



# CSS基础

# ■ CSS概述

## HTML 的局限性

HTML, **只关注内容的语义**。比如 `<h1>` 表明这是一个大标题, `<p>` 表明这是一个段落, `<img>` 表明图片, `<a>` 表示链接。

**CSS:** Cascading Style Sheets 层叠样式表---**内容和样式相分离, 便于修改样式**

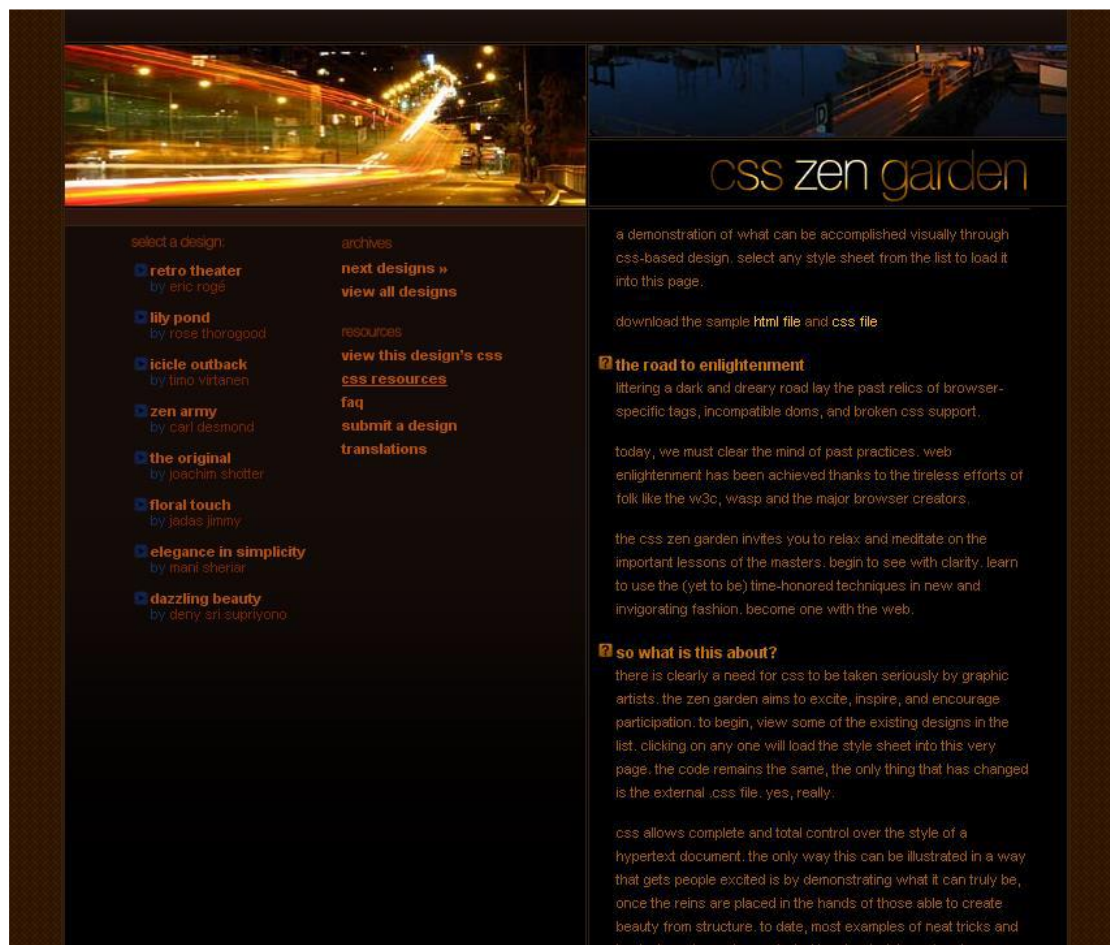
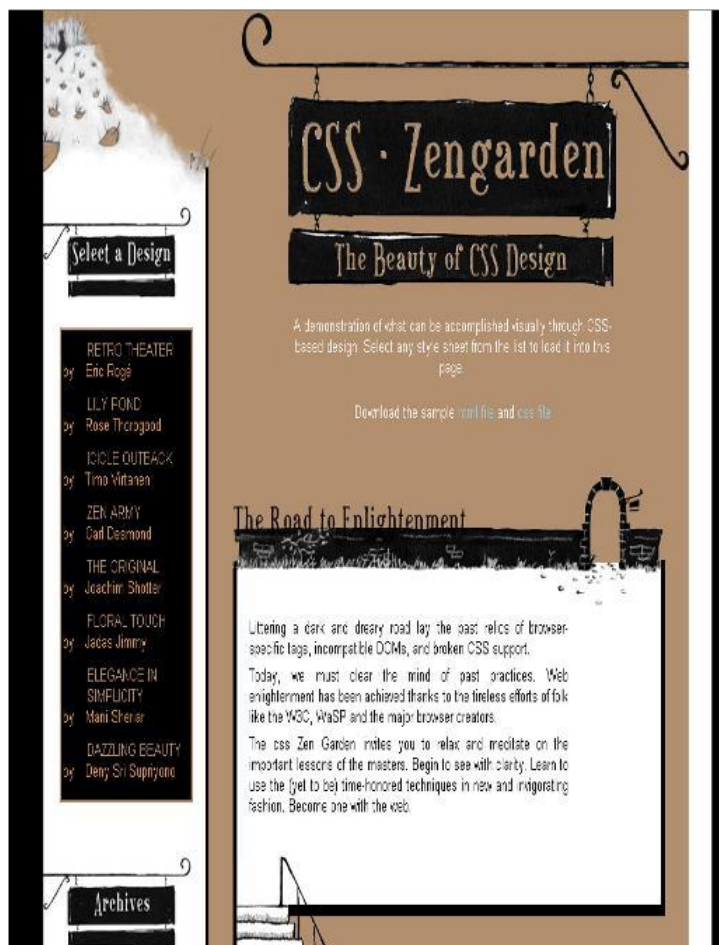
CSS 主要用于设置 HTML 页面中的**文本内容** (字体、大小、对齐方式等)、**图片的外形** (宽高、边框样式、边距等) 以及**版面的布局和外观显示样式**。

