# 列表标签

一、什么是CSS列表？

CSS列表是[HTML](http://www.w3cschool.cn/html)里一种很有用的显示方式，可以把相关的并列内容整齐地垂直排列，使网页显得整洁专业，并让浏览者有一目了然的感觉。 样式表为列表增加了一些功能，控制列表的样式包括[列表样式](http://www.w3cschool.cn/css/css-list.html)、图形符号、列表位置三个部分。

1．列表符号

列表符号是指显示于每一个列表项目前的符号标识。

基本格式如下：

list-style-type：参数

参数取值范围：

·disc：圆形

·circle：空心圆

·square：方块

·decimal：十进制数字

·lower-roman：小写罗马数字

·upper-roman：大写罗马数字

·lower-alpha：小写希腊字母

·upper-alpha：大写希腊字母

·none：无符号显示

{background: 颜色 } 安全显示列表符

参数中的disc是默认选项。

2．图形符号

图形符号指原来列表的项目符号将可以使用图形来代替。

基本格式如下：

list-style-image：URL

URL是用来代替项目符号的图形文件的地址，可以使用相对地址或绝对地址。

3．列表位置列表位置描述列表在何处显示。

基本格式如下：

list-style-position：参数

参数取值范围：

·inside：在BOX模型内部显示

·outside：在BOX模型外部显示

二、CSS列表分类

CSS列表样式分为有序列表、无序列表和定义列表。

* **无序列表（可嵌套）**<ul></ul>声明列表是无序的<li></li>包含一列项目。
* **有序列表（可嵌套）**<ol></ol>声明列表是有序的<li></li>包含一列项目。
* **定义列表**（dt定义概念、dd解释；有缩进）

<dl></dl>声明定义列表

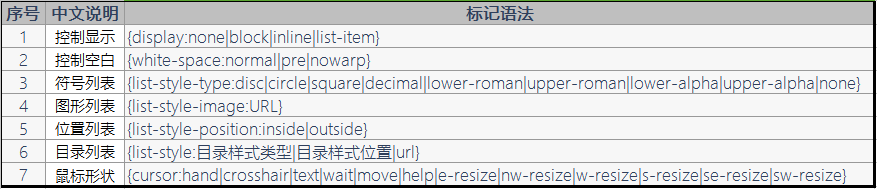
<dt></dt>概念

<dd></dd>概念的解释

有序列表是按照顺序排列如12345、ABCDE等；

无序列表包含圆点、方块、空心圆等。

CSS列表属性样式常见语法如下：



1、 控制显示：[display](http://www.w3cschool.cn/cssref/pr-class-display.html)

语法：{display:none|block|inline|list-item}

作用：改变元素的显示值，可以将元素类型线上，块和清单项目相互变换。

说明：

·none 不显示元素

·block 块显示，在元素前后设置分行符

·inline 删除元素前后的分行符，使其并入其它元素流中

·list-item 将元素设置为清单中的一行 注意：可用 display 属性值生成插入标题和补加清单或让图形变成线上显示。

2、控制空白：[white-space](http://www.w3cschool.cn/cssref/pr-text-white-space.html)

语法：{white-space:normal|pre|nowarp}

作用：控制元素內的空白。

说明：

·normal 不改变，保持缺省值，在浏览器页面长度处换行。

·pre 要求文档显示中采用源代码中的格式。

·nowarp 不让访问者在元素內换行。

3、符号列表：[list-style-type](http://www.w3cschool.cn/cssref/pr-list-style-type.html)

语法：{list-style-type:none|disc|circle|square|decimal|lower-roman|upper-roman|lower-alpha|upper-alpha}

作用：指定清单所用的强调符或编号类型

说明：

·none - 无强调符

·disc - 碟形强调符(实心圆)

·circle - 圆形强调符(空心圆)

·square - 方形强调符(实心)

·decimal - 十进制数强调符

·lower-roman - 小写罗馬字强调符

·upper-roman - 大写罗馬字强调符

·lower-alpha - 小写字母强调符

·upper-alpha - 大写字母强调符

例子：

LI.square { list-style-type: square }

UL.plain { list-style-type: none }

OL { list-style-type: upper-alpha } /\* A B C D E etc. \*/

OL OL { list-style-type: decimal } /\* 1 2 3 4 5 etc. \*/

OL OL OL { list-style-type: lower-roman } /\* i ii iii iv v etc. \*/

4、图片列表：[list-style-image](http://www.w3cschool.cn/cssref/pr-list-style-image.html)

语法：{list-style-image:URL}

作用：用于将清单中标准强调符换成所选的图形

说明： ·url - 图形URL地址

例子：

UL.check { list-style-image: url(/LI-markers/checkmark.gif) }

UL LI.x { list-style-image: url(x.png) }

1. 位置列表：[list-style-position](http://www.w3cschool.cn/cssref/pr-list-style-position.html)

