一. CSS简介

1. CSS概述

虽然Html标签的属性可以调整一些样式，但是效果不够理想，而我们更愿意把样式编写在 `<style>` 标签中，让页面设计更美观更丰富，实际上，这是通过CSS实现的。那么什么是CSS呢?

简单来说，HTML负责网页内的基础内容，而CSS负责让网页美观。

只有HTML的效果:

添加CSS以后:

CSS (**层叠样式表**——Cascading Style Sheets，缩写为 **CSS**），**简单的说，它是用于设置和布局网页的计算机语言。会告知浏览器如何渲染页面元素。例如，调整内容的字体，颜色，大小等样式，设置边框的样式，调整模块的间距等。**

CSS作用：简单来说：就是通过编码方式，对Html页面的各种标签元素进行样式控制。

样式表定义如何显示 HTML 元素，就像 HTML的字体标签和颜色属性所起的作用那样。样式通常保存在外部的 .css 文件中。通过仅仅编辑一个简单的 CSS 文档，外部样式表使你有能力同时改变站点中所有页面的布局和外观。

由于允许同时控制多重页面的样式和布局，CSS 可以称得上 WEB 设计领域的一个突破。作为网站开发者，你能够为每个 HTML 元素定义样式，并将之应用于你希望的任意多的页面中。如需进行全局的更新，只需简单地改变样式，然后网站中的所有元素均会自动地更新。

二. CSS基础语法

1. CSS三种引入方式

- 外部样式表（.css文档中定义）
- 内部样式表（head中定义）
- 内联样式（在标签上定义）

> **三种方式的优先级： 内联样式 > 内部样式表 = 外部样式表(后面两种优先级取决于定义的顺序，越靠后优先级越高)**

2. CSS语法

CSS 规则由两个主要的部分构成：

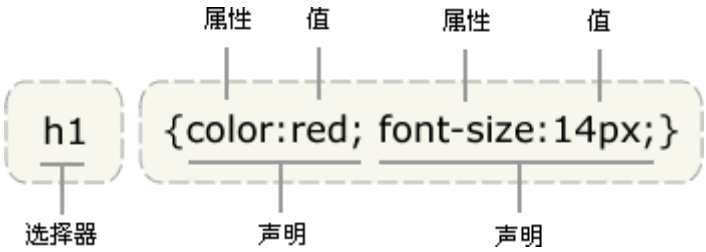
- **选择器**：指定将要添加样式的 HTML元素的方式。可以使用标签名，class值，id值等多种方式。
- **声明**：形式为**属性(property):值(value)**，用于设置特定元素的属性信息，可以设置多个声明，多个声明之间用英文的分号";"隔开。

```
selector{  // 选择器
    property1:value1;  //声明
    property2:value2;
    property3:value3;
}
```

示例：

```
/* 给h1标签设置前景色为红色，字体大小14像素 */
h1{
    color:red;
    font-size:14px;
}
```

示例示意图：



3. 入门案例

- 需求

  设置h1标签内容的颜色。

- 准备工作

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>入门案例</title>
    </head>
    <body>
        <h1>欢迎来到Css的世界</h1>
    </body>
</html>
```

- 样式设置
  - 内联样式

```
<!-- 内联样式 -->
<h1 style="color: red;">欢迎来到Css的世界</h1>
```

  - 内部样式

    在head标签中添加style标签

```
<!-- 内部样式 -->
<style>
    h1{
        /* color: red; */ /*英文单词写法*/

        /* color: #ff0000; */ /*十六进制写法，可以使用缩写#ff00*/
        /* color: #ff00; */ /*十六进制缩写写法*/

        /* color: rgb(255, 0, 0); */ /*rgb写法，rgb颜色值分为三组，第一组代表红色、第二组代表绿色、第三组代表蓝色。每一组的取值范围是0 - 255*/
        /* color: rgb(100%,0%,0%); */ /*rgb百分比写法*/
        color: rgb(100%,0%,0%,.1);  /*rgb百分比写法，带透明度*/
    }
</style>
```

> 注意：在使用内部样式时，需要将内联样式去掉，因为内联样式方式的优先级高于内部样式方式和外部样式方式。

- 外部样式

在项目的根目录创建css目录，在css目录中创建start.css文档，在文档中设置样式。

