# CSS

> CSS值层叠样式表（Cascading Style Sheets）
>
> 样式定义如何显示HTML元素
>
> 样式通常存储在样式表中
>
> 外部样式表可以极大的提高工作效率
>
> 外部样式表通常存储后缀为CSS的文件中
>
> 多个样式定义可层叠为一

- 有了CSS，html中大部分表现样式的标签就可以不用了
- html只负责文档的结构和内容，表现形式完全交给CSS，html文档变得更加简洁

## CSS的引入方式

- 内联式引入：直接赋予标签style属性进行样式编写
  - ```
    <body style="background: #00ff00">
    ```
- 嵌入式：直接在文档页面通过style标签创建嵌入的样式表
  - ```
    <style type="text/css">
        body{
            background: black;
        }
    </style>
    ```
- 外部式：在文档中通过link标签，将外部样式文件引入到页面中：
  - ```
    <link rel="stylesheet" type="text/css" href="css/main.css">
    ```
- 优先级：内联式 > 嵌入式 > 外部式 （就近原则）

---

## CSS语法

- CSS语法规则由两部分构成：选择器 & 一条或多条声明
  - ```
    selector{
        declaration1;
        declaration2;
    }
    ```

```
selector:
    选择器，一般是你需要改变的HTML标签
declaration1:
    每条声明，由一个属性和一个值组成
```

  - ```
    selector{
        property: value;
    }
    ```
    属性（property）是您希望设置的样式属性（style attribute）
    每个属性有一个值。属性和值被冒号分开

- 当属性中的值为若干单词时，加引号
  - ```
    h1{
        font-family: "sans serif";
    }
    ```

- 颜色的写法：
  - ```
    p{
        color: #00FF00;
    }
    ```

  - ```
    h1{
        color: #0F0
    }
    ```

  - ```
    body{
        background: rgb(255,0,0);
        background: rgb(100%,0%,0%);
    }
    ```
    请注意，当使用 RGB 百分比时，即使当值为 0 时也要写百分比符号
    当尺寸为 0 像素时，0 之后不需要使用 px 单位，因为 0 就是 0，无论单位是什么

---

# CSS选择器

## 标签选择器

> 影响范围大，常做通用设置，或用在层级选择器中

- ```
  <p>第一段文字</p>
  <p>再来一段</p>
  ```

- ```
  p{
      color: blue;
  }
  /*定义所有p标签字体为蓝色*/
  ```

---
  ```

## 类选择器

> 通过类名来选择元素，一个类可以用于多个元素；
>
> 一个元素也可以使用多个类，应用灵活，可复用；
>
> 是CSS中使用最多的一种选择器

- > **注意**：类名的第一个字符不能使用数字

- ```
  第一段文字
  再来一段
  ```

- ```
  .big{
      font-size: 20px
  }
  /*定义big类字体大小为20像素*/
  .red{
      color: red;
  }
  /*定义类为red的字体颜色为红色*/
  ```

- 类选择器还可基于所属标签进行派生选择：
  - ```
    再来一段

        文字测试

    ```
  - ```
    div.red{
        color: red;
    }
    /*在页面中只有div类为red的字体颜色会变为红色*/
    ```

---

## 属性选择器

> 对指定属性的HTML元素进行设置，而不限于class和id属性

- 通过属性
  - ```
    <a href="https://www.baidu.com">这是一个连接</a>
    测试
    ```
  - ```
    [href]{
        color: blue;
    }
    /*为所有具有href属性的元素设置字体颜色*/
    ```
- 通过属性的值（整个匹配）

```
<a href="https://www.baidu.com">这是一个连接</a>
<p href="test">测试</p>
```

```
[href="test"]{
    font-size: 20px;
}
/*为属性为test的元素设置字体大小*/
```

- 属性的值（属性的值中包含所匹配的单词）：~=

```
<p attr="test">测试</p> √
<p attr="test-xx">测试</p> x
<p attr="test_xx">测试</p> x
<p attr="test xx">测试</p> √
```

```
[attr~=test]{
    font-size: 20px;
}
/*为属性包含test的元素设置字体大小*/
```

> **注意**：不包含下划线和连字符的

- 属性的值（从开头整个匹配或带有连字符的属性值）：|=

```
<p attr="test">测试</p> √
<p attr="test-xx">测试</p> √
<p attr="test_xx">测试</p> x
<p attr="test xx">测试</p> x
```

```
[attr|=test]{
    font-size: 20px;
}
/*为单词为test或开头为test-的元素设置字体大小*/
```