语法：{list-style-position:inside|outside}

作用：用于设置强调符的缩排或伸排，这个属性可以让强调符突出于清单以外或与清单项目对齐。

说明：

·inside - 缩排，将强调符与清单项目内容左边界对齐

·outside - 伸排，强调符突出到清单项目内容左边界以外

6、 目录列表：[list-style](http://www.w3cschool.cn/cssref/pr-list-style.html)

语法：{list-style:目录样式类型|目录样式位置|url}

作用：目录样式属性是目录样式类型、目录样式位置，和目录样式图象属性的略写

说明：

·list-style-type ·list-style-position

·list-style-image

注意：这些值的细节见各个属性部分。

例子：

LI.square { list-style: square inside }

UL.plain { list-style: none }

UL.check { list-style: url(/LI-markers/checkmark.gif) circle }

OL { list-style: upper-alpha }

OL OL { list-style: lower-roman inside }

7、鼠标形状[cursor](http://www.w3cschool.cn/cssref/pr-class-cursor.html)

语法：{cursor:hand|crosshair|text|wait|move|help|e-resize|nw-resize|w-resize|s-resize|se-resize|sw-resize}

作用：CSS提供了多达13种的鼠标形状，供我们选择。

说明：

·hand 手形

·crosshair 十字形

·text 文本形

·wait 沙漏形

·move 十字箭头形

·help 问号形

·e-resize 右箭头形

·n-resize 上箭头形

·nw-resize 左上箭头形

·w-resize 左箭头形

·s-resize 下箭头形

·se-resize 右下箭头形

·sw-resize 左下箭头形

可参阅[W3Cschool官网](http://www.w3cschool.cn/)上的《[CSS 列表样式(ul)](http://www.w3cschool.cn/css/css-list.html)》及《[CSS list-style 属性](http://www.w3cschool.cn/cssref/pr-list-style.html)》

三、CSS列表样式属性list-style的使用方法

● [list-style](http://www.w3cschool.cn/cssref/pr-list-style.html)

定义:

用于在一个声明中设置一个列表的所有属性的简写属性，该属性是一个简写属性，涵盖了所有其他列表样式属性，仅作用于具有display值等于list-item的对象(如li对象)。

相关 : list-style-image || list-style-position || list-style-type

● [list-style-image](http://www.w3cschool.cn/cssref/pr-list-style-image.html)

说明:

设置或检索作为对象的列表项标记的图像。若此属性值为 none 或指定 url 地址的图片不能被显示时，[list-style-type 属性](http://www.w3cschool.cn/cssref/pr-list-style-type.html)将发生作用。

取值:

none : 默认值。不指定图像

url ( url ) : 使用绝对或相对 url 地址指定图像

● [list-style-position](http://www.w3cschool.cn/cssref/pr-list-style-position.html)

说明:

设置或检索作为对象的列表项标记如何根据文本排列。假如一个列表项目的左外补丁( margin-left )被设置为 0 ，则列表项目标记不会被显示。左外补丁( [margin-left](http://www.w3cschool.cn/cssref/pr-margin-left.html) )最小可以被设置为 30 。仅作用于具有 display 属性值等于 list-item 的对象。如 li 对象。

取值:

outside : 默认值。列表项目标记放置在文本以外，且环绕文本不根据标记对齐

inside : 列表项目标记放置在文本以内，且环绕文本根据标记对齐

● [list-style-type](http://www.w3cschool.cn/cssref/pr-list-style-type.html)

说明:

设置或检索对象的列表项所使用的预设标记。若 [list-style-image](http://www.w3cschool.cn/cssref/pr-list-style-image.html) 属性值为 none 或指定 url 地址的图片不能被显示时，此属性将发生作用。

取值:

disc : 　CSS1　实心圆

circle : 　CSS1　空心圆

square : 　CSS1　实心方块

decimal : 　CSS1　阿拉伯数字

lower-roman : 　CSS1　小写罗马数字

upper-roman : 　CSS1　大写罗马数字

lower-alpha : 　CSS1　小写英文字母

upper-alpha : 　CSS1　大写英文字母

none : 　CSS1　不使用项目符号

armenian : 　CSS2　传统的亚美尼亚数字

cjk-ideographic : 　CSS2　浅白的表意数字

georgian : 　CSS2　传统的乔治数字

lower-greek : 　CSS2　基本的希腊小写字母

hebrew : 　CSS2　传统的希伯莱数字

hiragana : 　CSS2　日文平假名字符

hiragana-iroha : 　CSS2　日文平假名序号

katakana : 　CSS2　日文片假名字符

katakana-iroha : 　CSS2　日文片假名序号

lower-latin : 　CSS2　小写拉丁字母

upper-latin : 　CSS2　大写拉丁字母

使用背景图片改变列表符

list-style-image：url(xxx.jpg);