```
h1{
    /* color: red; */ /*英文单词写法*/

    /* color: #ff0000; */ /*十六进制写法，可以使用缩写#ff00*/
    /* color: #ff00; */ /*十六进制缩写写法*/

    /* color: rgb(255, 0, 0); */ /*rgb写法，rgb颜色值分为三组， 第一组代表红色、第二组代表绿色、第三组代表蓝色。每一组的取值范围是0 - 255*/
    /* color: rgb(100%,0%,0%); */ /*rgb百分比写法*/
    color: rgb(100%,0%,0%,.1);  /*rgb百分比写法，带透明度*/
}
```

在html文件的head标签中使用link标签引入外部的css文件。

```
<link type="text/css" rel="stylesheet" href="css/start.css"/>
```

> 注意：在使用外部的css样式时，需要将之前的内部样式去掉，因为内部样式方式和外部样式方式的优先级取决于定义顺序，越往后优先级越高。

4. CSS选择器

4.1. 基础选择器

- 准备工作

  在项目的css目录中创建selector01.css文档。

  创建Html页面，引入selector01.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS基础选择器</title>
        <link type="text/css" rel="stylesheet"
href="css/selector01.css"/>
    </head>
    <body>
        <ul>
            <li><h1>CSS</h1></li>
            <li class="grey">CSS简介</li>
            <li id="css_selector">CSS选择器</li>
            <li class="grey">CSS属性</li>
            <li my_attr="anli">CSS案例</li>
            <li id="yingyong">CSS应用</li>
        </ul>
    </body>
</html>
```

- 通配选择器

  可以匹配页面中的所有标签

  语法：

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

- 标签选择器

  根据标签名称选中页面中指定的标签

```
/* margin-left：设置元素的左外边距 */
ul{
    margin-left: 40px;
}
h1{
    color: blue;
}
```

- ID选择器

  根据标签上的ID属性值来选中指定的标签

```
#css_selector{
    color: yellow;
}
```

- 类选择器

  根据标签上的class属性值来选中所有带此class属性的标签

```
.grey{
    color: hotpink;
}
```

- 属性选择器

  属性选择器可以根据标签上所携带的属性来选中标签

```
/* 如果给同一个标签设置多个相同样式，后设置的会生效 */
/* [属性名]{}  拥有此属性的标签都会设置成对应的样式 */
[my_attr]{
    color: orange;
}
/* [属性名="值"]{}  拥有此属性并且属性的值是对应的值的标签都会设置成对应的样式 */
[my_attr="anli"]{
    color: gainsboro;
}
[id="yingyong"]{
    color: aqua;
}
```

4.2. 进阶选择器

进阶选择器其实是将多个基础选择器组合到一起使用。

- 准备工作

  在项目的css目录中创建selector02.css文档。

  创建Html页面，引入selector02.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS进阶选择器</title>
        <link type="text/css" rel="stylesheet"
href="css/selector02.css"/>
    </head>
    <body>
        <div id="level1" class="level1_class">
            <h1>level1_标题1</h1>
```

```
            <h2>level1_标题2</h2>
            <h3>level1_标题3</h3>
            <h4>level1_标题4</h4>
            <h5>level1_标题5</h5>
            <h6>level1_标题6</h6>
            <div id="level11" class="level11_class">
                <h1>level11_标题1</h1>
                <h2>level11_标题2</h2>
                <h3>level11_标题3</h3>
                <h4>level11_标题4</h4>
                <h5>level11_标题5</h5>
                <h6>level11_标题6</h6>
            </div>
        </div>
        <div id="level2" class="level2_class">
            <p>level2_标题</p>
        </div>
        <p>level3_标题</p>
        <p>level4_标题</p>
        <p>level5_标题</p>

        <span id="test_id" class="test_class" attr="123">CSS样式优先级
</span>
    </body>
</html>
```

- 后代选择器

  语法：

```
selector1 selector2 ...selectorN{
    .......
}
```

  使用示例：设置ID为level1的标签下面所有h2子元素的样式。