> **注意**：适用于由连字符分隔的属性值

- 属性的值（从属性值的开头进行匹配）：^=

```
<p attr="test">测试</p> √
<p attr="test-xx">测试</p> √
<p attr="test_xx">测试</p> √
<p attr="test xx">测试</p> √
```

```
[attr^=test]{
    font-size: 20px;
}
/*为开头包含test的所有元素设置字体大小*/
```

- 属性的值（从属性值的结尾开始匹配）：$=

    - ```
      测试 √
      测试 √
      测试 √
      测试 √
      ```
    - ```
      [attr$=test]{
          font-size: 20px;
      }
      /*为结尾是test的所有元素设置字体大小*/
      ```
- 属性的值（只要含有则匹配）：*=
    - ```
      测试 √
      测试 √
      测试 √
      测试 √
      ```
    - ```
      [attr*=test]{
          font-size: 20px;
      }
      /*为含有test的所有元素设置字体大小*/
      ```
- 几种属性选择的匹配方式：
    - ~=：用于选取属性值中包含指定**词汇**的元素；
        - 必须是单独的词汇，不能是带有连字符或下划线组成的单词。
    - |=：用于选取带有以指定值开头的属性值的元素，该值必须是整个单词；
        - 可以有连字符组成，word或者是word-wild
    - ^=：匹配属性值以指定值开头的每个元素
    - $=：匹配属性值以指定值结尾的每个元素
    - *=：匹配属性值中包含指定值的每个元素

---

## 层级选择器

> 主要应用在标签嵌套的结构中，通过层级，限制样式的作用范围

- ```

      标题1
      作者1

          主要内容

      标题2
      作者2

  ```
  ```

```
            主要内容2
        </p>
    </div>
```

- ```
  .header .title{
      color: gold;
      font-size: 30px;
  }
  /*生效所有header类下的title类*/
  .header .author{
      color: blue;
      font-size: 15;
  }
  /*生效所有header类下的author类*/
  .header p{
      font-weight: bold;
  }
  /*生效所有header类中的p标签*/
  ```

---

## ID选择器

> 通过ID名进行元素选择，元素的ID名定义时在整个文档属于唯一
>
> 通过ID选择器只能对应页面元素中的一个
>
> ID名通常作为JS脚本定位使用，不推荐ID选择器

- ```
  <div class="header">
      <p id="title">标题1</p>
      <span id="author">作者1</span>
      <p id="content">
          主要内容
      </p>
  </div>
  ```

- ```
  #title{
      font-size: 25px;
      font-weight: bold;
  }
  #author{
      font-size: 10px;
  }
  #content{
      color: blue;
  }
  ```

---

## 伪类选择器

> CSS伪类选择器用于向某些选择器添加特殊的效果
  ```

- 伪类的语法
  - ```
    selector:pseudo-class{
        property: value;
    }
    ```
- 与css类搭配使用
  - ```
    selector.class:pseudo-class{
        property: value;
    }
    ```

---

**锚伪类**

> 控制连接访问状态，常见状态有：活动状态、已访问状态、未被访问状态、鼠标悬停状态

- ```
  <a class="baidu" href="https://www.baidu.com">访问这里</a>
  ```
- ```
  .baidu:link{
      color: blue;
  }
  /*未访问过的连接*/
  .baidu:visited{
      color: black;
  }
  /*访问过的连接*/
  .baidu:hover{
      color: gold;
  }
  /*鼠标划过的连接*/
  .baidu:active{
      color: red;
  }
  /*已选中的连接*/
  ```
- link、visited、hover、active

  > **注意**：
  >
  > hover必须定义在link和visited之后，才是有效的；
  >
  > active必须定义在hover之后，才是有效的

---

## CSS选择器的权重

> 当有多个同类样式作用于同一个元素时：
>
> 1. 权重高的样式对元素起作用
> 2. 权重相同时后写的样式覆盖前面写的

- 使用!important将样式权重设置为10000，将!important写到样式属性值后

- 权重值：就近原则
  - 内联式样式：1000
    - ```

      ```
  - ID选择器：100
    - ```
      #id {color: red;}
      ```
  - 类选择器：10
    - ```
      .class { background: blue;}
      ```
  - 标签选择器：1
    - ```
      p{font-weight:bold;}
      ```
- ```
  测试
  ```
- ```
  #test{
      color: red !important;
  }
  ```

