

# 层叠样式表CSS（上）

(Cascade Style Sheet)

2019.3.21

isszym sysu.edu.cn

[runoob](#) [w3.org](#) [w3school](#)

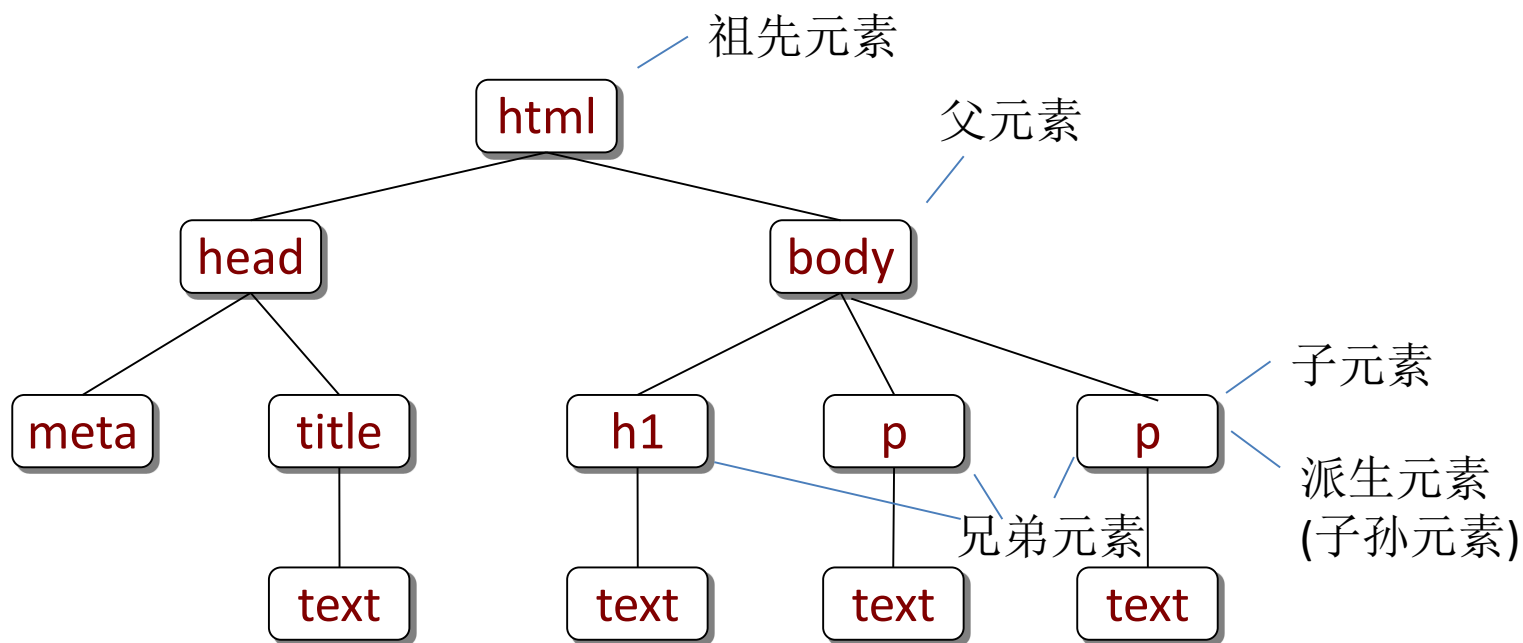
# 目录

- 概述
  - HTML网页
  - 文档树
- 样式表
  - 概述
  - 选择器
    - 基本选择器
    - 复合选择器
    - 属性选择器
    - 伪元素选择器
    - 伪类选择器
  - 样式可以放在哪里？
  - 样式层叠
- 附录：选择器大全

# 概述

- HTML具有一些表现元素或表现属性，例如：设置文字字体和颜色的元素**font**，用于文字对中的元素**center**，设置背景颜色的**bgcolor**属性等，但是，过多地使用这些元素会使内容与表现混合在一起而令HTML源代码难以阅读和维护，为此，W3C定义了层叠样式表。
- 层叠样式表是由样式（**style**）组成的，每个样式都定义一种表现方式，并作用于一个或一组元素。
- CSS功能主要有三个版本。CSS1和CSS2是由W3C分别于1996年12月17日和1999年1月11日发布的两个版本。CSS2的修订版CSS2.1得到了绝大部分浏览器的支持。
- CSS3采用了模块化方法划分样式。CSS3尽管还没有正式发布，但是其从2001年开始逐步公布的一些新模块已获得大部分主流浏览器的支持。

- 使用层叠样式表，除了可以利用id属性等选择一个或多个元素设置样式，还可以利用文档树的兄弟关系、父子关系和祖孙关系选择元素进行样式设置。



```
<!DOCTYPE html>
<html lang="zh-cn">
<head>
  <meta charset="utf-8">
  <title>This is my title</title>
</head>
<body>
  <h1>This is header</h1>
  <p>This is the first paragraph</p>
  <p>This is the second paragraph</p>
</body>
</html>
```