CSS 可以让网页更加丰富多彩, 布局更加灵活自如。

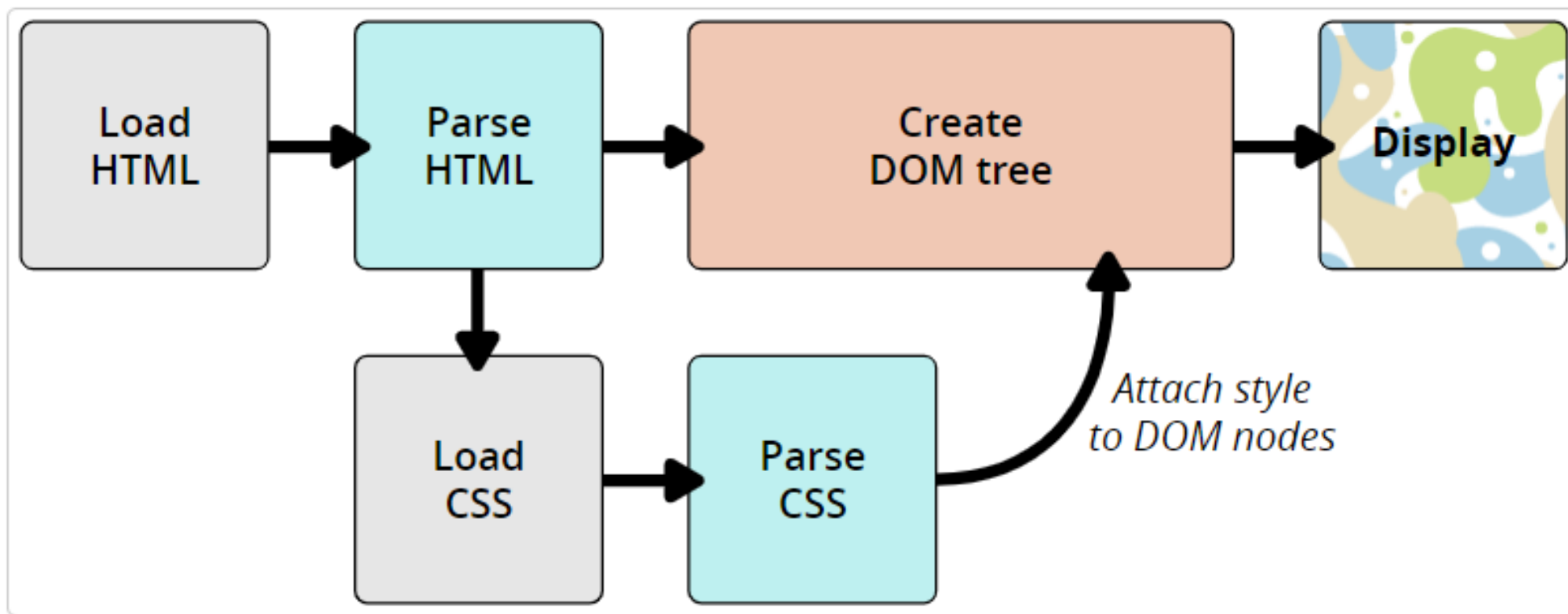
简单理解: CSS 美化 HTML, 让 HTML 网页更美观。

# ■ CSS概述

相同的HTML，不同的CSS，不同的显示效果。



## ■ CSS工作原理



# ■ CSS工作原理

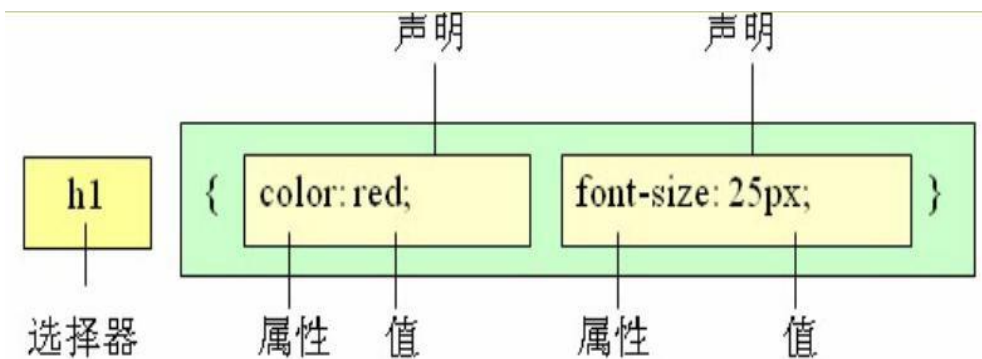
1. 浏览器载入 HTML 文件。
2. 将 HTML 文件转化成一个 DOM (Document Object Model文档对象模型) , DOM 是 HTML文件在计算机内存中的表现形式。
3. 接下来, 浏览器会拉取和该 HTML 相关的大部分资源, 比如嵌入到页面的图片、视频以及 CSS 样式, JavaScript 则会稍后进行处理。
4. 浏览器拉取到 CSS 样式之后会进行解析, 根据选择器的不同类型 (比如 element、class、id 等等) 把他们分到不同的“桶”中。浏览器基于它找到的不同的选择器, 将不同的规则 (基于选择器的规则, 如元素选择器、类选择器、id 选择器等) 应用在对应的 DOM 节点中, 并添加节点依赖的样式 (这个中间步骤称为渲染树) 。
5. 上述的规则应用于渲染树之后, 渲染树会依照应该出现的结构进行布局。
6. 网页展示在屏幕上 (这一步被称为着色) 。