```
#level1 h2{
    color: red;
}
```

- 子代选择器

  语法：

```
selector1>selector2....selectorN{
    .......
}
```

  > 和后台选择器的区别在于：只选择直接子代元素

使用示例：设置ID为level1的标签下面的h3子元素的样式。

```
#level1>h3{
    color: blue;
}
```

- 同级选择器

  语法：

  ```
  selector1~selector2...selectorN{
      .......
  }
  ```

  使用示例：选中class="level1_class"的元素后面所有的同级p标签。

  ```
  .level1_class~p{
      color: green;
  }
  ```

- 相邻兄弟选择器

  语法：

  ```
  selector1+selector2...selectorN{
      .......
  }
  ```

  使用示例：选中所有class="level2_class"的元素后面紧跟着的同级p标签

  ```
  .level2_class+p{
      background: yellow;
  }
  ```

- 伪类选择器
  - :hover

    语法：

    ```
    selector:hover{
       .......
    }
    ```

使用示例：当鼠标放到所有的p标签上时，将p标签的样式改变，当鼠标移开时，样式还原

```
p:hover {
      background: lightpink;
      color: white;
}
```

- 伪元素

  在selector元素内部的开头或结尾插入一个伪元素

  语法：

  ```
  selector:before{
      .......
  }
  selector:after{
      .......
  }
  ```

  使用示例：在class="level2_class"的元素内部的开头位置和结束位置添加一个伪元素，并且设置伪元素的内容、文字大小、文字粗度

  ```
  .level2_class:before{
      content: "before伪元素标题";
      font-size: 60px;
      font-weight: bold;
  }
  .level2_class::after{
      content: "ater伪元素标题";
      font-size: 60px;
      font-weight: bold;
  }
  ```

  常用的伪元素有两个：

  - before：在标签的所有内容的前面添加内容。
  - after：在标签的所有内容的后面添加内容。

4.3. 选择器分组

可以通过使用英文的逗号","将多个选择器连接到一起，同时给多个符合的元素设置统一的样式

例如：如果您想把很多元素显示为灰色，可以使用类似如下的规则

```
h6,.level1,#level2{
    background: gray;
}
```

4.4. 选择器优先级

优先级只有在单selector场景中比较才有意义。

优先级排名：

```
第一名：ID选择器
第二名：属性选择器/类选择器
第三名：标签选择器
第四名：通配选择器
```

测试代码：

```
/* 选择器优先级 */

/* id选择器 > 属性选择器=class选择器 > 标签选择器 > 通配符选择器 */

/* 第一 */
#test_id{
    color: yellow;
}

/* 第二 */
[attr="123"]{
    color: hotpink;
}

/* 第二 */
.test_class{
    color: blue;
}

/* 第三 */
span{
    color: green;
}

/* 第四 */
*{
    color: red;
}
```

三. CSS样式

1. 背景相关样式

- 准备工作

  在项目的css目录中创建css_background.css文档。

```
#css_background{
    width: 1200px;
    height: 800px;
}
```

  创建Html页面，引入css_background.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS背景样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
background.css"/>
    </head>
    <body>
        <div id="css_background">

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

- background-color：设置背景色

```
background-color: orange;
```

- background-image：设置背景图片

```
background-image: url(1.jpg);
```

- background-position：设置背景图片的起始位置，背景图像如果要重复，将从这一点开始

```
background-position:bottom center;
```

- background-repeat：设置背景图片是否及如何重复

```
/* 背景图片重复方式及是否重复  no-repeat：不能重复   repeat-x：横向重复
repeat-y：竖向重复*/
background-repeat: no-repeat;
```

- background-size：设置背景图片的尺寸

```
/* 背景图片大小 */
/* background-size: 800px 500px; */
background-size: 50% 50%;
```

- background-attachment：设置背景图片是固定还是随着页面的其余部分滚动

```
/* background-attachment: scroll; */
background-attachment: fixed;
```

- background: 简写属性，将上面列出的背景相关的多个属性设置在一个声明中（推荐）。

```
background: yellow url(1.jpg) bottom center no-repeat scroll;
```

2. 文本相关样式

- 准备工作

  在项目的css目录中创建css_text.css文档。