# 样式表

## • 默认样式

显示网页是否必须定义样式呢？当然不是，没有定义样式，浏览器会使用默认的样式。下图展示的是IE浏览器和Firefox浏览器下显示同一则新闻的情况。可以看出，IE浏览器默认采用的是宋体字而Firefox浏览器用的是黑体字，而且行高还不一样。



IE



FireFox

- 上面的问题可以通过为**body**或者\*（所有元素）增加**样式(style)**来解决。

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>默认样式</title>
  <style type="text/css">
    body {
      font-family: 宋体;
      line-height: 1.2em;
    }
  </style>
</head>
<body>
```

定义相同的字体和行高

```
  <h1>俄飞行员坠机北极圈 在浮冰上与北极熊斗智斗勇</h1>
```

```
  <p>历经直升飞机坠落、整整两天在浮冰上的生死漂流、与三只北极熊斗智斗勇，俄罗斯飞行员谢尔盖·阿纳诺夫终于等来了救援队——当地时间7月27日，加拿大海岸巡防船在巴芬岛和格陵兰岛之间的戴维斯海峡发现了他。</p>
```

```
  <p id="start" class="even">三天前，阿纳诺夫独自驾驶直升飞机，计划从加拿大城市伊魁特飞往格陵兰岛，飞行路程超过800公里。</p>
```

```
  <p>这次飞行是他环游世界计划的一部分，6月13日，阿纳托夫在莫斯科开启了他的环球之旅。若不发生这次意外，他便可以成为独自驾驶小型直升飞机绕飞北极圈的第一人。</p>
```

```
  <p>据CBC报道，阿纳诺夫驾驶罗宾逊R22直升机从伊魁特起飞时，天气状况良好。但飞行不久，直升机动力传输出现故障失去控制，随后坠入戴维斯海峡。他钻出直升机得以脱险，但身上只有三个照明弹、一个救生艇和少量物资。</p>
```

```
  <p class="even">阿纳诺夫游到附近的浮冰上，展开自救。身着普通衣物的他由于浑身湿透，只得穿上救生衣避寒，用救生艇遮挡寒风。气温在5摄氏度左右徘徊。他发射了两枚照明弹，但由于雾气越来越重，搜救队并不容易找到他。他把冰块含在嘴里获取水分。</p>
```

```
  <p>但阿纳诺夫的主要威胁来自三位不速之客——三只北极熊。在风力作用下，其他浮冰很容易被吹到阿纳诺夫盘卧的浮冰边，导致北极熊能轻易接近他。</p>
```

```
  <p class="even">“我不得不与它们斗智斗勇，试图吓跑这些‘访客’。”阿纳诺夫告诉CBC，他朝北极熊又喊又叫，才得以幸运脱险。</p>
```

```
  <p>在浮冰上困了两天之后，浓雾散去，阿纳诺夫终于看到希望之光：一艘加拿大海岸巡防船正经过。他释放了最后一只照明弹，引起海岸巡防队的注意，获得营救。加拿大联合搜救协调中心确认了阿纳诺夫获救的消息，他将被送回伊魁特，安排前往首都渥太华，随后回到俄罗斯。</p>
```

