框，以及元素  
    边框与其他元素间的距离，以及元素边框与元素内容间的距离；允许随意改变文本的大小写方式、修饰方式以及其  
    他页面效果。</p>  
15     <h3 class="title">易于使用和修改</h3>  
16     <p>CSS可以将样式定义在HTML元素的style属性中，也可以将其定义在HTML文档的header部分，也可以  
    将样式声明在一个专门的CSS文件中，以供HTML页面引用。总之，CSS样式表可以将所有的样式声明统一存放，进  
    行统一管理。</p>  
17     <h3 class="title">多页面应用</h3>  
18     <p>CSS样式表可以单独存放在一个CSS文件中，这样我们就可以在多个页面中使用同一个CSS样式表。CSS  
    样式表理论上不属于任何页面文件，在任何页面文件中都可以将其引用。这样就可以实现多个页面风格的统一。  
    </p>  
19 </body>  
20 </html>
```

## 7.3 取个唯一表示——ID选择器

### ID选择器

使用ID选择器，必须给标签添加上 `id` 属性，即为标签设置 `id` 属性。ID选择器名称的前面使用 `#`。

```
1 #id_selector {  
2     attribute: value;  
3 }
```

ID选择器与标签是一一对应的关系，即一个元素只能有一个id值，一个id值只能对应一个元素。id是全局唯一的，就像身份证号码一样。

#### 【代码】ID选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>ID选择器</title>  
6     <style type="text/css">  
7         div {  
8             width: 100px;  
9             height: 100px;  
10        }  
11  
12        #square1 {  
13            background-color: red;  
14        }  
15        #square2 {  
16            background-color: blue;  
17        }  
18    </style>  
19 </head>  
20 <body>  
21     <div id="square1"></div>  
22     <div id="square2"></div>  
23 </body>  
24 </html>
```

## 7.4 捡了个儿子——子选择器

### 子选择器

子选择器 `>`，用于选择指定标签元素的第一代子元素。

```
1 parent_selector > child_selector {  
2     attribute: value;  
3 }
```

#### 【代码】子选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>子选择器</title>  
6     <style type="text/css">  
7         .food > li {  
8             border: 1px solid red;  
9         }  
10    </style>  
11 </head>  
12 <body>  
13     <h2>食物</h2>  
14     <ul class="food">  
15         <li>  
16             水果  
17             <ul>  
18                 <li>苹果</li>  
19                 <li>香蕉</li>  
20                 <li>橘子</li>  
21             </ul>  
22         </li>  
23         <li>  
24             蔬菜  
25             <ul>  
26                 <li>白菜</li>  
27                 <li>油菜</li>  
28                 <li>卷心菜</li>  
29             </ul>  
30         </li>  
31     </ul>  
32 </body>  
33 </html>
```

## 7.5 这么快就当爷爷了——后代选择器

### 后代选择器

后代选择器（空格），也称包含选择器，用于选择指定标签元素的后辈元素。语法如下：

```
1 ancestor_selector descendant_selector {  
2     attribute: value;  
3 }
```

后代选择器与子选择器的区别在于，子选择器仅是指它的直接后代，而后代选择器是作用于所有子后代元素。

#### 【代码】后代选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>后代选择器</title>  
6     <style type="text/css">  
7         .food li {  
8             border: 1px solid red;  
9         }  
10    </style>  
11 </head>  
12 <body>  
13     <h2>食物</h2>  
14     <ul class="food">  
15         <li>  
16             水果  
17             <ul>  
18                 <li>苹果</li>  
19                 <li>香蕉</li>  
20                 <li>橘子</li>  
21             </ul>  
22         </li>  
23         <li>  
24             蔬菜  
25             <ul>  
26                 <li>白菜</li>  
27                 <li>油菜</li>  
28                 <li>卷心菜</li>  
29             </ul>  
30         </li>  
31     </ul>  
32 </body>  
33 </html>
```



## 7.6 我全都要——通配符选择器

### 通配符选择器

通配符选择器 `*`，也称通用选择器，是功能最强大的选择器，用于匹配HTML中所有的标签元素，包括 `<html>`、`<body>` 等。

```
1 * {  
2     attribute: value;  
3 }
```