# ■ CSS语法

## CSS语法规范

使用 HTML 时，需要遵从一定的规范，CSS 也是如此。要想熟练地使用 CSS 对网页进行修饰，首先需要了解 CSS 样式规则。

**CSS 规则由两个主要的部分构成：选择器以及一条或多条声明。**



- **选择器**是用于指定 CSS 样式的 **HTML 标签**，花括号内是对该对象设置的具体样式
- 属性和属性值以“键值对”的形式出现
- 属性是对指定的对象设置的样式属性，例如字体大小、文本颜色等
- 属性和属性值之间用英文 “:” 分开
- 多个“键值对”之间用英文 “;” 进行区分

# ■ CSS语法

## CSS语法规范

- 1、最后一条声明可以没有分号，但是为了以后修改方便，一般也加上分号。
- 2、为了使样式更加容易阅读，可以将每条代码写在一个新行内

## CSS 代码风格

样式格式书写

样式大小写风格

样式空格风格

# ■ CSS语法

## CSS 代码风格

### 1. 样式格式书写

#### ① 紧凑格式

```
h3 { color: deeppink;font-size: 20px;}
```

#### ② 展开格式

```
h3 {  
    color: pink;  
    font-size: 20px;  
}
```

**强烈推荐第二种格式**



# ■ CSS语法

## CSS 代码风格

### 2. 样式大小写

```
h3 {  
  color: pink;  
}
```

```
H3 {  
  COLOR: PINK;  
}
```

强烈推荐样式选择器，属性名，属性值关键字**全部使用小写字母**，特殊情况除外。

# ■ CSS语法

## CSS 代码风格

### 3. 空格规范

```
h3 {  
    color: pink;  
}
```

- ① 属性值前面，冒号后面，保留一个空格
- ② 选择器（标签）和大括号中间保留空格

# ■ CSS语法

## CSS 添加方法—内联样式表（行内样式）

```
<!DOCTYPE HTML>
<html>
  <head>
  </head>
  <body>
    <p style="color: red">Example Content
    </p>
  </body>
</html>
```



Example Content

# ■ CSS语法

## CSS 添加方法—内部样式表

```
<!DOCTYPE HTML>
<html>
  <head>
    <style type="text/css">
      p{
        color:red; /*设置字体颜色*/
      }
    </style>
  </head>
  <body>
    <p>
      Example Content
    </p>
  </body>
</html>
```



Example Content

- 即使有公共 CSS 代码，也是每个页面都要定义的
- 适合文件很少，CSS 代码也不多的情况
- 如果一个网站有很多页面，每个文件都会变大，后期维护难度也大

# ■ CSS语法

## CSS 添加方法—外部样式表

外部样式表文件 **style.css**

```
h1 {  
    color: orange;  
    font-size: 24px;  
}  
p {  
    color: gray;  
    font-size: 14px;  
    line-height: 1.5em;  
}
```



css



1.html



2.html

网页文件 **1.html**

```
<!DOCTYPE html>  
<html>  
<head>  
    <meta charset="UTF-8">  
    <title>Blog </title>  
    <link rel="stylesheet"  
        href="styles/style.css"/>  
</head>  
<body>  
    <h1>前端三大件: HTML、CSS、JavaScript</h1>  
    <p>CSS 美化 HTML ,布局网页</p>  
</body>  
</html>
```

# ■ CSS语法

## CSS 添加方法—外部样式表

外部样式表文件style.css

```
h1 {  
  color: orange;  
  font-size: 24px;  
}  
p {  
  color: gray;  
  font-size: 14px;  
  line-height: 1.5em;  
}
```



css



1.html



2.html

网页文件2.html

```
<!DOCTYPE html>  
<html>  
<head>  
  <meta charset="UTF-8">  
  <title>Blog </title>  
  <link rel="stylesheet"  
    href="styles/style.css"/>  
</head>  
<body>  
  <h1>Web前端学习</h1>  
  <p>HTML、CSS 、 JavaScript </p>  
</body>  
</html>
```



# ■ CSS语法

## CSS 添加方法—外部样式表

- ✓ 页面结构HTML 代码与样式 CSS 代码的完全分离
- ✓ 维护方便
- ✓ 如果需要改变网站风格，只需要修改公共CSS 文件
- ✓ 可以在同一个HTML 文档内部引用多个外部样式表

# ■ CSS语法

## CSS 添加方法—优先级

多重样式可以层叠，可以覆盖

- 样式的优先级按照“就近原则”
- 内联样式> 内部样式> 外部样式> 浏览器默认样式

外部样式表文件

```
h3 {  
    color: red;  
    text-align: left;  
    font-size: 8pt;  
}
```

内部样式表

```
h3 {  
    text-align: right;  
    font-size: 20pt;  
}
```

h3得到的样式是：

```
color:red;  
text-align:right;  
font-size:20pt;
```



# ■ CSS基础选择器

## CSS基础选择器类型：

- ✓ 元素选择器
- ✓ 类选择器
- ✓ id选择器

## ■ CSS基础选择器—元素选择器

```
<h1>This is heading </h1>  
<p>this is some paragraph. </p>
```

```
<style>  
h1 {  
  color: orange;  
}  
p {  
  color: gray;  
  font-size: 20px;  
}  
</style>
```

**This is heading**

this is some paragraph.