```
</body>
```

```
</html>
```



# • 设置样式

```
<!DOCTYPE html>
<html>
<head>
  <meta http-equiv="Content-Type" content="text/html; charset=utf-8"/>
  <title>默认样式</title>
  <style type="text/css">
    p {
      font-family: 宋体;
      line-height: 20px;
      text-indent: 1.5em;
    }
    #start {
      font-weight: bold;
    }
    .even {
      font-style: italic;
    }
  </style>
</head>
<body>
  <h1>俄飞行员坠机北极圈 在浮冰上与北极熊斗智斗勇</h1>
  <p>历经直升飞机坠落、整整两天在浮冰上的生死漂流、与三只北极熊斗智斗勇，俄罗斯飞行员谢尔盖·阿纳诺夫终于等来了救援队——当地时间7月27日，加拿大海岸巡防船在巴芬岛和格陵兰岛之间的戴维斯海峡发现了他。</p>
  <p id="start" class="even next">三天前，阿纳诺夫独自驾驶直升飞机，计划从加拿大城市伊魁特飞往格陵兰岛，飞行路程超过800公里。</p>
  <p>这次飞行是他环游世界计划的一部分，6月13日，阿纳托夫在莫斯科开启了他的环球之旅。若不发生这次意外，他便可以成为独自驾驶小型直升飞机绕飞北极圈的第一人。</p>
  <p>据CBC报道，阿纳诺夫驾驶罗宾逊R22直升机从伊魁特起飞时，天气状况良好。但飞行不久，直升机动力传输出现故障失去控制，随后坠入戴维斯海峡。他钻出直升机得以脱险，但身上只有三个照明弹、一个救生艇和少量物资。</p>
  <p class="even">阿纳诺夫游到附近的浮冰上，展开自救。身着普通衣物的他由于浑身湿透，只得穿上救生衣避寒，用救生艇遮挡寒风。气温在5摄氏度左右徘徊。他发射了两枚照明弹，但由于雾气越来越重，搜救队并不容易找到他。他把冰块含在嘴里获取水分。</p>
  <p>但阿纳诺夫的主要威胁来自三位不速之客——三只北极熊。在风力作用下，其他浮冰很容易被吹到阿纳诺夫盘卧的浮冰边，导致北极熊能轻易接近他。</p>
  <p class="even">“我不得不与它们斗智斗勇，试图吓跑这些‘访客’。”阿纳诺夫告诉CBC，他朝北极熊又喊又叫，才得以幸运脱险。</p>
  <p>在浮冰上困了两天之后，浓雾散去，阿纳诺夫终于看到希望之光：一艘加拿大海岸巡防船正经过。他释放了最后一枚照明弹，引起海岸巡防队的注意，获得营救。加拿大联合搜救协调中心确认了阿纳诺夫获救的消息，他将被送回伊魁特，安排前往首都渥太华，随后回到俄罗斯。</p>
</body>
</html>
```





```
p {
  font-family: 宋体;
  line-height: 20px;
  text-indent: 1.5em;
}
```

元素选择器

