CSS

CSS入门学习笔记+案例_css 入门案例-CSDN博客

网页制作基础学习——HTML+CSS常用代码_html网页制作代码大全-CSDN博客

【CSS3】CSS3 结构伪类选择器 (E:first-child / E:last-child 选择器 | E:nth-child(n) 选择器 | E:nth-of-type 选择器)_css伪类选择器-CSDN博客

CSS通常称为CSS样式表或层叠样式表（级联样式表），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）以及版面的布局等外观显示样式。CSS以HTML为基础，提供了丰富的功能，如字体、颜色、背景的控制及整体排版等，而且还可以针对不同的浏览器设置不同的样式.

1. css语法

```
<head>
    <style>
        选择器{
            属性名：属性值;
            属性名：属性值;
        }
    </style>
</head>
```

- 选择器：要修饰的对象（东西）
- 属性名：修饰对象的哪一个属性（样式）
- 属性值：样式的取值

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        h1{
            color: white;
            background: lightblue;
        }
    </style>
</head>
<body>
    <h1>这是1号标题</h1>
</body>
</html>
```


2.CSS应用方式

也称为CSS引用方式，有三种方式：内部样式、行内样式、外部样式

2.1 内部样式

也称为内嵌样式，在页面头部通过style标签定义,对当前页面中所有符合样式选择器的标签都起作用.（见上例）

2.2 行内样式

也称为嵌入样式，使用HTML标签的style属性定义,只对设置style属性的标签起作用.

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
</head>
<body>
    <h1 style="color: white;background: lightblue;">这是1号标题</h1>
</body>
</html>
```


2.3 外部样式

使用单独的 .CSS 文件定义，然后在页面中使用 link标签 或 @import指令 引入

- **使用 link标签 链接外部样式文件**

```
<link rel="stylesheet" type="text/css" href="CSS样式文件的路径">
```

提示：type属性可以省略

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <link rel="stylesheet" href="./test/style/hello.css">
</head>
<body>
    <h1>这是1号标题</h1>
</body>
</html>
```

style > # hello.css > h1

```css
h1{
    color: white;
    background: lightblue;
}
```


- @import 指令 导入外部样式文件

```
<style>
    @import "CSS样式文件路径";
    @import url(CSS样式文件路径);
</style>
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        @import url(./test/style/hello.css);
    </style>
</head>
<body>
    <h1>这是1号标题</h1>
</body>
</html>
```


3.选择器

3.1 基础选择器

3.1.1 标签选择器

也称为元素选择器，使用HTML标签作为选择器的名称，以标签名作为样式应用的依据

3.1.2 类选择器

使用自定义的名称，以 . 号作为前缀，然后再通过HTML标签的class属性调用类选择器

以标签的class属性作为样式应用的依据

注意事项：

调用时不能添加 . 号 同时调用多个类选择器时，以 空格 分隔 类选择器名称不能以 数字 开头

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        .hello{
            color: white;
            background: lightblue;
        }
    </style>
</head>
<body>
    <h1 class="hello">这是1号标题</h1>
</body>
</html>
```


3.1.3 ID选择器

使用自定义名称，以 # 作为前缀，然后通过HTML标签的id属性进行名称匹配

以标签的id属性作为样式应用的依据，一对一的关系

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        #hello{
            color: white;
            background: lightblue;
        }
    </style>
</head>
<body>
    <h1 id="hello">这是1号标题</h1>
</body>
</html>
```


3.2 复杂选择器

3.2.1 复合选择器

标签选择器和类选择器、标签选择器和ID选择器，一起使用

必须同时满足两个条件才能应用样式

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        /*标签选择器和类选择器合起来使用*/
        p.hello1{
            color: red;
            background: lightgreen;
        }
        /*标签选择器和ID选择器合起来使用*/
        h1#hello2{
            color: white;
            background: lightblue;
        }
    </style>
</head>
<body>
    <p>标签选择器和类选择器没有合起来使用</p>
    <p class="hello1">标签选择器和类选择器合起来使用</p>
    <h1>标签选择器和ID选择器没有合起来使用</h1>
    <h1 id="hello2">标签选择器和ID选择器合起来使用</h1>
</body>
</html>
```