---

# CSS基本属性

---

## 布局属性

- **width**：设置元素（标签）的宽
- **height**：设置元素（标签）的高
  - ```
    #button{width:100px; height:100px;}
    ```
- **background**：设置背景色或背景图
  - ```
    body{
        width: 100%;
        height: 100%;
        background: #00FF00 url("../img/1.jpg") no-repeat ;
    }
    ```

```
background属性可以分解为如下几个设置项
- background-color: 设置背景颜色
- background-image: 设置背景图片地址
    url(路径)
- background-repeat: 设置图片如何重复平铺
    repeat、repeat-x(水平方向重复)、repeat-y(垂直方向重复)、no-repeat(图片只显示一次)
- background-position: 设置图片的位置
    left、right、center
- background-attachment: 设置图片是固定的还是会随页面滚动
    scroll(背景图片会随着页面其余部分的滚动而移动)、fixed(页面的其余部分滚动时，背景图像不会移动)、
```

  - > **注意**：在背景图片路径填写时，如果使用内联式写法则从当前页面路径开始查找相对路径，如写在外部CSS文件中，则以CSS文件为相对基础。

- **border**：设置元素周围的边框
  - ```
    p{
        border: 10px double blue;
    }
    ```
  - 依次设置：border-width、border-style、border-color
  - 也可以拆分成四个边的样式选项
    - ```
      - border-top: 顶边框
      - border-buttom: 底边框
      - border-left: 左边框
      - border-right: 右边框
      ```
    - ```
      [attr]{
          border-top: 5px inset blue;
          border-bottom: 5px inset green;
          border-left: 5px outset red;
          border-right: 5px outset red;
      }
      ```
  - 设置时提供的边框样式属性：
    - ```
      - dotted: 点状
      - solid: 实线
      - double: 双线
      - dashed: 虚线
      - groove: 3D 凹槽边框
      - ridge: 定义 3D 垄状边框
      - inset: 定义 3D inset 边框
      - outset: 定义 3D outset 边框
      - inherit: 规定应该从父元素继承边框样式
      ```
  - ```
    aaaaaaa
    bbbbbbb
    ccccccc
    ```
    ```

```
<p class="p4">ddddddd</p>
<p class="p5">ggggggg</p>
<p class="p6">eeeeeee</p>
<p class="p7">fffffff</p>
<table class="table1" border="1">
    <tr>
        <th>姓名</th>
        <th>性别</th>
        <th>年纪</th>
    </tr>
    <tr>
        <td>张三</td>
        <td>女</td>
        <td>18</td>
    </tr>
    <tr>
        <td>李四</td>
        <td>男</td>
        <td>20</td>
    </tr>
</table>
```

- ```
  .p1{
      border: 1px dotted blue;
  }
  .p2{
      border: 1px solid blue;
  }
  .p3{
      border: 1px double blue;
  }
  .p4{
      border: 1px dashed blue;
  }
  .p5{
      border: 10px groove green;
  }
  .p6{
      border: 10px ridge blue;
  }
  .p7{
      border: 10px inset blue;
  }

  .table1{
      border: 5px double red;
  }
  ```

- **padding**：设置元素包含的内容和元素边框的距离，也叫**内边距**
  ```

```
p{
    padding: 100px;
    border: 1px solid black;
}
```

- 这个样式属性也可以拆分成以下单独四种，可以分别设置对应位置的内边距

```
- padding-top: 设置上内边距
- padding-bottom: 设置下内边距
- padding-left: 设置左内边距
- padding-right: 设置右内边距
```

- **margin**：设置元素和外界的边距，也叫**外边距**

```
p{
    margin: 10px;
}
/*同时设置四个边距为10px*/
```

- 与padding类似，margin属性也可以拆分为四个方向的单独设置

```
- margin-top: 设置上外边距
- margin-bottom: 设置下外边距
- margin-left: 设置左外边距
- margin-right: 设置右外边距
```

- **float**：定义元素在当前父元素下向哪个方向浮动，这个属性常用于图像，使文本围绕在图像周围

> 如果浮动方向空间不足，元素会跳至下一行，这个过程会持续到某一行拥有足够的空间为止

```
<input type="submit" value="提交">
```

```
input{
    float: left;
}
```

## 文本常用属性

- **color**：设置元素中的文字颜色

```
p{color: red;}
```

- **font-size**：设置元素中的文字大小

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

- **font-family**：设置元素中的文字字体