```
#start {
  font-weight: bold;
}
```

ID选择器

```
.even {
  font-style: italic;
}
```

类选择器

tp://localhost:64061/HtmlPage1.html

默认样式

文件(F) 编辑(E) 查看(V) 收藏夹(A) 工具(T) 帮助(H)

Java内存结构、类... Java中的main线程... 2345网址导航 - 开...

<h1>俄飞行员坠机北极圈 在浮冰上与北极熊斗智斗勇</h1>

<p id="start"  
class="even next">

<p>历经直升飞机坠落、整整两天在浮冰上的生死漂流、与三只北极熊斗智斗勇，俄罗斯飞行员谢尔盖·阿纳诺夫终于等来了救援队——当地时间7月27日，加拿大海岸巡防船在巴芬岛和格陵兰岛之间的戴维斯海峡发现了他。</p>

——三天前，阿纳诺夫独自驾驶直升飞机，计划从加拿大城市伊魁特飞往格陵兰岛，飞行路程超过800公里。</p>

<p>这次飞行是他环游世界计划的一部分，6月13日，阿纳托夫在莫斯科开启了他的环球之旅。若不发生这次意外，他便可以成为独自驾驶小型直升飞机绕飞北极圈的第一人。</p>

<p>据CBC报道，阿纳诺夫驾驶罗宾逊R22直升机从伊魁特起飞时，天气状况良好。但飞行不久，直升机动力传输出现故障失去控制，随后坠入戴维斯海峡。他钻出直升机得以脱险，但身上只有三个照明弹、一个救生艇和少量物资。</p>

<p class="even">

阿纳诺夫游到附近的浮冰上，展开自救。身着普通衣物的他由于浑身湿透，只得穿上救生衣避寒，用救生艇遮挡寒风。气温在5摄氏度左右徘徊。他发射了两枚照明弹，但由于雾气越来越重，搜救队并不容易找到他。他把冰块含在嘴里获取水分。</p>

<p>但阿纳诺夫的主要威胁来自三位不速之客——三只北极熊。在风力作用下，其他浮冰很容易被吹到阿纳诺夫盘卧的浮冰边，导致北极熊能轻易接近他。</p>

<p class="even">

“我不得不与它们斗智斗勇，试图吓跑这些‘访客’。”阿纳诺夫告诉CBC，他朝北极熊又喊又叫，才得以幸运脱险。</p>

<p>在浮冰上困了两天之后，浓雾散去，阿纳诺夫终于看到希望之光：一艘加拿大海岸巡防船正经过。他释放了最后一枚照明弹，引起海岸巡防队的注意，获得营救。加拿大联合搜救协调中心确认了阿纳诺夫获救的消息，他将被送回伊魁特，安排前往首都渥太华，随后回到俄罗斯。</p>

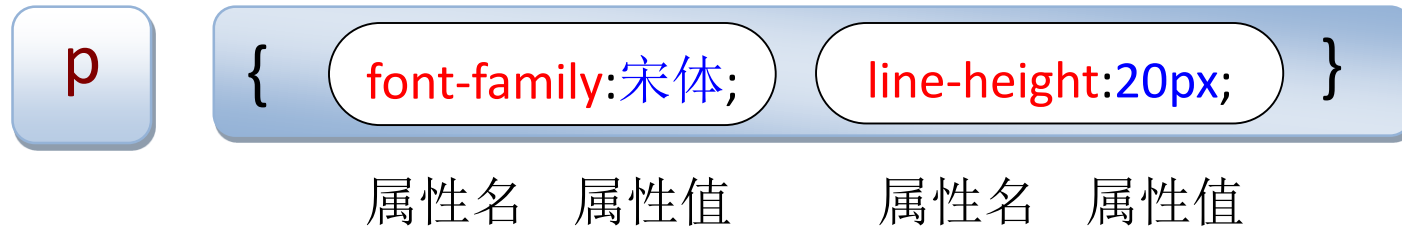


# • CSS选择器(selector)

[w3c](#)

选择器

声明块



元素选择器: `p {font-size: 14px;}`

选择所有p元素 (type selector)

ID选择器: `#start {font-weight: bold;}`

选择id为start的元素

类选择器: `.even {font-style: italic;}`

选择所有类为even的元素

通配选择器 `* {color: blue;}`

选择所有元素 (universal selector)

## 复合选择器

采用结合符( Combinators)可以形成复合选择器。

后代选择器:	<code>p span {font-size: 14px;}</code>	选择p元素的所有子孙元素中的span元素
子元素选择器:	<code>p &gt; span {font-size: 14px;}</code>	选择p元素的所有子女元素中的span元素
相邻兄弟选择器:	<code>h1 + p {color:red;}</code>	选择h1之后的相邻兄弟元素(必须为p元素)
普通兄弟选择器:	<code>h1 ~ p {color:red;}</code>	选择在 h1之后所有兄弟元素中的p元素

直接组合器	<code>p#start {color:red;}</code>	选择id为start的p元素(交集)
群组选择器:	<code>em, .even {color:red;}</code>	选择em元素或者类名为even的元素
否定选择器:	<code>p:not(#start){ color:red;}</code>	选择所有id不是start的p元素

- \* Descendant selector: 后代选择器, 又称为包含选择器或派生选择器
- \* Child selector: 子元素选择器, 也称为子女选择器或子选择器
- \* Adjacent selector: 相邻选择器, General sibling selector: 普通兄弟选择器
- \* Groups of selectors: 群组选择器

<code>#main div .left</code>	id为主ain的后代中的div元素的后代中的类名为left被选中
<code>#main&gt;div .left</code>	id为主ain的子女中的div元素的后代中的类名为left被选中
<code>#main div&gt;.left</code>	id为主ain的后代中的div元素的子女中的类名为left被选中
<code>div#main.left</code>	id为主ain类名为left的div元素被选中

如果div 变为\*, ?

## 属性选择器 (Attribute selector)

[参考](#)

`[alt] { width:200px; }`

选择所有具有属性alt的元素

`img[alt] { width:200px; }`

选择所有具有属性alt的img元素

`[alt="fig 2-1"] { width:20px; }`

选择所有属性alt的值等于"figure 2-1"的元素。

`img[alt^="fig"] { width:20px; }`

选择alt值以“figure”开头的所有img元素。

`img[alt$="2-1"] { width:20px; }`

选择alt值以“2-1”结尾的所有img元素。

`img[alt*="5-"] { width:20px; }`

选择alt值包含字符串“5-”的所有img元素。

`img[alt~="fig"] { width:20px; }`

选择alt值包含单词“fig”的所有img元素。

``

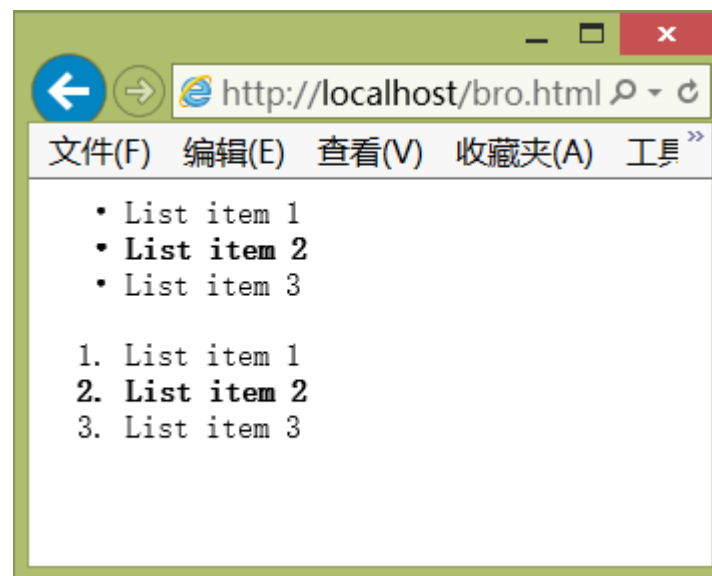
`div p *[href]?`

div的子孙p的带有属性href的子孙