## ■ CSS基础选择器—类选择器

```
<h2>Todo List </h2>
<ul>
  <li class="done">Learn HTML </li>
  <li class="done">Learn CSS </li>
  <li>Learn JavaScript </li>
</ul>
```

```
<style>
.done {
  color: gray;
  text-decoration: line-through;
}
</style>
```

### Todo List

~~Learn HTML~~  
~~Learn CSS~~  
Learn JavaScript

## ■ CSS基础选择器—类选择器

### CSS

```
<style>
.big{
    font-size: 20px;
}
</style>
```

### HTML

```
<body>
    <p class="big">HTML</p>
    <p>超文本标记语言 </p>
    <p class="big">CSS</p>
    <p>层叠样式表 </p>
</body>
```

HTML

超文本标记语言

CSS

层叠样式表

## ■ CSS基础选择器—多类选择器

```
<div class="red font18">VSCode开发工具</div>
```

- (1)在标签class 属性中写 多个类名;
- (2)多个类名中间必须用空格分开;
- (3)这个标签就可以分别具有这些类名的样式;

### 多类名开发中使用场景

- (1)可以把一些标签元素相同的样式(共同的部分)放到一个类里面.
- (2)这些元素都可以调用这个公共的类, 然后再调用自己独有的类.
- (3)这样节省CSS代码, 统一修改也方便.

## ■ CSS基础选择器—多类选择器

```
.big{  
    font-size: 18px;  
}  
.red{  
    color: red;  
}  
.bg {  
    background-color: #ccc;  
}
```

```
<p class="big bg">HTML</p>  
<p class="red bg">CSS</p>
```

HTML

CSS

多类名,表示一个标签有多个类名, 相同样式放在一起, 自己独有的样式放在一个类里面

## ■ CSS基础选择器—id选择器，具有唯一性

```
<header>
  <h1 id="logo">
    
```

HTML5 文档

```
<h1>
</header>
```

```
<style>
  #logo {
    font-size: 60px;
    font-weight: 700;
  }
</style>
```



# HTML5 文档

HTML 元素以 **id 属性** 来设置 id 选择器，CSS 中 id 选择器以“**#**”来定义。

## ■ CSS基础选择器——id选择器 & 类选择器

### id 选择器和类选择器的区别

- ✓ id 不得重复。
- ✓ id 选择器和类选择器最大的不同在于使用次数上。
- ✓ 类选择器在修改样式中用的最多，id 选择器一般用于页面唯一性的元素上。



## ■ CSS基础选择器—通配选择器 \*

```
<h1>This is heading </h1>  
<p>this is some paragraph. </p>
```

```
<style>  
* {  
  color: red;  
  font-size: 20px;  
}  
</style>
```

**This is heading**

this is some paragraph.

# ■ CSS基础选择器—总结

元素选择器：<标签>--→ 标签名

类选择器：<标签 class="类名">--→ .类名

ID选择器：<标签 id="id名">--→ #id名

基础选择器	作用	特点	使用情况	用法
元素选择器	可以选出所有相同的标签元素， 比如p	不能差异化选择	较多	p{color:black;}
类选择器	可以选出一个或多个元素	可根据需求选择	非常多	.big{font-size:20px;}
ID选择器	一次只能选出一个元素	id属性只能在每个HTML 文档中出现一次	一般在js中可以通过id属 性获取元素节点	#small{ font-size:14px; }
通配选择器	选择所有的元素	一次选择所有	特殊情况使用	*{margin:0;padding:0;}

- 每个基础选择器都有使用场景，都需要掌握
- 如果是修改样式，类选择器是使用最多的

# ■ CSS复合选择器

## 什么是复合选择器？

在 CSS 中，可以根据选择器的类型把选择器分为**基础选择器**和**复合选择器**，复合选择器是建立在基础选择器之上，对 基本选择器进行组合形成的。

- 复合选择器可以更准确、更高效的选择目标元素（标签）
- 复合选择器是由两个或多个基础选择器，通过不同的方式组合而成的
- 常用的复合选择器包括：后代选择器、子选择器、并集选择器、伪类选择器等等

## ■ CSS复合选择器--后代选择器

**后代选择器**又称为**包含选择器**，可以选择父元素里面的子元素。把外层标签写在前面，内层标签写在后面，中间用空格分隔。当标签发生嵌套时，内层标签就称为外层标签的后代。

语法：

元素1 元素2 { 样式声明 }

上述语法表示选择元素 1 里面的**所有**元素 2 (后代元素)。 例如：

ul li { 样式声明 } /\*选择 ul 里面所有的 li 标签元素\*/

- 元素1 和 元素2 中间用**空格隔开**
- 元素1 是父级，元素2 是子级，最终选择的是**元素2**
- 元素2 可以是儿子，也可以是孙子等，只要是元素1 的后代即可
- 元素1 和 元素2 可以是任意基础选择器

## ■ CSS复合选择器--子选择器

子元素选择器（子选择器）只能选择作为某元素的最近一级子元素。

语法：

元素1 > 元素2 { 样式声明 }

上述语法表示选择元素1 里面的所有直接后代(子元素) 元素2。 例如：

```
div > p { 样式声明 }    /*选择 div 里面所有最近一级 p 标签元素    */
```

- 元素1 和 元素2 中间用 **大于号** 隔开
- 元素1 是父级，元素2 是子级，**最终选择的是元素2**
- 元素2 最近一级。

## ■ CSS复合选择器--子选择器

1. 请将下面的[链接文字](#)修改为红色。

```
<div class="nav">
  <ul>
    <li><a href="#">百度</a></li>
    <li><a href="#">谷歌</a></li>
  </ul>
</div>
```

## ■ CSS复合选择器--子选择器

2. 请将下面的水果文字修改为红色。

```
<div class="hot">
  <a href="#">水果</a>
  <ul>
    <li><a href="#">香蕉</a></li>
    <li><a href="#">橙子</a></li>
  </ul>
</div>
```

## ■ CSS复合选择器—并集选择器

并集选择器可以选择多组标签,同时为他们定义相同的样式。通常用于集体声明。

**并集选择器**是各选择器通过**英文逗号** (,) 连接而成,任何形式的选择器都可以作为并集选择器的一部分。 语法:

元素1,元素2 { 样式声明 }

上述语法表示选择元素1 和 元素2。 例如:

ul,div { 样式声明 } /\* 选择 ul 和div标签元素 \*/

- 元素1 和 元素2 中间用**逗号**隔开
- 逗号可以理解为**和**的意思
- 并集选择器通常用于集体声明



## ■ CSS复合选择器—相邻兄弟选择器

用+号连接两个兄弟元素，表示给某元素其后紧邻的兄弟元素设置样式。

用这段CSS代码可以把h2标题后的第一个段落文字设置为粗体

```
h2+p{
```

```
    font-weight: 700;
```

```
}
```