改变列表符位置

[list-style-position](http://www.w3cschool.cn/cssref/pr-list-style-position.html):

inside列表项目标记放在文本内，且对标记对齐

outsid（默认）列表项目放在标记外，不根据标记对齐

## 注意：我们使用列表是时候往往不使用它的默认样式，需要把默认样式去掉然后再设置我们需要的样式，每一次都这样子非常麻烦，所以网上有一个叫做reset.css文件

它帮我们把一些元素的默认样式去掉，使用起来比较方便，下载地址：[CSS reset. Follow me on the twitters for more tips: https://twitter.com/davidwells (github.com)](https://gist.github.com/DavidWells/18e73022e723037a50d6) **使用的时候通过 <link rel="stylesheet" href="css/reset.css">引入即可**

# 伪类

[CSS](https://developer.mozilla.org/zh-CN/docs/Web/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。例如，伪类 [:hover](https://developer.mozilla.org/zh-CN/docs/Web/CSS/: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。

/\* 用户的指针悬停在其上的任何按钮 \*/

button:hover {

color: blue;

}

伪类由冒号（:）后跟着伪类名称组成（例如，:hover）。函数式伪类还包含一对括号来定义参数（例如，:dir()）。附上了伪类的元素被定义为锚元素（例如，button:hover 中的 button）。

伪类让你可以将样式应用于元素，不仅与文档树内容有关，也与外部因素有关——如与导航历史有关的（例如，[:visited](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:visited)）、与其内容的状态有关的（如某些表单元素上的 [:checked](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:checked)）或者与鼠标位置有关的（如 [:hover](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:hover)，它可以让你知道鼠标是否在一个元素上）。

**备注：** 相较于伪类，[伪元素](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements)可用于设置元素特定部分的样式。

## [元素显示状态伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E5%85%83%E7%B4%A0%E6%98%BE%E7%A4%BA%E7%8A%B6%E6%80%81%E4%BC%AA%E7%B1%BB)

这些伪类可以基于元素的显示状态来选择它们。

[:fullscreen](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:fullscreen)

匹配当前处于全屏模式的元素。

## [输入状态伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E8%BE%93%E5%85%A5%E7%8A%B6%E6%80%81%E4%BC%AA%E7%B1%BB)

这些伪类与表单元素相关，可以基于 HTML 属性或交互前后字段的状态来选择元素。

TBD

## [语言的伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E8%AF%AD%E8%A8%80%E7%9A%84%E4%BC%AA%E7%B1%BB)

这些伪类根据文档语言而反应，可以基于语言或书写方向来选择元素。

[:dir() (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:dir)

方向性伪类基于由文档语言所确定的其方向性来选择元素。

[:lang() (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:lang)

基于其内容语言来选择元素。

## [位置伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E4%BD%8D%E7%BD%AE%E4%BC%AA%E7%B1%BB)

这些伪类与链接以及当前文档中的目标元素有关。

TBD

## [资源状态伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E8%B5%84%E6%BA%90%E7%8A%B6%E6%80%81%E4%BC%AA%E7%B1%BB)

这些伪类应用于能够处于被描述为“正在播放”状态的媒体，例如视频。

TBD

## [时间方面的伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E6%97%B6%E9%97%B4%E6%96%B9%E9%9D%A2%E7%9A%84%E4%BC%AA%E7%B1%BB)

这些伪类适用于查看有计时功能的东西，如 [WebVTT](https://developer.mozilla.org/zh-CN/docs/Web/API/WebVTT_API) 字幕轨。

TBD

## [树结构的伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E6%A0%91%E7%BB%93%E6%9E%84%E7%9A%84%E4%BC%AA%E7%B1%BB)

这些伪类与文档树中的元素位置有关。

[:root](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:root)

表示文档的根元素。在 HTML 中这通常是 <html> 元素。

[:empty](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:empty)

表示除空白字符外没有子元素的元素。

TBD

## [用户行为伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E7%94%A8%E6%88%B7%E8%A1%8C%E4%B8%BA%E4%BC%AA%E7%B1%BB)

这些伪类需要用户进行一些交互才能够应用，例如将鼠标指针悬停在元素上。

TBD

## [函数式伪类](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E5%87%BD%E6%95%B0%E5%BC%8F%E4%BC%AA%E7%B1%BB)

这些伪类接受[可容错选择器列表](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Selector_list#%E5%8F%AF%E5%AE%B9%E9%94%99%E9%80%89%E6%8B%A9%E5%99%A8%E5%88%97%E8%A1%A8)作为参数。

[:is()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:is)

匹配与提供的列表中的任何选择器匹配的任何元素。

[:not()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:not)

表示其参数中未表示的任何元素。

[:where()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:where)

优先级调整伪类匹配与提供的列表中的任何选择器匹配的任何元素，但不添加任何优先级权重。

[:has()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:has)

该关系伪类表示与任何一个与锚定的元素的相对选择器相匹配（如果有的话）的元素。

## [语法](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E8%AF%AD%E6%B3%95)

selector:pseudo-class {

property: value;

}

像常规的类一样，你可以在选择器中将任意数量的伪类链在一起。

## [字母索引](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E5%AD%97%E6%AF%8D%E7%B4%A2%E5%BC%95)

由一组 CSS 规范定义的伪类包括以下内容：

A

* [:active](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:active)
* [:any-link](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:any-link)
* [:autofill](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:autofill)