```

<html>
<head>
<style type="text/css">
    .first + li { font-weight:bold;}
</style>
</head>
<body>
<div>
    <ul>
        <li class="first">List item 1</li>
        <li>List item 3</li>
        <li>List item 3</li>
    </ul>
    <ol>
        <li class=" first ">List item 1</li>
        <li>List item 2</li>
        <li>List item 3</li>
    </ol>
</div>
</body>
</html>

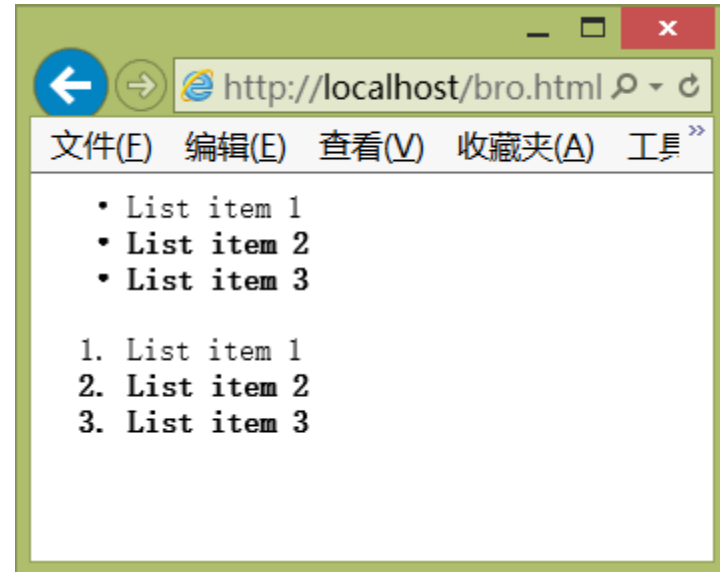
```



```

<html>
<head>
<style type="text/css">
    .first ~ li { font-weight:bold;}
</style>
</head>
<body>
<div>
    <ul>
        <li class="first">List item 1</li>
        <li class="para">List item 2</li>
        <li>List item 3</li>
    </ul>
    <ul>
        <li class="first para">List item 1</li>
        <li>List item 2</li>
        <li>List item 3</li>
    </ul>
</div>
</body>
</html>

```



## 伪元素选择器(pseudo-element):

`p:first-letter {font-size:2em;}`  
`p:first-line {color:red;}`  
`p:after {content:"..."; color:red;}`  
`p:before {content:url(a.jpg);}`  
`::selection`

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
p:first-letter {font-size:2em}
p:first-line {color:red}
p:after{content:"..."}
</style>
</head>
<body>
<h1>This is a heading.</h1>
<p>This is paragraph.This is paragraph.This is
paragraph.This is paragraph.This is paragraph.This is
paragraph.This is paragraph.This is paragraph.This is
paragraph.This is paragraph.</p>
<p>This is paragraph.</p>
<p>This is paragraph.</p>
<p>This is paragraph.</p>
<p>This is paragraph.</p>
</body>
</html>
```

p元素内容的第一个字母

p元素内容的第一行

在p元素的内容之后插入红色省略号

在p元素的内容之后插入图像a.jpg

为选中的文本设置样式

This is a heading.

This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.This is paragraph.

This is paragraph...

This is paragraph...

This is paragraph...

This is paragraph...

\* 在 CSS3 中伪元素采用双冒号::，例如，`p::after`，而伪类采用:。::也可以写成:。

## 伪类选择器(pseudo-class): LoVe/HAtE

a:link {color:blue;}

a:visited {color:red;}

a:hover {font-size:2em;}

a:active {color:green;}

input:focus {background-color:red;}

没有访问过的链接

访问过的链接

鼠标悬停下的元素(很多元素都可以用)

正在访问(按下鼠标)的链接

获得焦点的元素(一般为输入元素)