**h2标题**

段落文字1

段落文字2

**h2标题**

段落文字3

段落文字4

## ■ CSS选择器—属性选择器

属性选择器可以根据元素的属性及属性值来选择元素。用于筛选出属性符合条件的元素。

属性选择器	说明
E[att]	选择具有att属性的E元素
E[att="val"]	选择具有att属性，且其值等于val的E元素
E[att^="val"]	匹配具有att属性，且值以val开头的E元素
E[att\$="val"]	匹配具有att属性，且值以val结尾的E元素
E[att*="val"]	匹配具有att属性，且值中含有val的E元素

例如可以筛选出属性src取值包含“bg”的img元素，设置宽、高和边框，代码如下：

```
img[src*="bg"]{  
    width:180px;  
    height:90px;  
    border:1px solid #e4e4e4;  
}
```

## ■ CSS选择器—伪类选择器

**CSS 伪类**是添加到选择器的关键字，用于指定所选元素的特殊状态。比如给链接添加特殊效果，或者选择第1个，第n个元素。例如，伪类 **:hover** 可以用于选择一个按钮，当用户的指针悬停在按钮上时，设置此按钮的样式。

```
/* 用户的指针悬停在任何按钮 */  
button:hover {  
    color: blue;  
}
```

伪类选择器书写最大的特点是**用冒号 (:) 表示**，比如 **:hover**、**:first-child**。附上了伪类的元素被定义为锚元素，比如button:hover中的button即为锚元素。

伪类选择器很多，比如有链接伪类、结构伪类、函数式伪类等。

# ■ CSS选择器—链接伪类选择器

## **:link**

设置a对象在未被访问前的样式表属性。

```
a:link {  
    font-size: 14pt;  
    text-decoration: underline;  
    color: blue;  
}
```

## **:visited**

设置a对象在其链接地址已被访问过时的样式表属性。

```
a:visited {  
    font-size: 14pt;  
    text-decoration: underline;  
    color: blue; }
```

## **:hover**

设置对象在其鼠标悬停时的样式表属性。

```
a:hover {  
    font-size: 14pt;  
    text-decoration: underline;  
    color: blue;  
}  
a:hover span{  
    color:red;  
}
```

## **:active**

设置对象在被用户激活（在鼠标点击与释放之间发生的事件）时的样式表属性。

```
a:active {  
    font-size: 14pt;  
    text-decoration: underline;  
    color: blue;  
}
```

## ■ CSS选择器—链接伪类选择器

### 链接伪类选择器注意事项：

- 1.为了确保生效，需要按照 **LVHA** 的顺序声明 :link - :visited - :hover - :active。
- 2.因为 a 链接在浏览器中具有默认样式，所以实际工作中都需要给链接单独指定样式。

### 链接伪类选择器实际工作开发中的写法：

```
/* a 是标签选择器，所有的链接文字为灰色 */
a {
    color: gray;
}
/* :hover 是链接伪类选择器鼠标经过*/
a:hover {
    color: red;    /* 鼠标经过的时候，由原来的灰色变成了红色 */
}
```

## ■ CSS选择器—:focus伪类选择器

**:focus 伪类选择器**用于选取获得焦点的表单元素。

焦点就是光标，一般情况 `<input>` 类表单元素才能获取，因此这个选择器也主要针对于表单元素来说的。

```
<style>
    /* // 把获得光标的input表单元素选取出来 */
    input:focus {
        background-color: lightgreen;
        color: blue;
    }
</style>
<body>
    <input type="text">
    <input type="text">
    <input type="text">
</body>
```

# ■ CSS选择器--伪类选择器

## 1、:first-child和:last-child

:first-child选择一组兄弟元素中的第一个元素

:last-child选择一组兄弟元素中的最后一个元素

```
<p class="content">asdfga</p>
<p class="content">asdfg</p>
<p class="content">asdfg</p>
<p class="content">asdfg</p>
<p class="content">asdfg</p>
```

```
.content:first-child {
    background-color: pink;
}
.content:last-child {
    background-color: green;
}
```

asdfga

asdfg

asdfg

asdfg

asdfg

# ■ CSS选择器--伪类选择器

## 2、:first-of-type和:last-of-type

:first-of-type选择一组兄弟元素中指定类型的第一个元素。

:last-of-type选择一组兄弟元素中指定类型的最后一个元素。

```
<div class="content">1</div>
<p class="content">asdfga</p>
<p class="content">asdfg</p>
<p class="content">asdfg</p>
<div class="content">2</div>
<div class="content">3</div>
```

```
.content:first-child {
    background-color: pink;
}
.content:last-child {
    background-color: green;
}
p:first-of-type {
    color: brown;
}
.content:last-of-type {
    font-size: 40px;
}
```