#### 【代码】通配符选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>通配符选择器</title>  
6     <style type="text/css">  
7         * {  
8             background-color: yellow;  
9         }  
10    </style>  
11 </head>  
12 <body>  
13  
14 </body>  
15 </html>
```

## 7.7 给选择器分个组——分组选择器

### 分组选择器

分组选择器 `,`，用于为HTML中多个标签元素设置同一个样式。

```
1 selector, selector {  
2     attribute: value;  
3 }
```

#### 【代码】分组选择器

```
1 <!DOCTYPE html>  
2 <html lang="en">  
3 <head>  
4     <meta charset="UTF-8">  
5     <title>分组选择器</title>  
6     <style type="text/css">  
7         h1, h2, h3 {  
8             color: red;  
9         }  
10    </style>  
11 </head>  
12 <body>  
13     <h1>HTML</h1>  
14     <h2>CSS</h2>  
15     <h3>JavaScript</h3>  
16 </body>  
17 </html>
```

## 7.8 伪装者——伪类选择器

### 伪类选择器

伪类选择器允许给HTML标签的某种状态设置样式，例如给一个标签元素的鼠标滑过的状态设置字体颜色。

```
1 selector:pseudo_selector {
2     attribute: value;
3 }
```

伪类选择器	功能
:link	未访问
:visited	已访问
:hover	鼠标悬停
:active	鼠标按下
:enabled	可用的时候触发
:disabled	不可用的时候触发

到目前为止，可以兼容所有浏览器的伪类选择器就是在 `<a>` 标签上使用 `:hover`。

#### 【代码】伪类选择器

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>伪类选择器</title>
6     <style type="text/css">
7         a:hover {
8             color: orange;
9         }
10    </style>
11 </head>
12 <body>
13     <a href="http://www.baidu.com">百度一下，你就知道</a>
14 </body>
15 </html>
```

## 7.9 别插队啊——选择器的优先级

### 选择器优先级

每个选择器都是有优先级的，如果一个元素使用了多个选择器，则会按照选择器的优先级来给定样式。

选择器的优先级依次是：

内联样式 > *ID*选择器 > 类选择器 > 标签选择器 > 通配符选择器

### 选择器权重

浏览器是根据权值来判断使用哪种CSS样式的，权值高的优先级更高。

选择器	权值
!important	$\infty$
行间样式	1000
ID	100
class	10
标签	1
通配符	0

## 7.10 为所欲为——选择器最高层级!important

### !important

有些特殊情况需要为某些样式设置具有最高权值，这时候可以使用 `!important`，注意 `!important` 要写在分号的前面。

#### 【代码】!important

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>!important</title>
6     <style type="text/css">
7         h1 {
8             color: red !important;
9         }
10
11        h1 {
12            color: blue;
13        }
14    </style>
15 </head>
16 <body>
17     <h1>为所欲为</h1>
18 </body>
19 </html>
```

# 第8章 字体、文本样式

## 8.1 字体样式

### 字体样式

使用CSS样式可以为网页中的文字设置字体。注意不要设置不常用的字体，因为如果用户本地电脑上没有安装该字体，就会显示浏览器默认的字體。

浏览器默认的字号为 16px，使用 font-size 可以修改字号大小。

为文字设置粗体是有单独的CSS样式来实现的，再也不用为了实现粗体样式而使用 <h1>-<h6> 或 <strong> 标签了。

font-weight 的默认值为 normal，通过设置属性值为 lighter、bold、bolder 或 100-900 之间的整百数值改变文字的粗细。注意，字体能否被 bolder 或 lighter 更改取决于字体包是否存在该样式。

font-style 可以设置字体样式，并且有3种设置方式：

1. 正常字体为 normal，也是 font-style 的默认值。
2. italic 为字体设置为斜体，用于字体本身就有倾斜的样式。
3. oblique 强制将字体倾斜。

### 【代码】字体样式

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>字体样式</title>
6     <style type="text/css">
7         p {
8             font-family: "arial";
9             font-size: 20px;
10            font-weight: bold;
11            font-style: italic;
12        }
13    </style>
14 </head>
15 <body>
16     <p>Cascading Style Sheets (CSS) is a style sheet language used for describing
        the presentation of a document written in a markup language such as HTML.
        [1] CSS is a cornerstone technology of the world wide web, alongside HTML and JavaScript.</p>
17 </body>
18 </html>
```