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
  a:link {color:blue;}
  a:visited {color:red;}
  a:hover {font-size:2em;}
  a:active {color:green;}
  input:focus {background-color:red;}
  ::selection {color:brown;}
</style>
</head>
<body>
<h1>This is a heading.</h1>
<a href="sohu.com">This is paragraph.</a>
<a href="sina.com">This is paragraph.</a>
<a href="#">This is paragraph.</a>
<p>This is paragraph.</p>
<p>This is paragraph.</p>
输入1: <input type="text" value="12345678">
输入2: <input type="text" value="abcdefgh">
输入3: <input type="text" value="ABCDEFGH">
</body>
</html>
```

This is a heading.

[This is paragraph.](#) [This is paragraph.](#) [This is paragraph.](#)

This is paragraph.

This is paragraph.

输入1:

输入2:

输入3:

**E:checked** 具有属性checked的元素E. 例如: input:checked+label  
**E:disabled** 具有属性disabled的元素E  
**E:enabled** 具有属性eabled的元素E  
**E:root** 选择元素E的根元素(html). 例如: :root {color:red;}  
**E:empty** 选择不包含任何文本或其它元素的元素E  
**E:target** 选择点击a元素所选择的书签

链接选择器a:... 动态选择器(:hover :active)



## nth-child选择器(结构伪类):

## Structural pseudo-classes

`p:first-child {color:red;}`

p的双亲的第一个子女

`p:last-child {color:red;}`

p的双亲的最后一个子女

`p:only-child {color:red;}`

p的双亲的唯一一个子女(只有一个子女)

`p:nth-child(5) {color:red;}`

p的双亲的第5个子女

`p:nth-child(even) {color:red;}`

p的双亲的第偶数个子女 (奇数:odd)

`p:nth-child(3n+1) {color:red;}`

p的双亲的选择第1、4、7、...个子女

`p:nth-last-child(5) {color:red;}`

p的双亲的倒数第5个子女

\* 双亲的各种类型的子女元素均参与计数, 但是**只有相应位置为元素p才会被选中**。 \* n = 0, 1, 2, 3, ...

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
  p:nth-child(even) {color:red;}
</style>
</head>
<body>
  <p>Let's see </p>
  <h1>This is a heading.</h1>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
</body>
</html>
```

Let's see

**This is a heading.**

This is paragraph.

**This is paragraph.**

This is paragraph.

**This is paragraph.**

This is paragraph.

## nth-of-type选择器(结构伪类):

p:first-of-type {color:red;}  
p:last-of-type {color:red;}  
p:only-of-type {color:red;}  
p:nth-of-type(5) {color:red;}  
p:nth-of-type(even) {color:red;}  
p:nth-of-type(3n+1) {color:red;}  
p:nth-last-of-type(5) {color:red;}  
p:nth-last-of-type(even) {color:red;}

\* 双亲的子女元素中只有p元素参与计数。

p的双亲的第一个子女(只计算p元素)  
p的双亲的最后一个子女  
p的双亲的唯一一个子女(只有一个子女)  
p的双亲的第5个子女  
p的双亲的第偶数个子女  
p的双亲的选择第1、3、6、...个子女  
p的双亲的倒数第5个子女

```
<!DOCTYPE HTML>
<html>
<head>
<style type="text/css">
  p:nth-of-type(even) {color:red;}
</style>
</head>
<body>
  <p>Let's see </p>
  <h1>This is a heading.</h1>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
  <p>This is paragraph.</p>
</body>
</html>
```

Let's see

This is a heading.

This is paragraph.

This is paragraph.

This is paragraph.

This is paragraph.

This is paragraph.

## nth-child的其它用法:

:nth-child(n+4)

选中从第4个开始的子元素

:nth-child(-n+12)

选中从第1个到第12个子元素

:nth-child(n+4):nth-child(-n+12)

选中第4~12个子元素

:nth-child(n+4):nth-child(-n+12):nth-child(odd)

选中的子元素是从第4位到第12位，  
并且只包含奇数位，既选中第5、7、  
9、11个元素。

\* 多个组合相当于交集运算

:nth-child(2n+4):nth-child(-3n+20) ?

nth-child(2n+4) 4 6 8 10 12 14 16 18 20

nth-child(-3n+20) 20 17 14 11 8 5 2

## • 样式的继承性

很多时候一个元素的样式并非直接对元素定义，而是对其祖先元素定义，然后通过**继承性**得到样式，例如，在body元素中定义的文字颜色(color)样式，在其子孙元素中也有效。其他像font-family, line-height等都有继承性。通过继承性可以简化CSS的编写。

有些样式属性默认是没有继承性的，而有些元素也不会继承某些样式属性。例如，border没有继承性，input和textarea元素不会继承任何字体的属性。可以使用通配符强制所有元素继承某些属性：

```
* { font-family: inherit;
    line-height: inherit;
    color: inherit;
}
html { font-size: 125%;
       font-family: sans-serif;
       line-height: 1.5;
       color: #222;
}
h1 {
    font-size: 3rem;
}
```

# • 样式定义可以放在哪里？

(1) 直接放在元素的style属性中。这种样式称为**内联样式**(inline style)。

```
<p style="font-size:2em">
  This is paragraph.
</p>
```

(2) 用style元素存放。 style元素可以放在html文档的任何地方， 并可以多次出现。一般放在元素head中。这种样式表称为**内部样式表**(Internal style sheet)

```
<style type="text/css">                                注释: // 和 /* */
  p {font-size:2em}
</style>
```

(3) 存放在一个或多个外部文件(.css)中，即采用**外部样式表**(External style sheet)。在head元素中定义link元素把样式引入网页，

例如<link type="text/css" href="/css/main.css">

css文件还可以在style元素中或者css文件中用@import语句 加入  
@import url(/css/main.css);