1

asdfga

asdfg

asdfg

2

3

区别在于：

:first-child匹配该元素的所有兄弟元素的第一个，不管该元素是什么类型，再判断类型是不是可以匹配

:first-of-type先按类型匹配，选择指定类型中的第一个元素



# ■ CSS选择器--伪类选择器

## 3、:nth-child和:nth-last-child

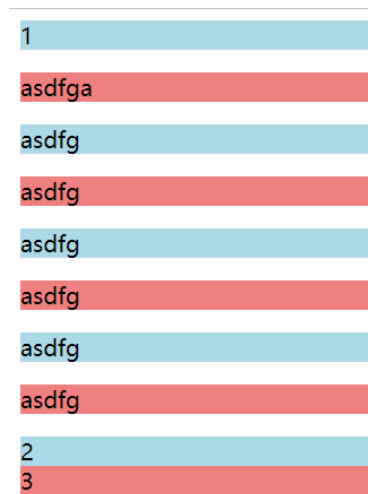
这两个选择器用于按照元素排序选择某些兄弟元素。

:nth-child按照从前向后的顺序匹配，:nth-last-child相反。以:nth-child为例，其参数取值分为以下几种情况：

- (1) :nth-child(an+b)首先会找到当前元素的兄弟元素，然后按照位置的先后顺序从1开始排序，选择的结果为第an+b个元素的集合。（n=0,1,2,3...）
- (2) :nth-child(-n+b)选择前b+1个元素
- (3) :nth-child(odd|even)选择当前元素的所有奇数个或偶数个兄弟元素，相当于参数取值为2n+1和2n

例如.content:nth-child(3) .content:nth-child(odd) .content:nth-child(even)

```
.content:nth-child(3) {  
    background-color: yellow;  
}  
.content:nth-child(odd) {  
    background-color: lightblue;  
}  
.content:nth-child(even) {  
    background-color: lightcoral;  
}
```



## ■ CSS选择器--伪类选择器

### 3、:nth-child和:nth-last-child

选择器示例:

tr:nth-child(2n+1): 表示 HTML 表格中的奇数行。等同于tr:nth-child(odd)

span:nth-child(0n+1): 表示子元素中第一个且为 span 的元素, 与 :first-child 选择器作用相同

span:nth-child(-n+3): 匹配前三个span 元素。

tr:nth-last-child(odd) or tr:nth-last-child(2n+1): 表示 HTML 表的倒数的奇数行:1、3、5 等。

:nth-last-child(-n+3): 表示一组兄弟节点中的最后三个元素。

# ■ CSS选择器--伪元素

## 1、::before和::after



伪元素创建了不存在DOM树中的元素，并为其添加样式。伪元素的前缀在旧标准中是一个冒号，新标准中为了和伪类区分开，前缀是两个冒号。

::before和::after会在真正页面元素之前和之后插入一个额外的元素。通过content属性为一个元素添加修饰性的内容，此元素默认为行内元素，格式如下：

```
元素::before|after{  
    content: "待插入内容";  
    其余样式设定  
}
```

例如应用::before向列表项开头处添加图标，应用::after向第一项结尾处添加文字表示热度。

```
<div class="item">  
  <ul>  
    <li>正确答案</li>  
    <li>错误答案</li>  
    <li>热度词</li>  
  </ul>  
</div>
```

-  正确答案
-  错误答案
- 热度词 **hot!**

```
.item ul li:first-child::before {  
    content: url(images/success.png);  
}  
.item ul li:nth-child(2)::before {  
    content: url(images/error.png);  
}  
.item ul li:last-child::after {  
    content: 'hot!';  
    color: red;  
    font-size: 10px;  
    margin-left: 10px;  
}
```

# ■ CSS选择器--伪元素

## 2、::first-letter和::first-line

::first-letter选择元素的第一个字母， ::first-line选择元素的第一行。

例如， 可以使用::first-letter和::first-line设置每段第一个字母以红色两倍字号显示， 第一行粗体显示。

```
p::first-letter {  
    font-size: 2em;  
    color: red;  
}  
p::first-line {  
    font-weight: 700;  
}
```

**H**ow can one person bask in the sunshine of good health, while another person looks old before her time?

Humans have been asking this question for millennia, and recently, it's becoming clearer and clearer to scientists that the differences between people's rates of aging lie in the complex interactions among genes, social relationships, environments and lifestyles.

**E**ven though you are born with a particular set of genes, the way you live can influence how they express themselves. Some lifestyle factors may even turn genes on or shut them off.

## ■ CSS选择器—小结

选择器	作用	特征	使用情况	隔开符号及用法
后代	选择后代元素	可以是子孙后代	较多	空格, .nav p
子代	选择最近一级元素	只选亲儿子	较少	大于>, .nav>p
并集	选择某些相同样式的元素	可以用于集体声明	较多	逗号, .nav, .header
链接伪类	选择不同状态的链接	跟链接相关	较多	重点是a: hover
伪元素	创建不存在DOM树中的元素	可用于创建元素	较少	::before, ::after
:focus	选择获得光标的表单	跟表单相关	较少	input: focus

## ■ CSS层叠与继承

样式表层叠——简单的说，就是 CSS 规则的顺序很重要；当两条同级别的规则应用到一个元素的时候，写在后面的就是实际使用的规则。

```
<h1>This is h1 title.</h1>
```

```
-----
```

```
h1 {  
  color: red;  
}  
h1 {  
  color: blue;  
}
```

**This is h1 title.**

两个CSS规则具有相同的元素选择器，有相同的优先级，所以顺序在最后的生效。

## ■ CSS层叠与继承

CSS样式可以由父元素传递给子元素，而且多个样式可以叠加作用于一个元素，将所有继承和叠加关系进行计算并得到最终样式就是层叠的过程。层叠过程需要同时考虑继承关系和样式优先级（Specificity）

父元素的有些样式默认由后代元素继承。

### 继承属性：

- 1、文字属性：font、font-family、font-size、font-style、font-weight
- 2、文本属性：color、letter-spacing、line-height、text-decoration
- 3、列表属性：list-style、list-style-type、list-style-position、list-style-image

### 非继承属性：

与盒子模型相关的属性不会被继承，主要包括：margin、padding、border、display、background、width、height、float、clear、position、left、right、top、bottom、z-index