```
p{font-family:"微软雅黑"}
/*为了避免中文兼容问题，常用字体的英文标识*/
p{font-family:"Microsoft Yahei"}
```

- **font-widght**：设置元素中的文字是否加粗
  - ```
    p{font-widght:bold;}
    /*设置加粗*/
    p{font-widght:normal;}
    /*设置不加粗*/
    ```
- **line-height**：设置元素中的文字行高
  - ```
    p{line-height:24px;}
    ```
- **text-decoration**：设置元素中文字的下划线
  - ```
    p{text-decoration:underline;}
    ```
  - ```
    - none：默认文本格式，无下划线
    - underline：定义文本下的一条线
    - overline：定义文本上的一条线
    - line-through：定义穿过文本的一条线
        - blink：定义闪烁的文本
    - inherit：规定应该从父元素继承 text-decoration 属性的值
    ```

---

- **text-align**：设置元素中文字对齐方式
  - ```
    p{text-align:center;}
    ```
- **text-indent**：设置元素中文字的首行缩进
  - ```
    p{text-indent:24px;}
    ```
- **display**：设置元素的类型及隐藏方式
  - ```
    p{display:none;}
    ```
  - ```
    - none：元素不会显示
    - block：元素将显示为块级元素，此元素前后会带有换行符
    - inline：此元素被显示为内联元素，元素前后没有换行符
    - list-item：元素作为列表显示
    - table：元素作为块级表格来显示（类似 <table>），表格前后带有换行符
    - inline-table：元素作为内联表格来显示（类似 <table>），表格前后没有换行符
    - table-cell：此元素会作为一个表格单元格显示（类似 <td> 和 <th>）
    - table-caption：此元素会作为一个表格标题显示（类似 <caption>）
    ```

## 元素溢出

- **overflow**：当子元素的大小超过所承载的父元素大小时，需要设置父元素对于溢出的子元素显示方式
  - ```
    123456789
    ```
    ```

```
p{
    width:500px;
    text-indent: 498px;
    border: 1px solid blue;
    overflow:auto;
}
```

- ```
  - visible: 默认值;内容不会被修剪，会呈现在元素框之外
  - hidden: 内容会被修剪，并且其余内容是不可见的
  - scroll: 内容会被修剪，但是浏览器会显示滚动条以便查看其余的内容
  - auto: 如果内容被修剪，则浏览器会显示滚动条以便查看其余的内容
  ```

## 盒子模型

> 使用浏览器F12查看元素

## 定位

- **文档流**：文档流，是指盒子按照html标签编写的顺序依次从上到下，从左到右排列，块元素占一行，行内元素在一行之内从左到右排列，先写的先排列，后写的排在后面，每个盒子都占据自己的位置。
- CSS3主要有三种定位：**普通流、浮动、绝对定位**
- **static**：元素框正常生成，块级元素生成一个矩形框、作为文档流的一部分、行内元素则会创建一个或多个行框，置于其父元素中
- **relative**：相对定位元素，元素还会保持定位前的形状，并且移动前的位置也会保留下来，不会脱离文档流
  - 一般是将父级设置相对定位（relative），子级设置绝对定位（absolute），子级就以父级作为参照来定位，否则子级相对于body来定位
  - **相对定位会按照元素的原始位置对该元素进行移动**
- **absolute**：绝对定位元素，元素脱离文档流，移动前的位置在文档流中关闭，定位后生成一个新的块级框，不论他之前在原始文档流中生成何种类型的框
  - 也可以理解为漂流在文档流的上方，相对于上一个设置了定位的父级元素来进行定位，如果找不到，则相对于body元素进行定位
  - **通过绝对定位，元素可以放置到页面上的任何位置**
- **fixed**：固定定位元素，元素脱离文档流，不占据文档流的位置，相对于浏览器窗口进行定位
- 元素偏移的设置：
  - ```
    - top: 定位元素的上外边距边界与其包含块上边界之间的偏移
    - bottom: 定位元素下外边距边界与其包含块下边界之间的偏移
    - right: 定位元素右外边距边界与其包含块下边界之间的偏移
    - left: 定位元素左外边距边界与其包含块下边界之间的偏移
    - z-index: 设置堆叠元素的层级，这里的层级不是从上到下，而是从里到外
    ```

- ```
  body{
      padding: 0px;
      margin: 0px;
  }
  .container{
      position: relative;
  ```
  ```

```
    width: 100%;
    height: 100%;
    background: black;
    margin: 0px;
}
.top{
    position: relative;
    width: 100%;
    height: 20%;
    background: yellow;
}
.left{
    width: 20%;
    height: 70%;
    background: green;
}
.right{
    position: absolute;
    top: 20%;
    left: 30%;
    width: 70%;
    height: 70%;
    background: red;
}
.bottom{
    position: absolute;
    top: 90%;
    width: 100%;
    height: 10%;

    background: white;
}
```

- ```
  <html>
  <head>
  </head>

  <body>

          123

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