例如，文件main.css包含： p {font-size:2em}

MIME类型



\* link也可以用来引入地址栏图标： <link rel="icon" href="favicon.ico" type="image/x-icon" />

## main.html

```
<!DOCTYPE HTML>
<html>
<head>
<link type="text/css" href= "main.css" >
<style type="text/css">
  body {color:yellow}
  p {color:red}
</style>
</head>
<body>
<h1>This is a heading.</h1>
<p>This is paragraph.This is paragraph.This is
paragraph.This is paragraph.This is
paragraph.This is paragraph.This is
paragraph.This is paragraph.</p>
<p>This is paragraph.</p>
<p style="color:blue"> This is paragraph.</p>
<p>This is paragraph.</p>
<p>This is paragraph.</p>
</body>
</html>
```

## main.css

```
p          {color:green}
p:hover    {color:blue}
```

## • 样式层叠

如果一个元素在多个地方定义了某个样式，最后起作用的是哪个呢？

例如：`body`中定义了字体颜色，某个元素`p`的属性`style`中也定义了文字颜色，内部样式表和外部样式表中都定义了元素`p`的文字颜色，这会导致复杂的层叠关系，到底谁可以起作用呢？

首先，要把所有组合样式拆解成最小样式属性，也就是不可能再拆的样式属性，然后才来看谁能起作用。

例如，`p {font-size: 32px; text-indent: 1.5em}` 和 `p {border-left: solid 1px black}`

拆成     `p {font-size: 32px;}`  
          `p { text-indent: 1.5em}`

`p {border-left-style: solid}`  
`p {border-left-width: 1px }`  
`p {border-left-color: black}`

对于一个元素的某个最小样式属性，如果被多个选择器所作用，就出现了**样式层叠**，CSS需要通过选择器的特殊性(`specificity`)来确定要使用的哪个选择器设置的属性值。

**越特殊的选择器优先权越高**。如果具有多个最高优先权的选择器，使用离元素最近的选择器。一个元素只会被在它之前定义的选择器所作用。



## 选择器的特殊性的值可以由下面规则确定：

- (1) 每个元素只能定义一个行内样式，行内样式的特殊性为1, 0, 0, 0。
- (2) 对于ID选择器，每个特殊性加0, 1, 0, 0。
- (3) 对于类选择器、属性选择器、伪类选择器，每个特殊性加0, 0, 1, 0。
- (4) 对于元素选择器和伪元素选择器，每个特殊性加0, 0, 0, 1。
- (5) 结合符和通配选择器，对特殊性没有任何贡献，即其特殊性为0, 0, 0, 0。
- (6) 继承得来的选择器没有任何特殊性，即其特殊性为0, 0, 0, 0。
- (7) 加上!important的样式具有最高优先权。例如，p {color:blue!important; }

\* 结合符: ~ > + 等

优先权: 0, 1, 0, 0 > 0, 0, 2, 0  
0, 0, 2, 1 > 0, 0, 2, 0

试一下计算特殊性：

p	{color:blue}	0, 0, 0, 1
body p	{color:red}	0, 0, 0, 2
p.hot	{color:green}	0, 0, 1, 1
p#start	{color:black}	0, 1, 0, 1
body > div ul li[id^="files"] ul li:hover	{color:yellow}	0, 0, 2, 6

h1, h2.section {color:blue; font-size:12px}



分解为四个选择器

h1 {color:blue; } /\* 0,0,0,1 \*/

h1 {font-size:12px} /\* 0,0,0,1 \*/

h2.section {color:blue; } /\* 0,0,1,1 \*/

h2.section {font-size:12px} /\* 0,0,1,1 \*/

li:nth-child(n+3):nth-child(-n+9) a:hover span {color: red; }

?