# ■ CSS层叠与继承

## 样式优先级

由样式添加顺序、添加方式、选择器类型决定

### 1、添加顺序

有冲突时，先添加的样式会被后添加的样式覆盖。

### 2、添加方式

如果没有定义CSS样式，浏览器有默认样式。多重样式作用于同一元素可以层叠，可以覆盖。样式添加方式不同，优先级不同，样式的优先级按照“就近原则”。

**内联样式 > 内部样式 > 外部样式 > 默认样式**

### 3、选择器类型

对于行内样式的设置，优先级顺序如下：

**style属性 > ID选择器 > 类选择器**



# ■ CSS层叠与继承

## 样式优先级

内联样式，即 style 属性内的样式声明，优先于所有普通的样式，无论其优先级如何。

## !important

是一个特殊的 CSS，可以用来覆盖所有优先级计算，不过需要很小心的使用。主要用于修改特定属性的值，能够覆盖普通规则的层叠。

```
#winning {  
  background-color: red;  
  border: 1px solid black;  
}
```

```
<p class="better">This is a paragraph.</p>  
<p class="better" id="winning">One selector to rule them all!</p>
```

```
.better {  
  background-color: gray;  
  border: none !important;  
}
```

This is a paragraph.

One selector to rule them all!

```
p {  
  background-color: blue;  
  color: white;  
  padding: 5px;  
}
```

## ■ CSS层叠与继承

```
<p id="blue" class="red" style="color:lightgreen">Paragraph</p>
```

```
#blue{
```

```
    color:blue;
```

```
    font-size:16px;
```

```
}
```

```
.red{
```

```
    color:red;
```

```
    font-size:18px;
```

```
}
```

Paragraph文本的字号多少， 颜色多少？

## ■ Emmet 语法

Emmet是一个Web开发工具，用于加快HTML和CSS代码的编写速度。使用Emmet能够通过简短的表达式生成HTML或CSS代码片段。另外，截至2022年，主流的编辑器工具如Visual Studio Code、WebStorm都已经内置了Emmet工具，无需手动安装即可使用。

在后缀为 .html 或 .css 文件中输入 emmet 缩写语句，然后按 tab 键即会自动生成相应代码。

关于emmet，更多请参考Visual Studio Code官方：<https://code.visualstudio.com/docs/editor/emmet>

# ■ Emmet 语法

## 快速生成HTML结构语法

1. 生成标签 直接输入标签名 按tab键即可 比如 div, 然后tab 键, 就可以生成 `<div></div>`
2. 如果想要生成多个相同标签 加上 \* 就可以了 比如 `div*3` 就可以快速生成3个div
3. 如果有父子级关系的标签, 可以用 > 比如 `ul > li`就可以了
4. 如果有兄弟关系的标签, 用 + 就可以了 比如 `div+p`
5. 如果生成带有类名或者id名字的, 直接写 `.demo` 或者 `#two` tab 键就可以了
6. 如果生成的div 类名是有顺序的, 可以用 自增符号 \$
7. 如果想要在生成的标签内部写内容可以用 {} 表示, 比如`a{click me}`,按 tab 键即可生成 `<a href="">click me</a>`

# ■ Emmet 语法

## 快速生成CSS样式语法

CSS 基本采取简写形式即可.

1. 比如 w200 按tab 可以生成 width: 200px;
2. 比如 lh26px 按tab 可以生成 line-height: 26px;
3. 比如fwb 按tab 可以生成 font-weight: bold;
4. 比如poa 按tab 可以生成 position: absolute;
5. 比如tac 按tab 可以生成 text-align: center;

根据上面的例子，可以发现规律，Emmet中用首字母+具体值的形式生成CSS代码片段

# ■ Emmet 语法

## 快速格式化代码

Vscode 快速格式化代码: shift+alt+f

**也可以设置保存页面的时候自动格式化代码:**

- 1) 文件 -----> 【首选项】 -----> 【设置】 ;
- 2 在settings.json下的【工作区设置】中添加以下语句:

```
"editor.formatOnSave": true,  
"editor.formatOnType": true,
```

# ■ 网站结构

一个网站包含许多文件：文本内容、代码、样式表、媒体内容，等等。当建立一个网站时，我们需要在计算机上将这些文件以合理的结构组织起来，确保文件之间交互畅通，并在最终将它们上传到服务器之前使所有内容看起来正确。

- web-projects：存储网站项目。
- 在web-projects 文件夹下新建一个名字类似为test-site的文件夹存放网站具体内容。

## ■ 文件命名注意事项

要求统一用小写字母命名文件和文件夹，不允许有空格，文件名或者文件夹名中可以使用连字符。例如test-file.html。使用连字符而不要用下划线。

因为：

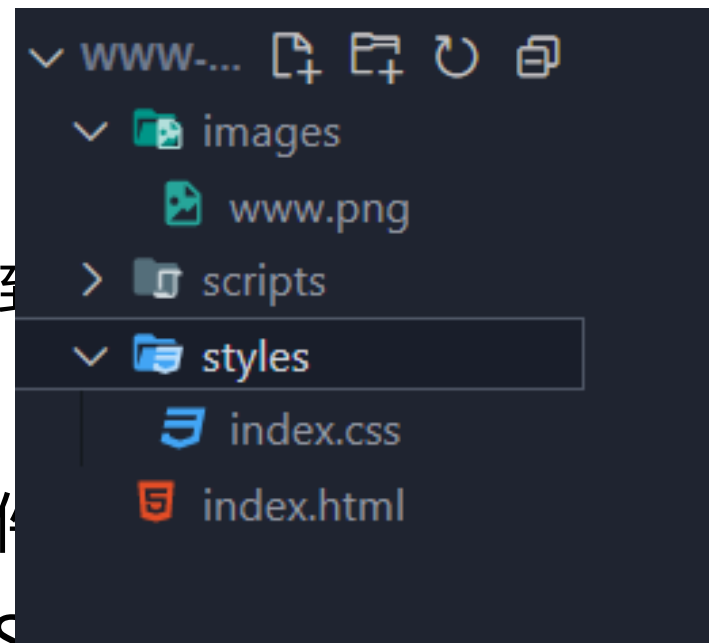
1. 许多计算机，特别是网络服务器，是区分大小写的。
2. 浏览器、网络服务器和编程语言对空格的处理并不一致。一些系统可能将其视为两个文件名。一些服务器会用“%20”（URL 中空格的字符代码）替换文件名中的区域，导致所有链接被破坏。