标签选择器和类选择器没有合起来使用

标签选择器和类选择器合起来使用

标签选择器和ID选择器没有合起来使用

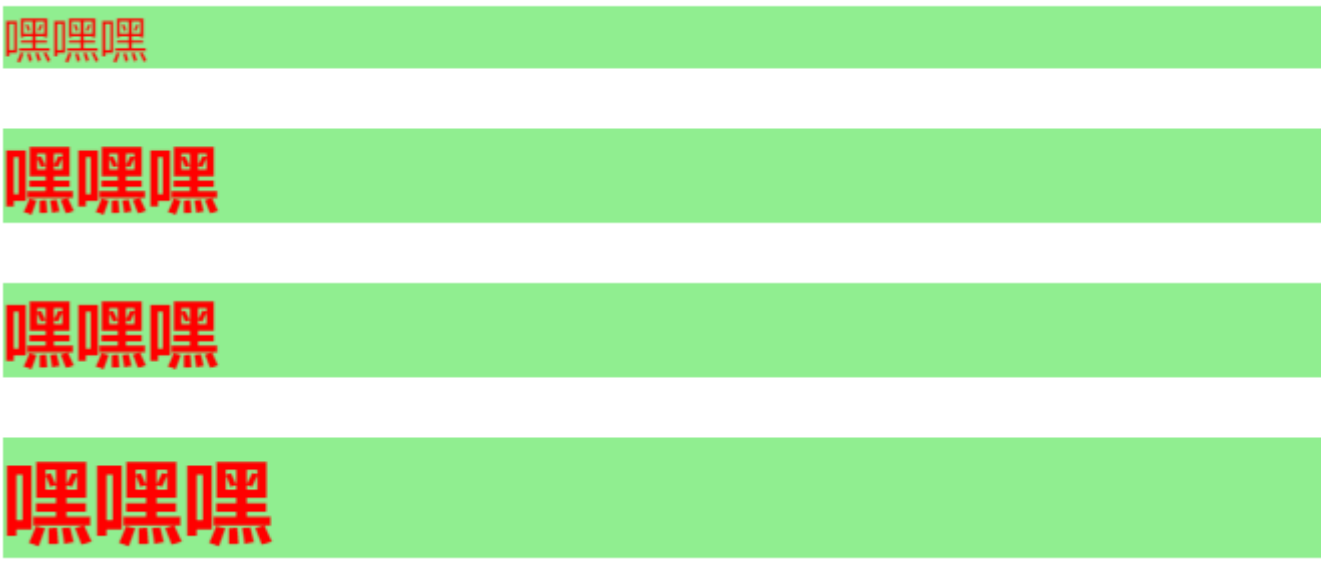
标签选择器和ID选择器合起来使用

3.2.2组合选择器

也称为集体声明

将多个具有相同样式的选择器放在一起声明，使用逗号隔开

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        /*组合选择器*/
        p,h1,.hello1,#hello2{
            color: red;
            background: lightgreen;
        }
        h1{
            font-size: 30px;
        }
    </style>
</head>
<body>
    <p>嘿嘿嘿</p>
    <h2 class="hello1">嘿嘿嘿</h2>
    <h2 id="hello2">嘿嘿嘿</h2>
    <h1>嘿嘿嘿</h1>
</body>
</html>
```



3.2.3 嵌套选择器

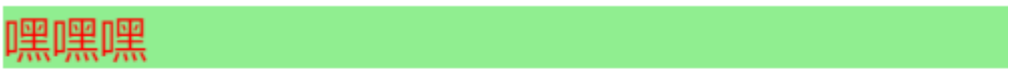
在某个选择器内部再设置选择器，通过空格隔开

只有满足层次关系最里层的选择器所对应的标签才会应用样式

注意：使用 空格 时不区分父子还是后代 **（后代选择器）**

使用CSS3中新增的 > 时必须是父子关系才行 **（子选择器）**

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        /*嵌套选择器*/
        div .hello1{
            color: red;
            background: lightgreen;
        }
        div>p{
            color: aqua;
            background: lightpink;
        }
    </style>
</head>
<body>
    <div><p class="hello1">嘿嘿嘿</p></div>
    <div><p>嘿嘿嘿</p></div>
    <div><h1><p>嘿嘿嘿</p></h1></div>
</body>
</html>
```