B

* [:blank](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:blank) 实验性

C

* [:checked](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:checked)
* [:current (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:current) 实验性

D

* [:default](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:default)
* [:defined](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:defined)
* [:dir()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:dir) 实验性
* [:disabled](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:disabled)

E

* [:empty](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:empty)
* [:enabled](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:enabled)

F

* [:first](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:first)
* [:first-child](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:first-child)
* [:first-of-type](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:first-of-type)
* [:fullscreen](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:fullscreen)
* [:future (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:future) 实验性
* [:focus](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:focus)
* [:focus-visible](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:focus-visible)
* [:focus-within](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:focus-within)

H

* [:has()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:has) 实验性
* [:host](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:host)
* [:host()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:host)
* [:host-context()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:host-context) 实验性
* [:hover](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:hover)

I

* [:indeterminate](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:indeterminate)
* [:in-range](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:in-range)
* [:invalid](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:invalid)
* [:is()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:is)

L

* [:lang()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:lang)
* [:last-child](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:last-child)
* [:last-of-type](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:last-of-type)
* [:left](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:left)
* [:link](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:link)
* [:local-link (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:local-link) 实验性

M

* [:modal (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:modal)

N

* [:not()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:not)

## .not(类选择器或者其他选择器)实例

|  |
| --- |
| 有如下 CSS 样式声明：  div.outer p {  color: orange;  }  div:not(.outer) p {  color: blueviolet;  }  Copy to Clipboard  将其应用于以下的 HTML 时：  <div class="outer">  <p>This is in the outer div.</p>  <div class="inner">  <p>This text is in the inner div.</p>  </div>  </div> |
| 效果 |

* [:nth-child()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-child)
* [:nth-col() (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-col) 实验性
* [:nth-last-child()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-last-child)
* [:nth-last-col() (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:nth-last-col) 实验性
* [:nth-last-of-type()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-last-of-type)
* [:nth-of-type()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-of-type)

O

* [:only-child](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:only-child)
* [:only-of-type](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:only-of-type)
* [:optional](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:optional)
* [:out-of-range](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:out-of-range)

P

* [:past (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:past) 实验性
* [:picture-in-picture](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:picture-in-picture)
* [:placeholder-shown](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:placeholder-shown)
* [:paused (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:paused)
* [:playing (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:playing)

R

* [:read-only](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:read-only)
* [:read-write](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:read-write)
* [:required](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:required)
* [:right](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:right)
* [:root](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:root)

S

* [:scope](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:scope)
* :state() 实验性

T

* [:target](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:target)
* [:target-within (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:target-within) 实验性

U

* [:user-invalid (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/:user-invalid) 实验性

V

* [:valid](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:valid)
* [:visited](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:visited)

W

* [:where()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:where)

# 结构选择器

## 在伪类选择器中有一些和位置有关的伪类又叫做结构选择器，

## 1）:first-child, :last-child等

## 实例1

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <style>    /\*结构选择器也伪类选择器的一种\*/      li:first-child{          color:orange;          font-size: 30px;      }      li:last-child{          color:turquoise;          font-size: 24px;      }      </style>    </head>  <body>        <ul>          <li>php</li>          <li>cpp</li>          <li class='three'>java</li>          <li>python</li>          <li>web</li>      </ul>  </body>  </html> |
| 效果 |

## 需要注意的是，如果这个元素不是第一个孩子，即使你设置了:first-child,也不会起作用

## 还可以所有.类名:first-child{}的写法，这个写法笔记灵活，只要是这个类名不管是什么标签，只要是第一个孩子，都会有效果

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <style>       /\* 结构选择器，也是伪类选择器的一种 \*/      .o:first-child{          color:orange;          font-size: 30px;      }      li:last-child{          color:turquoise;          font-size: 24px;      }      </style>    </head>  <body>        <ul>          <!-- <p class='o'>golang</p> -->          <li class='o'>php</li>          <li>cpp</li>          <li class='three'>java</li>          <li>python</li>          <li>web</li>      </ul>  </body>  </html> |

## 效果：

|  |
| --- |
|  |

## 2）[:nth-child(数字)](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-child)，[:nth-last-child(数字)](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-last-child)表示第几个孩子和倒数第几个孩子

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <style>        .child:nth-child(2){          color:turquoise;          font-size: 24px;      }      .child:nth-last-child(2){          border: 1px solid darkviolet;          color: dodgerblue;  width: 500px;      }      </style>    </head>  <body>        <ul>          <li class='child'>php</li>          <li class='child'>.net</li>          <li class='child'>cpp</li>          <li class='child'>java</li>          <li class='child'>python</li>          <li class='child'>web</li>      </ul>  </body>  </html> |
| 效果 |

## 注意：必须是指定的位置和指定的标签或者指定的类才有作用

## 3）[:nth-of-type()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-of-type)和[:nth-last-of-type()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:nth-last-of-type)就是某一种标签或者某个类的第几个和倒数第几个元素