# 8.2 上个色——color

## color

color 属性可以设置字体颜色。color 的值有3种设置方式：

1. 英文命令颜色。

```
1 | color: red;
```

2. 十六进制颜色代码：使用 6 位十六进制数表示光学三原色 红绿蓝。

R	G	B
00-ff	00-ff	00-ff

如果每两位十六进制数都相同，可简写。

颜色代码	颜色
#000000	黑色
#ffffff	白色
#ff0000	正红
#00ff00	正绿
#0000ff	正蓝
#00ffff	青色
#f40	淘宝红

3. 颜色函数 rgb()：由光学三原色 RGB 的比例来配色。rgb() 函数中每一项的值可以是 0-255 之间的整数，也可以是 0%-100% 的百分数。

```
1 | color: rgb(133, 45, 200);
2 | color: rgb(20%, 33%, 25%);
```





## 8.3 文本样式

### 文本样式

`text-decoration` 可以设置添加到文本的修饰，默认值为 `none`。属性值为 `underline` 为文本添加下划线，属性值为 `overline` 为文本添加上划线，属性值为 `line-through` 为添加穿过文本的线，一般用于商品折扣价。

使用 `line-height` 可以设置段落中的行间距离（行高）。

`text-align` 可以为文本设置对齐方式，属性值包括 `left`、`right` 和 `center`。

#### 【代码】文本样式

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>文本样式</title>
6      <style type="text/css">
7          h2, p {
8              text-decoration: underline;
9              line-height: 2em;    /* 两倍行间距 */
10             text-align: center;
11         }
12     </style>
13 </head>
14 <body>
15     <h2>《望庐山瀑布》</h2>
16     <p>唐·李白</p>
17     <p>
18         日照香炉生紫烟，<br/>
19         遥看瀑布挂前川。<br/>
20         飞流直下三千尺，<br/>
21         疑是银河落九天。
22     </p>
23 </body>
24 </html>
```

# 第9章 盒子模型与布局模型

## 9.1 元素分类

### 我要独占一行——块级元素

在HTML中 `<div>`、`<p>`、`<h1>`、`<form>`、`<ul>`、`<li>` 等都是块级元素。每个块级元素都从新的一行开始，并且其后的元素也另起一行。块级元素的高度、宽度、行高、顶边距和底边距都可以设置，宽度在不设定的情况下，是它本身父容器的 100%（和父元素的宽度一致）。

#### 【代码】块级元素

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>块级元素</title>
6 </head>
7 <body>
8     <div>这是一个div标签</div>
9     <p>这是一个p标签</p>
10    <h1>这是一个h1标签</h1>
11 </body>
12 </html>
```

通过设置 `display: block` 可以将元素显示为块级元素。

### 我要和你站一起——内联元素

在HTML中，`<span>`、`<a>`、`<label>`、`<strong>`、`<em>` 等都是内联元素（行内元素）。内联元素和其它元素都在一行上，元素的高度、宽度及顶部和底部边距不可设置，元素的宽度就是它包含的文字或图片的宽度，不可改变。

块级元素也可以设置 `display: inline` 将元素设置为内联元素。

#### 【代码】内联元素与块级元素转换

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>内联元素与块级元素转换</title>
6     <style type="text/css">
7         a {
8             display: block;
9         }
10
11        div {
12            display: inline;
13        }
14    </style>
```