嘿嘿嘿

- **相邻兄弟选择器**

所谓相邻兄弟选择器，就是把当前元素的下一个元素找出来。

```
元素1 + 元素2 { 样式声明 }
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        .hello+h1{
            color: blue;
        }

    </style>
</head>
<body>
    <h1>嘿嘿</h1>
    <p class="hello">哈哈</p>
    <h1>嘿嘿哈</h1>
    <h1>嘿嘿哈哈</h1>
</body>
</html>
```

> # 嘿嘿
>
> 哈哈
>
> # 嘿嘿哈
>
> # 嘿嘿哈哈

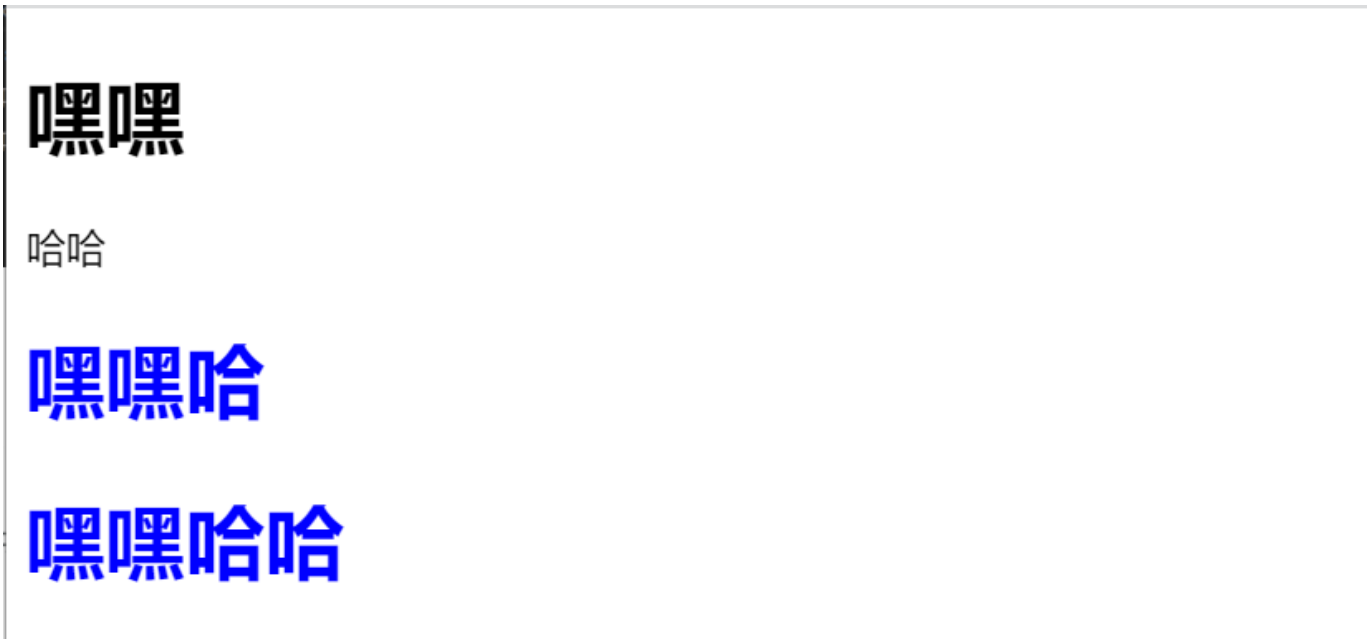
- **通用兄弟选择器**

所谓通用兄弟选择器，就是把当前元素的后面所有兄弟元素都找出来。

```
元素1 ~ 元素2 { 样式声明 }
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        .hello~h1{
            color: blue;
        }

    </style>
</head>
<body>
    <h1>嘿嘿</h1>
    <p class="hello">哈哈</p>
    <h1>嘿嘿哈</h1>
    <h1>嘿嘿哈哈</h1>
</body>
</html>
```