```
#css_text{
    width: 500px;
    height: 100px;
    background: yellow;
}
```

  创建Html页面，引入css_text.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS文本样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
text.css"/>
    </head>
    <body>
        <div id="css_text">
            CSS文本样式
        </div>
    </body>
</html>
```

- 文本缩进 text-indent

```
#css_text{
    text-indent: 100px;
}
```

- 水平对齐 text-align

```
/* left：水平左对齐 right：水平右对齐 center：水平居中对齐 */
text-align: center;
```

- 行高 line-height

  当块元素中只有一行文本时，将行高设置为容器的高度，可实现垂直方向上的居中

```
line-height: 100px;
```

- 词间隔 word-spacing

```
/* 词间隔，词语之间需要有空格 */
word-spacing: 2em;
```

- 字母间隔 letter-spacing

```
/* 字符间隔，不需要空格 */
letter-spacing: 2em;
```

- 大小写转换 text-transform

```
/* 大小写转化  lowercase：大写转小写  uppercase：小写转大写*/
text-transform: lowercase;
```

- 文本装饰 text-decoration

```
/* 文本装饰  line-through：定义穿过文本的一条线。overline：定义文本上的一条线  underline：定义文本下的一条线*/
text-decoration: line-through;
```

- 空白字符处理 white-space

```
/* 空白字符处理*/
white-space: nowrap;
```

- 文本阴影 text-shadow

```
/* 文本阴影  x距离,y距离,模糊,颜色*/
text-shadow: 2px 2px 8px red;
```

3. 字体相关样式

- 准备工作

  在项目的css目录中创建css_font.css文档。

```
#css_text{
    width: 500px;
    height: 100px;
    background: yellow;
}
```

  创建Html页面，引入css_font.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS字体样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
font.css"/>
    </head>
    <body>
        <div id="css_text">
            CSS字体样式
        </div>
    </body>
</html>
```

- 设置字体 font-family

```
font-family: Impact, Haettenschweiler, 'Arial Narrow Bold', sans-serif
```

- **字体大小 font-size**

```
font-size: 40px;
```

- **字体粗细 font-weight**

```
font-weight: 900;
```

- 字体风格 font-style

```
/*italic：斜体*/
font-style: italic;
```

4. 尺寸相关样式

- 准备工作

  在项目的css目录中创建css_wh.css文档。

  创建Html页面，引入css_wh.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS尺寸样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-wh.css"/>
    </head>
    <body>
        <div id="css_text">
            CSS尺寸样式
        </div>
    </body>
</html>
```

- **设置宽度 width**
- **设置高度 height**

```
width: 500px;
height: 100px;
```

- 设置最大宽度 max-width
- 设置最大高度 max-height

```
max-width: 300px;
max-height: 100px;
```

- 设置最小宽度 min-width
- 设置最小高度 min-height

```
min-width: 50px;
min-height: 20px;
```

5. 链接相关样式

我们可以通过伪类选择器来设置超链接不同状态的样式

- 链接的四种状态（本质上其实就是给a标签设置伪类选择器）：
  - a:link - 普通的、未被访问的链接
  - a:visited - 用户已访问的链接
  - a:hover - 鼠标指针位于链接的上方
  - a:active - 链接被点击的时刻

- 准备工作

  在项目的css目录中创建css_link.css文档。

  创建Html页面，引入css_link.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS链接样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
link.css"/>
    </head>
    <body>
        <a href="https://www.baidu.com">CSS链接样式</a>
    </body>
</html>
```

- 示例

```
/* 超链接没有被点击访问时的样式 */
a:link{
    color: antiquewhite;
    text-decoration: none; /*去除下划线*/
}
/* 鼠标悬停在超链接上时的样式 */
a:hover{
    color: red;
}

/*链接被点击时的样式*/
a:active{
    color: green;
}

/* 链接被访问之后的样式 */
a:visited{
    color: gainsboro;
}
```

6. 列表相关样式

- 准备工作

  在项目的css目录中创建css_list.css文档。