## 实例

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <style>        /\* 按位置和按种类来找第几个的伪类 \*/        h2:nth-last-of-type(2){          color:gold;          font-weight: 800;      }       div:nth-of-type(2){          border: 2px dotted green;          width: 300px;          background-color:purple;      }      p:nth-of-type(2)      {          border: 1px solid darkviolet;          color: dodgerblue;          width: 500px;      }      section:nth-last-of-type(1){          width: 300px;          height: 100px;          border: 1px solid salmon;           background-color: pink;      }      </style>      </head>  <body>        <div>          <h2>我是h2标签</h2>          <section>我是section</section>          <p>我是p标签</p>          <section>我是section</section>          <h2>我是h2标签</h2>          <p>我是p标签</p>          <section>我是section</section>          <p>我是p标签</p>          <p>我是p标签</p>          <h2>我是h2标签</h2>          <section>我是section</section>          <h2>我是h2标签</h2>      </div>  </body>  </html> |
| 效果： |

## 结构选择器还可以这么用：

## p:nth-child(odd) 或者 p:nth-child(2n-1) ->选择奇数的p如第1，3，5，7，9.。。。个，

## p:nth-child(even) 或者 p:nth-child(2n) ->选择偶数的p如第2，4，6，8，。。。个，

## p:nth-child(3n) –》只会找到3的倍数的如第3，6，9.。。个

## p:nth-child(-n+5) –》找到前面5个标签

## 实例

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>   <style>        li:nth-child(-n+5){          width: 300px;          border: 1px dotted purple;          margin: 5px 0;      }    </style>      </head>  <body>        <ul>          <li class='child'>php</li>          <li class='child'>.net</li>          <li class='child'>cpp</li>          <li class='child'>java</li>          <li class='child'>python</li>          <li class='child'>web</li>      </ul>    </body>  </html> |
| 效果、 |

## p:nth-last-child(-n+3) –》找到后面3个标签

## [规范](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E8%A7%84%E8%8C%83)

| **Specification** |
| --- |
| [HTML Standard  # pseudo-classes](https://html.spec.whatwg.org/multipage/#pseudo-classes) |
| [Selectors Level 4](https://drafts.csswg.org/selectors/) |
| [CSS Basic User Interface Module Level 4](https://drafts.csswg.org/css-ui/) |

## [参见](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes#%E5%8F%82%E8%A7%81)

* [伪元素](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements)

### Found a content problem with this page?

* [Edit the page on GitHub](https://github.com/mdn/translated-content/edit/main/files/zh-cn/web/css/pseudo-classes/index.md).
* [Report the content issue](https://github.com/mdn/translated-content/issues/new?template=page-report-zh-cn.yml&mdn-url=https%3A%2F%2Fdeveloper.mozilla.org%2Fzh-CN%2Fdocs%2FWeb%2FCSS%2FPseudo-classes&metadata=%3C%21--+Do+not+make+changes+below+this+line+--%3E%0A%3Cdetails%3E%0A%3Csummary%3EPage+report+details%3C%2Fsummary%3E%0A%0A*+Folder%3A+%60zh-cn%2Fweb%2Fcss%2Fpseudo-classes%60%0A*+MDN+URL%3A+https%3A%2F%2Fdeveloper.mozilla.org%2Fzh-CN%2Fdocs%2FWeb%2FCSS%2FPseudo-classes%0A*+GitHub+URL%3A+https%3A%2F%2Fgithub.com%2Fmdn%2Ftranslated-content%2Fblob%2Fmain%2Ffiles%2Fzh-cn%2Fweb%2Fcss%2Fpseudo-classes%2Findex.md%0A*+Last+commit%3A+https%3A%2F%2Fgithub.com%2Fmdn%2Ftranslated-content%2Fcommit%2F4cfef8dab59ad70eec6583810ae8283598560575%0A*+Document+last+modified%3A+2022-11-04T08%3A47%3A25.000Z%0A%0A%3C%2Fdetails%3E).
* [View the source on GitHub](https://github.com/mdn/translated-content/blob/main/files/zh-cn/web/css/pseudo-classes/index.md?plain=1).

# a标签常用的伪类选择器

  a:link{ /\*未被访问过的时候的样式\*/

            color: ccc;

         }

  a:visited{ /\*被访问过后的样式\*/

            color: fuchsia;

         }

 a:hover{ /\*鼠标悬停在元素上面时的样式，这个最常用\*/

             color: gold;

             text-decoration: underline;

         }

a:active{ /\*表示鼠标按住不放时的样式\*/

            color: green;

         }

a[:focus](https://developer.mozilla.org/zh-CN/docs/Web/CSS/:focus){ /\*表示获得焦点后的样式\*/

 color: lawngreen;

}

## 注意顺序，否则没有效果 :link->:visited->:hover->:active。

## 其实最常用的是:hover,其他可以不用。其他没有什么大用

# 伪元素

伪元素是一个附加至选择器末的关键词，允许你对被选择元素的特定部分修改样式。下例中的 [::first-line](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::first-line) 伪元素可改变段落首行文字的样式。