只会对元素1下面的兄弟元素2（所谓兄弟就是处于并列状态，之间不存在包含关系）进行添加样式。

3.3 伪类选择器

根据不同的状态显示不同的样式，一般多用于 标签

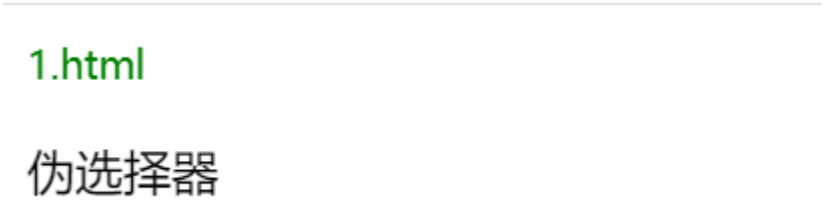
四种状态：

- :link 未访问的链接

- :visited 已访问的链接
- :hover 鼠标悬浮到连接上，即移动在连接上
- :active 选定的链接，被激活

注：默认超链接为：蓝色、下划线

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        a:link,a:visited{
            color: green;
            font-size: 13px;
            text-decoration: none;
        }
        a:hover,a:active{
            color: yellow;
            text-decoration: underline;
        }
        /*普通的标签也可以使用伪类选择器*/
        p:hover{
            color: red;
        }
        p:active{
            color: blue;
        }
    </style>
</head>
<body>
    <a href="1.html">1.html</a>
    <p>伪选择器</p>
</body>
</html>
```



3.4 伪元素选择器

- :first-letter 为第一个字符的样式

- :first-line 为第一行添加样式
- :before 在元素内容的最前面添加的内容，需要配合content属性使用
- :after 在元素内容的最后面添加的内容，需要配合content属性使用

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <meta name="viewport" content="width=device-width>, initial-scale=1.0">
    <title>Document</title>
    <style>
        p:first-letter{
            color:red;
            font-size:30px;
        }
        p:first-line{
            background:pink;
        }
        p:before{
            content:"嘿嘿";
        }
        p:after{
            content:"哈哈";
        }
    </style>
</head>
<body>
    <p>hello world!</p><hr>
    <p>hello world! <br>welcome to css!</p>
</body>
</html>
```

嘿嘿hello world!哈哈

嘿嘿hello world!
welcome to css!哈哈

3.5 选择器优先级

3.5.1优先级

行内样式>ID选择器>类选择器>标签选择器

原因：首先加载标签选择器,再加载类选择器，然后加载ID选择器，最后加载行内样式

后加载会覆盖先加载的同名样式

3.5.2 内外部样式加载顺

就近原则

原因：按照书写顺序依次加载，在同优先级的前提下，后加载的会覆盖先加载的同名样式，所以离的越近越优先

3.5.3 !important

可以使用!important使某个样式有最高的优先级

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        div{
            font-size:20px;
        }
        .hello{
            font-weight:bold;
            color:blue;
        }
        #world{
            text-decoration: underline;
            color:green;
        }
    </style>
</head>
<body>
    <div class="hello" id="world" style="color:#ff7300">hahahahah</div>
</body>
</html>
```

hahahahah

3.6 属性选择器

属性选择器可以根据元素特定属性的来选择元素。 这样就可以不用借助于类或者id选择器。

[ ]表示属性的意思 里面写具体属性的名称

3.6.1 E[attr]属性选择器

E[attr]属性选择器，选择匹配具有属性attr的E元素。