```
ul{

}
```

创建Html页面，引入css_list.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS列表样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
list.css"/>
    </head>
    <body>
        <ul>
            <li>CSS列表样式</li>
            <li>CSS简介</li>
            <li>CSS选择器</li>
            <li>CSS属性</li>
            <li>CSS案例</li>
            <li>CSS应用</li>
            <li>CSS项目</li>
            <li>CSS问题</li>
        </ul>
    </body>
</html>
```

- 列表标记的类型：list-style-type

```
list-style-type: decimal;
```

- 将列表符号设置为图像 list-style-image

```
list-style-image: url(1.jpg);
```

7. 表格相关样式

- 准备工作

  在项目的css目录中创建css_table.css文档。

  创建Html页面，引入css_table.css文件，内容如下：

```
<!DOCTYPE html>
<html>
    <head>
        <meta charset="utf-8">
        <title>CSS表格样式</title>
        <link type="text/css" rel="stylesheet" href="css/css-
table.css"/>
    </head>
    <body>
```

```
        <table>
            <caption>表格样式</caption>
            <tr>
                <td>姓名</td>
                <td>年龄</td>
            </tr>
            <tr>
                <td>张三</td>
                <td>30</td>
            </tr>
            <tr>
                <td>李四</td>
                <td>50</td>
            </tr>
        </table>
    </body>
</html>
```

- 设置边框线宽度
- 设置边框线颜色
- 设置边框线样式

```
table{
    border: 1px;
    border-color: red;
    border-style: solid;
}

td{
    border: 1px;
    border-color: red;
    border-style: solid;
}
```

- 合并重复边框 border-collapse

```
border-collapse: collapse;
```

- 单元格与内容之间空白的距离 border-spacing

```
/*设置border-collapse: collapse时失效 */
border-spacing: 10px;
```

- 表格标题的位置 caption-side

```
caption-side: bottom;
```

- 上下左右边框样式设置

```
/* 左边框样式设置 */
border-left: 5px;
border-left-color: green;
border-left-style: dotted;

/* 右边框样式设置 */
border-right: 10px;
border-right-color: blue;
border-right-style: groove;

/* 上边框样式设置 */
border-top: 5px;
border-top-color: red;
border-top-style: dashed;

/* 下边框样式设置 */
border-bottom: 5px;
border-bottom-color: yellow;
border-bottom-style: solid;
```

四. 修改元素默认显示行为

我们可以通过给html元素设置display样式来改变其默认的显示行为，display的常用取值如下：

- 转化为行内元素(内联元素)显示： inline

```
display: inline;
```

- 转化为块级元素显示：block

```
display: block;
```

- 转化为行内块显示：inline-block

```
display: inline-block;
```

- 不显示：none

```
display: none;
```

五. 清除默认样式

不同的浏览器默认的样式可能不尽相同，所以开发时的第一件事可能就是如何把它们统一。如果没清除默认的CSS样式往往会出现浏览器之间的页面差异。

每次新开发网站或新网页时候通过初始化CSS样式的属性，可以让用到的CSS或html标签更加方便准确，使得我们开发网页内容时更加方便简洁，同时减少CSS代码量，节约网页下载时间。

```
html {
    color: #000;
    background: #FFF;
}

body,
div,
dl,
dt,
dd,
ul,
ol,
li,
h1,
h2,
h3,
h4,
h5,
h6,
pre,
code,
form,
fieldset,
legend,
input,
textarea,
p,
blockquote,
th,
td {
    margin: 0;
    padding: 0;
}

table {
    border-collapse: collapse;
    border-spacing: 0;
}

fieldset,
img {
    border: 0;
}

address,
caption,
cite,
code,
dfn,
em,
strong,
th,
```

```
var {
    font-style: normal;
    font-weight: normal;
}

ol,
ul {
    list-style: none;
}

caption,
th {
    text-align: left;
}

h1,
h2,
h3,
h4,
h5,
h6 {
    font-size: 100%;
    font-weight: normal;
}

q:before,
q:after {
    content: '';
}

abbr,
acronym {
    border: 0;
    font-variant: normal;
}

sup {
    vertical-align: text-top;
}

sub {
    vertical-align: text-bottom;
}

input,
textarea,
select {
    font-family: inherit;
    font-size: inherit;
    font-weight: inherit;
    *font-size: 100%;
}
```

```
legend {
    color: #000;
}
```