```
15 </head>
16 <body>
17     <a>我要变成块级元素</a>
18     <a>我也要变成块级元素</a>
19
20     <div>我要变成内联元素</div>
21     <div>我也要变成内联元素</div>
22 </body>
23 </html>
```

### 我还要占个大位置——内联块状元素

内联块状元素就是同时具备内联元素和块级元素的特点。通过设置 `display: inline-block` 就可以将元素设置为内联块状元素。如 `<img>`、`<input>` 标签就是这种内联块状标签。

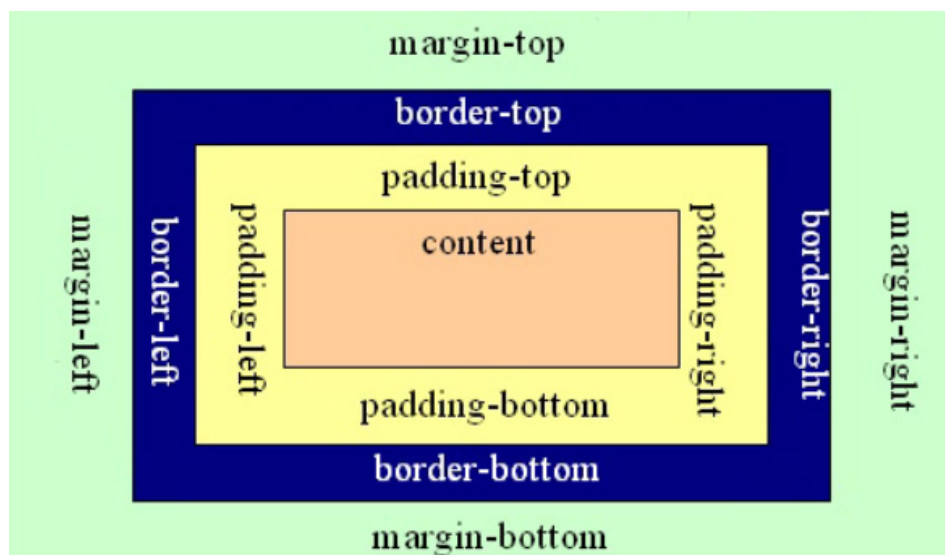
内联块状元素的特点是和其它元素都在一行上，但元素的高度、宽度、行高遗迹顶和底边距都可以设置。

## 9.2 盒子模型

### 盒子模型

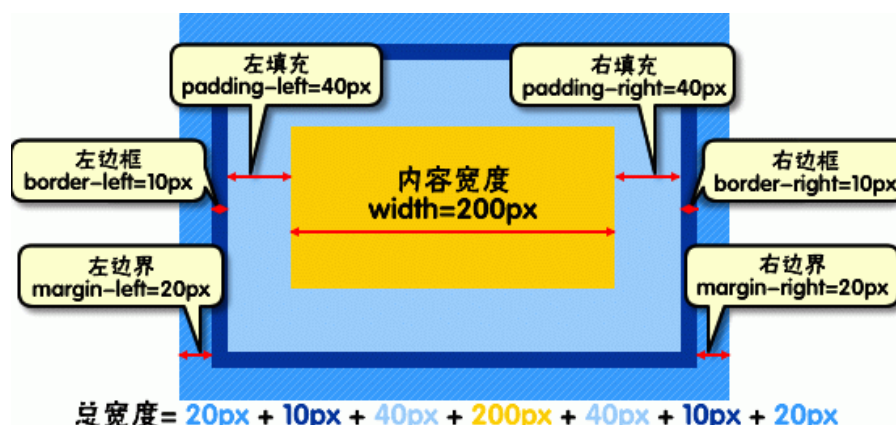
盒子模型包含4个部分：

- 外边距 ( margin )
- 外边框 ( border )
- 内边距 ( padding )
- 内容 ( content )



盒模型的宽度和高度和平常所说的物体的宽度和高度的理解是不一样的，CSS内定义的宽和高，指的是盒模型中内容的宽和高。

元素实际的宽度（盒子的宽度）= 左外边距 + 左边框 + 左内边距 + 内容宽度 + 右内边距 + 右边框 + 右外边距



### 外边框 ( Border )

盒子模型的外边框可以设置粗细、样式和颜色：

1. border-width：设置边框的宽度，常用单位为像素 ( px )。
2. border-style：边框样式包括实线 solid、虚线 dashed、dotted 点线。
3. border-color：设置边框颜色。

当三个属性都需要被设置时，也可以使用 border 属性简写。

例如设置一个粗细为 1px 实心的黑色边框：

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

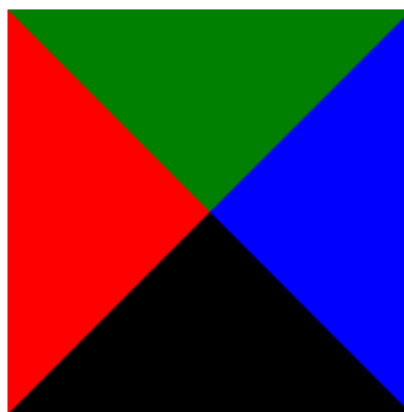
CSS中也允许只为一个方向的边框设置属性，使用 `border-top`、`border-bottom`、`border-left`、`border-right` 可以单独设置上、下、左、右四条边框。

元素边框的圆角效果可以使用 `border-radius` 属性来设置。设置圆角的值的顺序为左上、右上、右下、左下。如果 `border-radius` 属性只给出一个值表示四个圆角都被设置成该值。如果只给出两个值表示左上角和右下角设置为第一个值，右上角和左下角设置为第二个值。当给一个正方形的圆角效果值设置为其宽度一半时，显示效果为圆形。

#### 【代码】圆形

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>圆形</title>
6     <style type="text/css">
7         div {
8             width: 100px;
9             height: 100px;
10            background-color: red;
11            border-radius: 50%;
12        }
13    </style>
14 </head>
15 <body>
16     <div></div>
17 </body>
18 </html>
```