```
元素[属性] { 样式声明 }
```

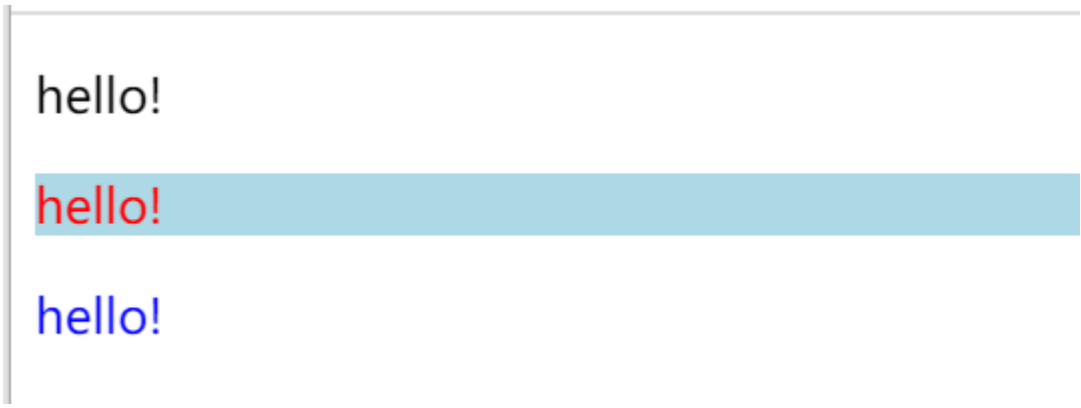
```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        #hello{
            color: red;
        }
        p[id]{
            background: lightblue;
        }
    </style>
</head>
<body>
    <h1 id="hello">hello!</h1>
    <p>hello!</p>
    <p id="hello">hello!</p>
</body>
</html>
```

3.6.2 E[attr=val]属性选择器

E[attr=val]属性选择器，选择匹配具有属性attr，且值为val的E元素，属性和属性值必须是完全匹配。

```
元素[属性=值] { 样式声明 }
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        #hello{
            color: red;
        }
        #world{
            color: blue;
        }
        p[id=hello]{
            background: lightblue;
        }
    </style>
</head>
<body>
    <p>hello!</p>
    <p id="hello">hello!</p>
    <p id="world">hello!</p>
</body>
</html>
```



3.6.3 E[attr*=val]属性选择器

E[attr^=val]:选择匹配元素E，且E元素定义了属性attr，其属性值是以val开头的任意字符串
E[attr$=val]:选择匹配元素E，且E元素定义了属性attr，其属性值是以val结尾的任意字符串
E[attr*=val]:选择匹配元素E，且E元素定义了属性attr，其属性值包含了val

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        #hello{
            color: red;
        }
        #world{
            color: blue;
        }
        p[id^=he]{
            background: lightblue;
        }
        p[id$=ld]{
            background: lightgreen;
        }
    </style>
</head>
<body>
    <p>hello!</p>
    <p id="hello">hello!</p>
    <p id="world">hello!</p>
</body>
</html>
```

hello!

hello!

hello!

3.7 结构伪类选择器

```
:first-child 用于父元素的第一个子元素设置样式
:last-child 用于父元素的最后一个子元素设置样式
:nth-child(n) 用于父元素的第n个子元素设置样式
:first-of-type 用于选择父元素下相同类型子元素的第一个
:last-of-type 用于选择父元素下相同类型子元素的最后一个
:nth-of-type(n)选择器：用于选择父元素下相同类型子元素的第n个
```

3.7.1 E:first-child 选择器

E:first-child 选择器 : E 表示 HTML 标签类型 , 该选择器 选择 匹配的父容器 中的 第一个 E 类型标签 子元素 ;

3.7.2 E:last-child 选择器

E:last-child 选择器 : 该选择器 选择 匹配的父容器 中的 最后一个 E 类型标签 子元素 ;

```
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        ul li:first-child{
            background: lightblue;
        }
        ul li:last-child{
            background: lightgreen;
        }
    </style>
</head>
<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
        <li>4</li>
    </ul>
</body>
</html>
```

- 1
- 2
- 3
- 4

3.7.3 E:nth-child(n) 选择器

E:nth-child(n) 选择器 : 该选择器 选择 匹配的父容器 中的 第 n 个 E 类型标签 子元素 ;

- 数字 : 数字代表 父容器中子元素的索引值 ; n 从 1 开始计数 ;
- 关键字 : 借助下面的 奇数 / 偶数 关键字 , 可以实现各行变色效果 ;
    - 偶数关键字 : even ;
    - 奇数关键字 : odd ;
- 公式 : 公式中的 n 是从 0 开始计数的 , 数字中的 n 是从 1 开始计数的 ;
    - n : 表示所有的数值 , 取值范围 { n = 0 , 1 , 2 , 3 , 4 … } 非负整数 ;
    - 2n : 表示偶数 ;
    - 2n + 1 : 表示奇数 ;
    - 5n : 表示 第 5 / 10 / 15 / 20 / 25 等索引数字 ; 将 { n = 0 , 1 , 2 , 3 , 4 … } 代入即可