/\* 每一个 <p> 元素的第一行。 \*/

p::first-line {

color: blue;

text-transform: uppercase;

}

Copy to Clipboard

**备注：** 与伪元素比较，[pseudo-classes](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-classes) 能够根据状态改变元素样式。

## [语法](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements#%E8%AF%AD%E6%B3%95)

selector::pseudo-element {

property: value;

}

Copy to Clipboard

一个选择器中只能使用一个伪元素。伪元素必须紧跟在语句中的简单选择器/基础选择器之后。

**备注：** 按照规范，应该使用双冒号（::）而不是单个冒号（:），以便区分伪类和伪元素。但是，由于旧版本的 W3C 规范并未对此进行特别区分，因此目前绝大多数的浏览器都同时支持使用这两种方式来表示伪元素。

## [字母索引](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements#%E5%AD%97%E6%AF%8D%E7%B4%A2%E5%BC%95)

一组 CSS 规范定义的伪元素包括以下内容：

A

* [::after](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::after)

B

* [::backdrop](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::backdrop)
* [::before](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::before)

C

* [::cue](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::cue)
* [::cue-region (en-US)](https://developer.mozilla.org/en-US/docs/Web/CSS/::cue-region)

F

* [::first-letter](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::first-letter)
* [::first-line](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::first-line)
* [::file-selector-button](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::file-selector-button)

G

* [::grammar-error](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::grammar-error) 实验性

M

* [::marker](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::marker)

P

* [::part()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::part)
* [::placeholder](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::placeholder)

S

* [::selection](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::selection)
* [::slotted()](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::slotted)
* [::spelling-error](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::spelling-error) 实验性

T

* [::target-text](https://developer.mozilla.org/zh-CN/docs/Web/CSS/::target-text) 实验性

## [规范](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements#%E8%A7%84%E8%8C%83)

| **Specification** |
| --- |
| [Cascading Style Sheets Level 2 Revision 2 (CSS 2.2) Specification # pseudo-element-selectors](https://www.w3.org/TR/CSS22/selector.html#pseudo-element-selectors) |

## [浏览器兼容性](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements#%E6%B5%8F%E8%A7%88%E5%99%A8%E5%85%BC%E5%AE%B9%E6%80%A7)

|  |  |  |
| --- | --- | --- |
| **浏览器** | **最低版本** | **支持的伪元素** |
| Internet Explorer | 8.0 | :pseudo-element |
| 9.0 | :pseudo-element ::pseudo-element |
| Firefox (Gecko) | 1.0 (1.0) | :pseudo-element |
| 1.0 (1.5) | :pseudo-element ::pseudo-element |
| Opera | 4.0 | :pseudo-element |
| 7.0 | :pseudo-element ::pseudo-element |
| Safari (WebKit) | 1.0 (85) | :pseudo-element ::pseudo-element |

## [参见](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Pseudo-elements#%E5%8F%82%E8%A7%81)

# 组合选择器

## 1）后代选择器，祖先元素和后代元素用空格隔开，可以把儿子，孙子。。。所有的后代都找到

## 如：ul li { color：red；}此时如果li下面又有ul，这个ul的li的文本颜色也会变红，如此类推，这个很常用

## 实例：

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <!—后代选择器 -->      <style>        .father li{         border: 1px dashed #ccc;         width: 500px;         list-style: none;      }      </style>    </head>  <body>      <ul class="father">          <li class="son">          广东           <ul class="gson">               <li>东莞</li>               <li>广州</li>           </ul>          </li>          <li>广西</li>          <li>湖南</li>          <li>湖北</li>          <li>山东</li>      </ul>  </body>  </html> |
|  |

## 2）儿子选择器，只选择儿子，对儿子的儿子即孙子不起作用，可以继承的顺序除外

## ul>li {font-size:30px;}

## 实例

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <!-- 儿子选择器 -->      <style>        .father > li{         border: 1px dashed #ccc;         width: 500px;         list-style: none;      }      </style>    </head>  <body>      <ul class="father">          <li class="son">          广东           <ul class="gson">               <li>东莞</li>               <li>广州</li>           </ul>          </li>          <li>广西</li>          <li>湖南</li>          <li>湖北</li>          <li>山东</li>      </ul>  </body>  </html> |
|  |

## 3）邻近兄弟选择器，只选择到一个标签

## X + Y{ font-size:30px;}

## 需要注意的是这样子设置后样式对x以后的兄弟y（x后面的紧跟着的第一个元素）才起作用对x前面的兄弟是不起作用的，因为html代码是从上往下执行的。

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>      <style>        /\* 邻近兄弟选择器，只选择它后面最近的一个兄弟 \*/      .gx + li {         border: 1px double deeppink;         width: 500px;         list-style: none;         color: deepskyblue;      }        </style>    </head>  <body>      <ul class="father">          <li class="son">          广东           <ul class="gson">               <li>东莞</li>               <li>广州</li>           </ul>          </li>          <li class="gx">广西</li>          <li>湖南</li>          <li>湖北</li>          <li>山东</li>      </ul>    </body>  </html> |
|  |