通过单独设置每条边的属性，可以在正方形内展现不同的颜色。



#### 【代码】四色正方形

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>四色正方形</title>
6     <style type="text/css">
```

```

7      div {
8          width: 0;
9          height: 0;
10         border: 100px solid black;
11         border-left-color: red;
12         border-top-color: green;
13         border-right-color: blue;
14     }
15 </style>
16 </head>
17 <body>
18     <div></div>
19 </body>
20 </html>

```

通过设置透明色（transparent），可以绘制出三角形。



#### 【代码】三角形

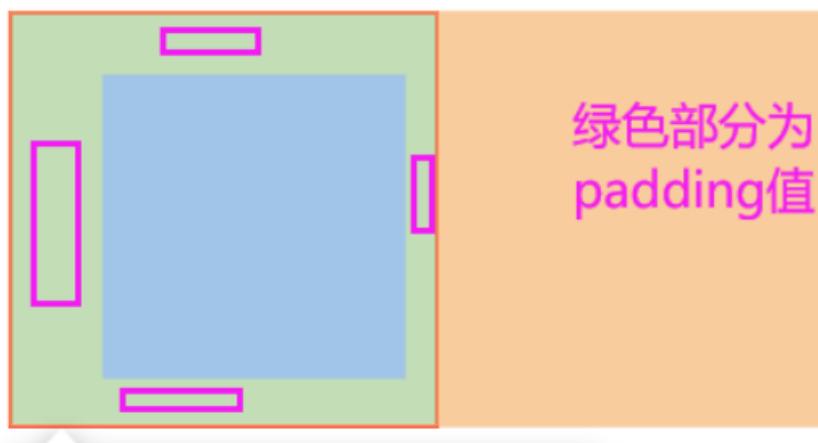
```