# ■ 网站结构

网站文件夹test-site

- index.html: 包含主页内容, 是第一次访问网站时看到
- images: 存放网站上使用的所有图片
- styles: 存放用于设置内容样式的 CSS 代码文件
- scripts: 存放用于向网站添加交互功能的JavaScript代码文件



## ■ web font

在网页中不仅可以使⽤本地已安装的字⽴，为了使网页美观，还可以向网页中添加web字⽴。

Google Fonts 使⽤⽴式，两步：

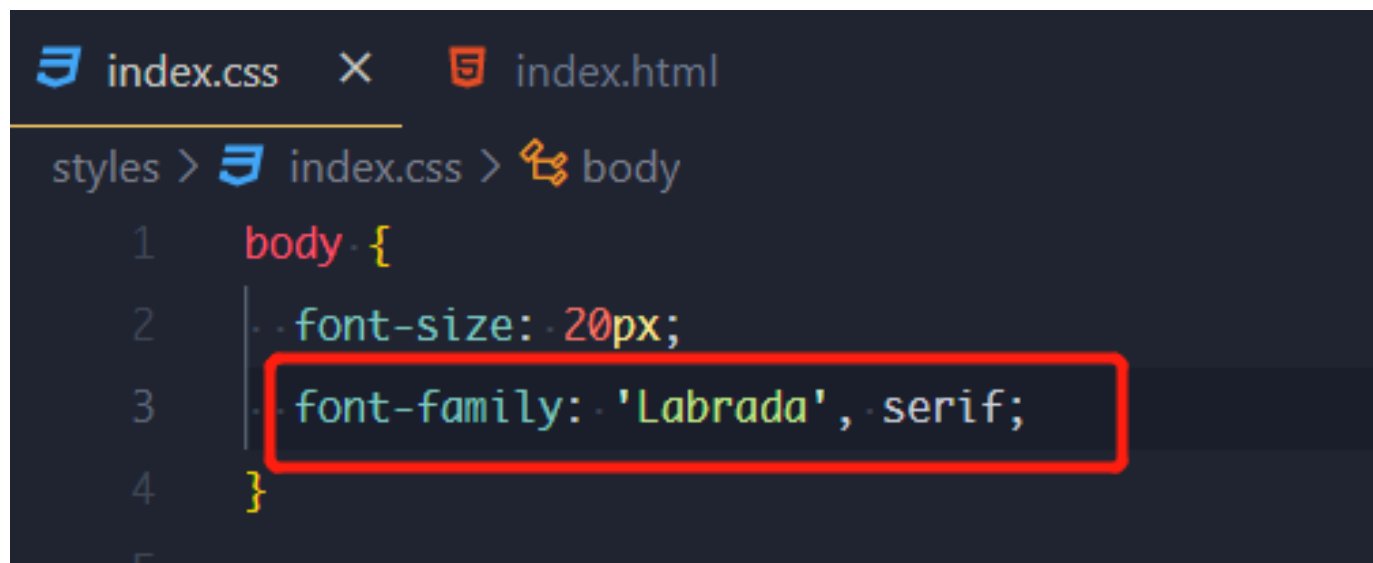
```
<head>
  <meta charset="UTF-8">
  <meta http-equiv="X-UA-Compatible" content="IE=edge">
  ...
  <meta name="viewport" content="width=device-width, initial-scale=1.0">
  <link rel="stylesheet" href="styles/index.css">
  <link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Labrada">
  <title>Website</title>
</head>
```

1、添加样式表链接以请求所需的网页字⽴：

```
<link rel="stylesheet" href="https://fonts.googleapis.com/css?family=Labrada">
```

## ■ web font

2、在样式表中使用请求的 Web 字体设置元素的样式。

A screenshot of a code editor with a dark theme. At the top, there are two tabs: 'index.css' (active) and 'index.html'. Below the tabs, the breadcrumb 'styles > index.css > body' is visible. The code editor shows the following CSS code:

```
1  body {  
2    font-size: 20px;  
3    font-family: 'Labrada', serif;  
4  }
```

The line 'font-family: 'Labrada', serif;' is highlighted with a red rectangular box.

注意：始终至少列出一个后备网络安全字体，避免意外行为。

## ■ web font

如果要请求多个字体系列，可以使用竖线字符 (|) 分隔名称，如果字体名称含有空格，将空格替换为加号+。

```
<link rel="stylesheet"  
href="https://fonts.googleapis.com/css?family=Labrada|Inconsolata|Droid+Sans">
```

不要添加过多字体，一般网页无需太多字体，而且过多网络字体也会使页面加载速度变慢。

## ■ web font

默认地，Google Fonts API 提供的是字体的常规版本。需要其他样式或粗细，需要在字体名称后面附加一个冒号 (:)，后跟一个样式或粗细的列表，以英文逗号分隔 (,)。

```
<link rel="stylesheet"
href="https://fonts.googleapis.com/css?family=Labrada:italic,bold|Inconsolata|Droid+Sans:700">
```

样式	说明符
斜体	<code>italic</code> 或 <code>i</code>
粗体	<code>bold</code> 、 <code>b</code> 或数字权重 (例如 700)
粗体斜体	<code>bolditalic</code> 或 <code>bi</code>

## ■ web font

### 优化字体请求

假如仅是网页的标题文本需要使用网络字体，那么可以将标题文本设定为text参数的值，即

```
<link rel="stylesheet"  
href="https://fonts.googleapis.com/css?family=Labrada&text=Hello">
```

更多具体内容，请参考：[https://developers.google.com/fonts/docs/getting\\_started](https://developers.google.com/fonts/docs/getting_started)