## 4）一般兄弟选择器，选择该元素后面的所有兄弟

## x~y{color:deeppink;}

## 如： p~span{ color:deeppink;}会选中p标签后面的所有span标签

## 实例

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>后代选择器，子选择器</title>     <style>      /\* 一般兄弟选择器，选择该元素后面的所有兄弟 \*/      .three ~ li{          width: 300px;          border: 2px dashed lime;          margin-top: 5px;          background-color: yellow;      }      </style>    </head>  <body>        <ul>          <li>php</li>          <li>cpp</li>          <li class='three'>java</li>          <li>python</li>          <li>web</li>      </ul>  </body>  </html> |
|  |

## 5）并列选择器，也叫做选择器分组，用逗号隔开

## a，div，p{ font-size:30px;}

## 上面这几种选择器最常用的是后代选择器，儿子选择器只有特定的场景有用，兄弟选择器几乎不用

# 组合选择器的作用是和:hover伪类组合使用可以用来做二级菜单

## 实例

|  |
| --- |
| <!DOCTYPE html>  <html lang="en">  <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">      <title>利用组合选择器实现二级菜单</title>      <style>          .two{              display: none;              list-style: none;          }          /\* 当鼠标悬停在.one上面时让.two显示,注意写法 \*/          .one:hover .two{              display: block;              list-style: none;          }      </style>  </head>  <body>      <ul>          <li  class="one">cpp              <ul class="two">                  <li>cpp基础</li>                  <li>cpp进阶</li>                  <li>cpp实战</li>              </ul>          </li>          <li>java</li>          <li>python</li>          <li>web</li>      </ul>  </body>  </html> |
| 悬停前悬停中 |

## 注意css的写法，是当.one:hover状态的时候的.two

# 属性选择器

## 语法：[属性名]{样式名称:样式值;}

## 其实就是通过标签的属性来找到标签

## 一个有一些变体

## 如 :

## a[href="https://www.google.com"]{

## color:deeppink;

## }

## 表示只有href属性是https://www.google.com 的a标签才可以有里面的样式

## 还可以进行模糊匹配

## CSS **属性选择器**匹配那些具有特定属性或属性值的元素。

### /\* 存在 title 属性的 <a> 元素 \*/

a[title] {

color: purple;

}

### /\* 存在 href 属性并且属性值匹配"https://example.org"的 <a> 元素 \*/

a[href="https://example.org"] {

color: green;

}

### /\* 存在 href 属性并且属性值包含"example"的 <a> 元素 \*/

a[href\*="example"] {

font-size: 2em;

}

### /\* 存在 href 属性并且属性值结尾是".org"的 <a> 元素 \*/

a[href$=".org"] {

font-style: italic;

}

### /\* 存在 class 属性并且属性值包含单词"logo"的<a>元素 \*/

a[class~="logo"] {

padding: 2px;

}

Copy to Clipboard

## [语法](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Attribute_selectors#%E8%AF%AD%E6%B3%95)

|  |  |  |
| --- | --- | --- |
| [attr] | 常 | 表示带有以 attr 命名的属性的元素。 |
| [attr=value] | 常 | 表示带有以 attr 命名的属性，且属性值为 value 的元素。 |
| [attr~=value] |  | 表示带有以 attr 命名的属性的元素，并且该属性是一个以空格作为分隔的值列表，其中至少有一个值为 value。 |
| [attr|=value] |  | 表示带有以 attr 命名的属性的元素，属性值为“value”或是以“value-”为前缀（- 为连字符，Unicode 编码为 U+002D）开头。典型的应用场景是用来匹配语言简写代码（如 zh-CN、zh-TW 可以用 zh 作为 value）。 |
| [attr^=value] |  | 表示带有以 attr 命名的属性，且属性值是以 value 开头的元素。 |
| [attr$=value] |  | 表示带有以 attr 命名的属性，且属性值是以 value 结尾的元素。 |
| [attr\*=value] | 常 | 表示带有以 attr 命名的属性，且属性值至少包含一个 value 值的元素。 |
| [attr operator value i] |  | 在属性选择器的右方括号前添加一个用空格隔开的字母 i（或 I），可以在匹配属性值时忽略大小写（支持 ASCII 字符范围之内的字母）。 |

#### 上表中有常子的说明是常用的

#### CSS

a {

color: blue;

}

/\* 以 "#" 开头的页面内部链接 \*/

a[href^="#"] {

background-color: gold;

}

/\* 包含 "example" 的链接 \*/

a[href\*="example"] {

background-color: silver;

}

/\* 包含 "insensitive" 的链接，不区分大小写 \*/

a[href\*="insensitive" i] {

color: cyan;

}

/\* 包含 "cAsE" 的链接，区分大小写 \*/

a[href\*="cAsE" s] {

color: pink;

}

/\* 以 ".org" 结尾的链接 \*/

a[href$=".org"] {

color: red;

}

/\* 以 "https" 开始，".org" 结尾的链接 \*/

a[href^="https"][href$=".org"] {

color: green;