\*使用包含ID选择器的复合选择器可以用于区分不同网页中具有相同id的元素。

.login #start {color:black}

.main #start {color:blue}

<html>...<body class="login"> ... </body></html>

<html>...<body class="main">...</body></html>

# 附录1：选择器大全

选择器	例子	例子描述	CSS
<a href="#"><u>.class</u></a>	.intro	选择 class="intro" 的所有元素。	1
<a href="#"><u>#id</u></a>	#firstname	选择 id="firstname" 的所有元素。	1
<a href="#"><u>*</u></a>	*	选择所有元素。	2
<a href="#"><u>element</u></a>	p	选择所有 <p> 元素。	1
<a href="#"><u>element,element</u></a>	div,p	选择所有 <div> 元素和所有 <p> 元素。	1
<a href="#"><u>element element</u></a>	div p	选择 <div> 元素内部的所有 <p> 元素。	1
<a href="#"><u>element&gt;element</u></a>	div>p	选择父元素为 <div> 元素的所有 <p> 元素	2
<a href="#"><u>element+element</u></a>	div+p	选择紧接在 <div> 元素之后的 <p> 元素。	2
<a href="#"><u>[attribute]</u></a>	[target]	选择带有 target 属性所有元素。	2
<a href="#"><u>[attribute=value]</u></a>	[target=_blank]	选择 target="_blank" 的所有元素。	2
<a href="#"><u>[attribute~=value]</u></a>	[title~=flower]	选择 title 属性包含单词 "flower" 的所有元素。	2
<a href="#"><u>[attribute =value]</u></a>	[lang =en]	选择 lang 属性值以 "en" 开头的元素	2
<a href="#"><u>:link</u></a>	a:link	选择所有未被访问的链接。	1
<a href="#"><u>:visited</u></a>	a:visited	选择所有已被访问的链接。	1
<a href="#"><u>:active</u></a>	a:active	选择活动链接。	1
<a href="#"><u>:hover</u></a>	a:hover	选择鼠标指针位于其上的链接。	1
<a href="#"><u>:focus</u></a>	input:focus	选择获得焦点的 input 元素。	2

选择器	例子	例子描述	CSS
<a href="#"><u>:first-letter</u></a>	p:first-letter	选择每个 <p> 元素的首字母。	1
<a href="#"><u>:first-line</u></a>	p:first-line	选择每个 <p> 元素的首行。	1
<a href="#"><u>:first-child</u></a>	p:first-child	选择属于父元素的第一个子元素的每个 <p> 元素。	2
<a href="#"><u>:before</u></a>	p:before	在每个 <p> 元素的内容之前插入内容。	2
<a href="#"><u>:after</u></a>	p:after	在每个 <p> 元素的内容之后插入内容。	2
<a href="#"><u>:lang(<i>language</i>)</u></a>	p:lang(it)	选择带有以 "it" 开头的 lang 属性值的每个 <p> 元素。	2
<a href="#"><u>element1~element2</u></a>	p~ul	选择前面有 <p> 元素的每个 <ul> 元素。	3
<a href="#"><u>[attribute^=value]</u></a>	a[src^="https"]	选择其 src 属性值以 "https" 开头的每个 <a> 元素。	3
<a href="#"><u>[attribute\$=value]</u></a>	a[src\$=".pdf"]	选择其 src 属性以 ".pdf" 结尾的所有 <a> 元素。	3
<a href="#"><u>[attribute*=value]</u></a>	a[src*="abc"]	选择其 src 属性中包含 "abc" 子串的每个 <a> 元素。	3
<a href="#"><u>:first-of-type</u></a>	p:first-of-type	选择属于其父元素的首个 <p> 元素的每个 <p> 元素。	3
<a href="#"><u>:last-of-type</u></a>	p:last-of-type	选择属于其父元素的最后 <p> 元素的每个 <p> 元素。	3
<a href="#"><u>:only-of-type</u></a>	p:only-of-type	选择属于其父元素唯一的 <p> 元素的每个 <p> 元素。	3
<a href="#"><u>:only-child</u></a>	p:only-child	选择属于其父元素的唯一子元素的每个 <p> 元素。	3
<a href="#"><u>:nth-child(<i>n</i>)</u></a>	p:nth-child(2)	选择属于其父元素的第二个子元素的每个 <p> 元素。	3
<a href="#"><u>:nth-last-child(<i>n</i>)</u></a>	p:nth-last-child(2)	同上，从最后一个子元素开始计数。	3
<a href="#"><u>:nth-of-type(<i>n</i>)</u></a>	p:nth-of-type(2)	选择属于其父元素第二个 <p> 元素的每个 <p> 元素。	3
<a href="#"><u>:nth-last-of-type(<i>n</i>)</u></a>	p:nth-last-of-type(2)	同上，但是从最后一个子元素开始计数。	3

选择器	例子	例子描述	CSS
<a href="#"><u>:last-child</u></a>	p:last-child	选择属于其父元素最后一个子元素每个 <p> 元素。	3
<a href="#"><u>:root</u></a>	:root	选择文档的根元素。	3
<a href="#"><u>:empty</u></a>	p:empty	选择没有子元素的每个 <p> 元素（包括文本节点）。	3
<a href="#"><u>:target</u></a>	#news:target	选择当前活动的 #news 元素。	3
<a href="#"><u>:enabled</u></a>	input:enabled	选择每个启用的 <input> 元素。	3
<a href="#"><u>:disabled</u></a>	input:disabled	选择每个禁用的 <input> 元素	3
<a href="#"><u>:checked</u></a>	input:checked	选择每个被选中的 <input> 元素。	3
<a href="#"><u>:not(selector)</u></a>	:not(p)	选择非 <p> 元素的每个元素。	3
<a href="#"><u>::selection</u></a>	::selection	选择被用户选取的元素部分。	3