1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>三角形</title>
6      <style type="text/css">
7          div {
8              width: 0;
9              height: 0;
10             border: 100px solid black;
11             border-left-color: black;
12             border-top-color: transparent;
13             border-right-color: transparent;
14         }
15     </style>
16 </head>
17 <body>
18     <div></div>
19 </body>
20 </html>

```

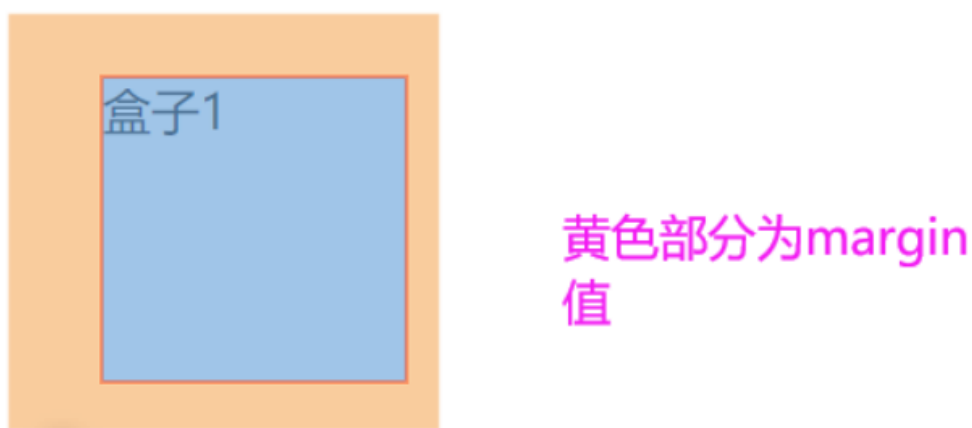
## 内边距 ( Padding )

元素内容与边框之间是可以设置距离的，称之为内边距或填充。内边距也可分为上、右、下、左（顺时针）。



## 外边距 ( Margin )

元素与其他元素之间的距离可以使用外边距来设置，外边距也可分为上、右、下、左。



`margin` 与 `padding` 的区别在于，`padding` 在外边框里，`margin` 在外边框外。

## 9.3 布局模型

### 布局模型

布局模型建立在盒子模型的基础上，在网页中，元素有3种布局模型：

1. 流动模型 ( flow )
2. 浮动模型 ( float )
3. 层模型 ( layer )

### 流动模型

流动模型是默认的网页布局模式，也就是说网页在默认状态下的HTML网页元素都是根据流动模型来分布网页内容的。

流动布局模型有2个典型的特征：

1. 块级元素都会在所处的包含元素内自上而下按顺序垂直延伸分布，因为在默认状态下，块级元素的宽度都为 100%，即都会以行的形式占据位置。

我是一个div标签

我是一个p标签

我是h1标签

2. 内联元素都会在所处的包含元素内从左到右水平分布显示。

我是一个a标签 我是一个span标签 我是一个strong标签

### 浮动模型

块级元素这么霸道都是独占一行，如果想让两个块级元素并排显示，可以设置元素浮动。任何元素在默认情况下是不能浮动的，但可以设置 float 属性定义为浮动，如 <div>、<p>、<table>、<img> 等元素都可以被定义为浮动。

#### 【代码】浮动模型

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>浮动模型</title>
6      <style type="text/css">
7          div {
8              width: 100px;
9              height: 100px;
10             border: 1px solid red;
11         }
12
13         #div1 {
14             float: left;
```



```
15     }
16
17     #div2 {
18         float: right;
19     }
20 </style>
21 </head>
22 <body>
23     <div id="div1"></div>
24     <div id="div2"></div>
25 </body>
26 </html>
```

---

## 9.4 定位

### 层模型

层布局模型就像是图像软件PhotoShop中非常流行的图层编辑功能一样，每个图形能够精确定位操作（positioning）。由于有些元素是定点展示的，定位技术可以让元素在特定的位置出现。

层模型有3种形式：

1. 绝对定位（absolute）
2. 相对定位（relative）
3. 固定定位（fixed）

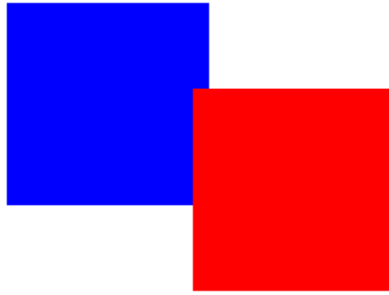
### 万事无绝对——绝对定位

如果想为元素设置层模型中的绝对定位，需要设置 `position: absolute`，这条语句的作用是将元素从文档流中拖出来，然后使用 `left`、`right`、`top`、`bottom` 属性相对于其最接近的一个具有定位属性的父包含块进行绝对定位。如果不存在这样的包含块，则相对于 `body` 元素，即相对于浏览器窗口。

当一个元素成为绝对定位元素，它就脱离了原来的层面，原来的位置会空出，下面的元素都会上来。

#### 【代码】绝对定位

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>绝对定位</title>
6      <style type="text/css">
7          div {
8              width: 100px;
9              height: 100px;
10         }
11
12         #div1 {
13             background-color: red;
14             position: absolute;
15             top: 50px;      /* 距离窗口上边50px */
16             left: 100px;   /* 距离窗口左边100px */
17         }
18
19         #div2 {
20             background-color: blue;
21         }
22     </style>
23 </head>
24 <body>
25     <div id="div1"></div>
26     <div id="div2"></div>
27 </body>
28 </html>
```

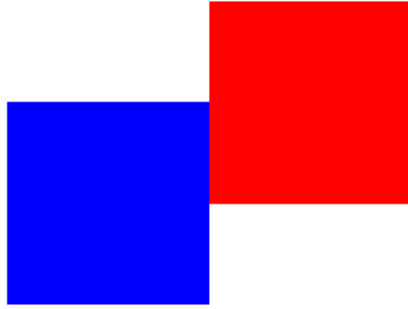


## 相对于自己的位置——相对定位

如果想为元素设置层模型中的相对定位，需要设置 `position: relative`，它通过 `left`、`right`、`top`、`bottom` 属性确定元素在正常文档流中的偏移位置。相对定位完成的过程是首先按 `static` (`float`) 方式生成一个元素，然后相对于以前的位置移动，并且偏移前的位置保留不动，因此下面的元素无法上来。

### 【代码】相对定位