如果 n 索引 是 第 0 个元素 , 或者 超出 了元素个数 , 该选择器会被忽略 ;

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        ul li:nth-child(odd){
            background: lightblue;
        }
        ul li:nth-child(even){
            background: lightgreen;
        }
    </style>
</head>
<body>
    <ul>
        <li>1</li>
        <li>2</li>
        <li>3</li>
        <li>4</li>
    </ul>
</body>
</html>
```

- 1
- 2
- 3
- 4

E:nth-child(n) 选择器 缺陷 : 如果 父容器 中的子元素类型不同 , 那么使用 E:nth-child(n) 选择器 选择标签 , 必须精准的指出要选择的子元素的索引和类型 , 设置错误 , 则无法选择出想要的标签 ;

3.7.4 E:first-of-type / E:last-of-type / E:nth-of-type 选择器

- E:first-of-type 选择器 : 该选择器 指定 父容器中第一个 E 类型标签 ;
- E:last-of-type 选择器 : 该选择器 指定 父容器中最后一个 E 类型标签 ;
- E:nth-of-type 选择器 : 该选择器 指定 父容器中第 n 个 E 类型标签 ;

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        div span:first-of-type {
            /* 选择 div 父容器下的 第一个 span 标签 */
            background-color: aqua;
        }
        div span:nth-of-type(2) {
            /* 选择 div 父容器下的 第二个 span 标签 */
            background-color: red;
        }
    </style>
</head>
<body>
    <div>
        <span>span 标签 (第 2 个元素)</span><br>
        <span>span 标签 (第 3 个元素)</span><br>
        <span>span 标签 (第 4 个元素)</span>
    </div>
</body>
</html>
```

span 标签 (第 2 个元素)
span 标签 (第 3 个元素)
span 标签 (第 4 个元素)

4.常用CSS属性

4.1 字体样式

```
font-family:设置字体类型
font-size:设置字体大小
font-style:设置字体风格
font-weight:设置字体粗细
font:用于字体综合设置

可以同时指定多个字体
p{font-family:Arial,"Times New Roman","微软雅黑","黑体","宋体"}
1.多个字符之间用英文字符隔开
2.英文字体位于中文字体之前
3.中文字体需要添加英文符号
4.英文字体中包含空格，#，￥等符号，须添加英文引号
5.当浏览器不支持第一个字体时，会尝试下一个，直到匹配到支持的字体为止。
如果都不支持，则使用默认的字体
```

4.2 文本样式

```
color 设置字体颜色
text-align 设置字体文本水平对齐的方式
text-indent 设置文本首行缩进
line-height 设置文本行高
text-decoration 设置文本装饰
vertival-align 设置文本垂直对齐方式
text-shadow 设置文本阴影效果
```

4.3 列表样式

```
list-style-type:列表项标记的类型
list-style-image:使用图片替换列表项的标记
list-style-position:设置在何处放置列表项的标记
list-style:设置所有的列表属性

list-style-type常用来去除无序列表的小圆点
list-style-type:none;
```

4.4 背景属性

```
background-color 背景颜色
background-image 背景图片
background-repeat 规定如何重复图片
background-position 背景位置
background-size 背景尺寸
background-image:linear-gradient(方向，颜色，颜色)
```

4.4.1 background-repeat

```
background-repeat 规定如何重复图片
repeat 默认
repeat-x 背景图像在水平方向重复
repeat-y 背景图像在垂直方向重复
no-repeat 背景图像仅显示一次
```

4.4.2 background

```
background 复合属性，可以将背景相关的样式都定义在符合属性中
background:背景图像 背景定位 背景重复方式
```

4.4.3 渐变

```
语法：
background-image:linear-gradient(direction,color-stopl,color-stop2,....);
direction 渐变方向
color-stopl,color-stop2 渐变颜色
```

```html
<!DOCTYPE html>
<html lang="en">
<head>
    <meta charset="UTF-8">
    <title>Document</title>
    <style>
        p{
            background-image: linear-gradient(yellow,green);
        }
    </style>
</head>
<body>
    <p>哈哈哈哈哈哈</p>
</body>
</html>
```

哈哈哈哈哈哈