}

Copy to Clipboard

#### HTML

<ul>

<li><a href="#internal">内部链接</a></li>

<li><a href="http://example.com">示例链接</a></li>

<li><a href="#InSensitive">非敏感内部链接</a></li>

<li><a href="http://example.org">示例 org 链接</a></li>

<li><a href="https://example.org">示例 https org 链接</a></li>

</ul>

Copy to Clipboard

#### 结果

|  |
| --- |
|  |

### [多语言](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Attribute_selectors#%E5%A4%9A%E8%AF%AD%E8%A8%80)

#### CSS

/\* 将所有包含 `lang` 属性的 <div> 元素的字重设为 bold \*/

div[lang] {

font-weight: bold;

}

/\* 将所有语言为美式英语的 <div> 元素的文本颜色设为蓝色 \*/

div[lang~="en-us"] {

color: blue;

}

/\* 将所有语言为葡萄牙语的 <div> 元素的文本颜色设为绿色 \*/

div[lang="pt"] {

color: green;

}

/\* 将所有语言为中文的 <div> 元素的文本颜色设为红色

无论是简体中文（zh-CN）还是繁体中文（zh-TW） \*/

div[lang|="zh"] {

color: red;

}

/\* 将所有 `data-lang` 属性的值为 "zh-TW" 的 <div> 元素的文本颜色设为紫色 \*/

/\* 备注：和 JS 不同，CSS 可以在不使用双引号的情况下直接使用带连字符的属性名 \*/

div[data-lang="zh-TW"] {

color: purple;

}

Copy to Clipboard

#### HTML

<div lang="en-us en-gb en-au en-nz">Hello World!</div>

<div lang="pt">Olá Mundo!</div>

<div lang="zh-CN">世界您好！</div>

<div lang="zh-TW">世界您好！</div>

<div data-lang="zh-TW">世界您好！</div>

Copy to Clipboard

#### 结果

|  |
| --- |
|  |

### [HTML 有序列表](https://developer.mozilla.org/zh-CN/docs/Web/CSS/Attribute_selectors#html_%E6%9C%89%E5%BA%8F%E5%88%97%E8%A1%A8)

HTML 规范要求 [type](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input#type) 属性被不分大小写地匹配，因为它主要用于 [<input>](https://developer.mozilla.org/zh-CN/docs/Web/HTML/Element/input) 元素中。请注意，如果用户代理不支持这些修饰符，那么选择器将无法匹配。

#### CSS

/\* 根据文档语言不同，大小写敏感性不同 \*/

ol[type="a"] {

list-style-type: lower-alpha;

background: red;

}

ol[type="b" s] {

list-style-type: lower-alpha;

background: lime;

}

ol[type="B" s] {

list-style-type: upper-alpha;

background: grey;

}

ol[type="c" i] {

list-style-type: upper-alpha;

background: green;

}

Copy to Clipboard

#### HTML

<ol type="A">

<li>

非大小写敏感匹配呈现红色背景（类型选择器默认值）

</li>

</ol>

<ol type="b">

<li>如果支持 `s` 修饰符（大小写敏感匹配），拥有亮绿色背景</li>

</ol>

<ol type="B">

<li>如果支持 `s` 修饰符（大小写敏感匹配），拥有灰色背景</li>

</ol>

<ol type="C">

<li>

如果支持 `i` 修饰符（大小写不敏感匹配），拥有绿色背景

</li>

</ol>

Copy to Clipboard

#### 结果

|  |
| --- |
|  |

## 还有一种通配符选择器，\*，不建议使用，会很卡

## \*{color:red;}

# 选择器的权重计算方法

## 1.加法

### id > 同时有几个class在一起的类选择器 > 一个class = 属性选择器 = 伪类选择器 > 标签选择器

### 内嵌样式权重1000

### id权重100

### 类，属性选择器，伪类选择器权重10

### 标签选择器，伪元素选择器权重1

### 通配符\*和继承样式，权重为0

#### 注意：这种算法不满足10进一，也就是10个10是100，不管你有多少给class，他们的权重都没有一个id高，所以这种算法比较难理解

## 2.4个0法

### 内嵌样式权重(1，0，0，0)

### id权重100(0,1,0,0)

### 类，属性选择器，伪类选择器权重10(0,0,1,0)

### 标签选择器，伪元素选择器权重1(0,0,0,1)

### 通配符\*和继承样式，权重为0(0,0,0,0)

#### 用这种方法比较权重从从左往右依次比较一旦有结果就马上停止

## !important

!important是特殊的存在，!important 是不在 css 选择器的权重计算范围内的, 而它之所以能让 css 选择器生效是因为浏览器在遇见 !important 时会进行特殊的判断。当多个 !important 需要进行比较时, 会先计算其权重再进行比较。通常来说, 不提倡使用 !important。不过当我们需要覆盖style属性中定义的样式时，可以选择使用!important。

## 一般不建议使用，只有在特色的情况下使用

## 语法,如 p { color:pink !important; }