```
1  <!DOCTYPE html>
2  <html lang="en">
3  <head>
4      <meta charset="UTF-8">
5      <title>相对定位</title>
6      <style type="text/css">
7          div {
8              width: 100px;
9              height: 100px;
10         }
11
12         #div1 {
13             background-color: red;
14             position: relative;
15             top: 50px;
16             left: 100px;
17         }
18
19         #div2 {
20             background-color: blue;
21         }
22     </style>
23 </head>
24 <body>
25     <div id="div1"></div>
26     <div id="div2"></div>
27 </body>
28 </html>
```



使用 `position: absolute` 可以实现元素相对于浏览器 ( `body` ) 设置定位，那可不可以相对于其它元素进行定位呢？当然可以，但是必须要满足以下的3点条件：

1. 参照定位的元素必须是相对定位元素的前辈元素。
2. 参照定位的元素必须加入 `position: relative`。
3. 定位元素加入 `position: absolute` 后便可以使用 `top`、`bottom`、`left`、`right` 来进行偏移了。

#### 【代码】相对父元素定位

```
1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>相对父元素定位</title>
6     <style type="text/css">
7         #div1 {
8             width: 100px;
9             height: 100px;
10            background-color: red;
11            position: relative;
12        }
13
14        #div2 {
15            width: 100px;
16            height: 20px;
17            background-color: blue;
18            position: absolute;
19            top: 40px;
20        }
21    </style>
22 </head>
23 <body>
24     <div id="div1">
25         <div id="div2"></div>
26     </div>
27 </body>
28 </html>
```



通过设置 `position: fixed` 让一个元素固定在一个位置，它的相对移动的坐标是视图（屏幕内的网页窗口）本身。由于视图本身是固定的，它不会随着浏览器窗口的滚动条滚动而变化，除非在屏幕中移动浏览器窗口的屏幕位置，或改变浏览器窗口的显示大小。因此固定定位的元素会始终位于浏览器窗口内视图的某个位置，不会受文档流影响。

【代码】固定定位

[illegible]

【代码】奥运五环



```

1 <!DOCTYPE html>
2 <html lang="en">
3 <head>
4     <meta charset="UTF-8">
5     <title>奥运五环</title>
6     <link rel="stylesheet" type="text/css" href="olympic.css">
7 </head>
8 <body>
9     <!-- 展示奥运五环的区域 -->
10    <div class="stage">
11        <div id="circle1"></div>
12        <div id="circle2"></div>
13        <div id="circle3"></div>
14        <div id="circle4"></div>
15        <div id="circle5"></div>
16    </div>
17 </body>
18 </html>

```

- style.css

```

1 * {
2     margin: 0;
3     padding: 0;
4 }
5
6 .stage {
7     position: absolute;
8     left: 50%;
9     top: 50%;
10    margin-left: -190px;
11    margin-top: -93px;
12    height: 185px;
13    width: 380px;
14 }
15
16 #circle1,
17 #circle2,
18 #circle3,
19 #circle4,
20 #circle5 {
21     position: absolute;
22     width: 100px;
23     height: 100px;
24     border: 10px solid black;
25     border-radius: 50%;
26 }
27
28 #circle1 {
29     border-color: blue;
30     left: 0;
31     top: 0;
32 }
33
34 #circle2 {

```

```
35     border-color: black;
36     left: 130px;
37     top: 0;
38 }
39
40 #circle3 {
41     border-color: red;
42     left: 260px;
43     top: 0;
44 }
45
46 #circle4 {
47     border-color: yellow;
48     left: 65px;
49     top: 70px;
50 }
51
52 #circle5 {
53     border-color: green;
54     left: 195px;
55     top: 70px;
56 }
```

---

## 第10章 JavaScript介绍

---







