你知道 CSS 选择器有哪些

CSS 世界其实是很庞大的，在我们前端中的分量虽说没有 JavaScript 大，但是也是不可或缺的。所以对于 CSS 的态度，大家也要端正，这样才更有利于写出更好的布局以及交互。

今天我们先说说 CSS 选择器。对于选择器，相信同学们都有所了解。在日常的学习中，我们多少都会用到。除了大家常用的选择器外还有其他选择器么，接下来给大家总结一下。

选择器的分类

根据选择器的基本形式，选择器被分为 5 类：

1. **元素选择符**
2. **关系选择符**
3. **属性选择符**
4. **伪元素选择符**
5. **伪类选择符**

接下来我们一次来看一看这五类选择器都有哪些，该如何使用？

元素选择符

元素选择符包括：**通配符选择器（*）、标签选择器（element）、id 选择器（#）、class 选择器（.）。**

1. 通配符选择器（*）
   可以选择所有元素的选择器。web 设计者经常用它将页面中所有元素的 margin 和 padding 设置为 0。 *选择符也可以在子选择器中使用。兼容浏览器：IE6+、Firefox、Chrome、Safari、Opera

   代码示例：
   选择所有元素

```
* {
    margin: 0;
    padding: 0;
  }
```

因为他的作用域太大，相当耗浏览器资源。所以要尽可能的减少使用

2. 标签选择器（element）
   标签选择器也被称为元素选择器。使用标签选择器作用于作用域范围内的所有对应标签。兼容浏览器：IE6+、Firefox、Chrome、Safari、Opera

   代码示例：
   选择，a，ul 元素

```
a {
    color: red;
  }
ul {
    margin-left: 0;
    }
```

3. Id 选择器（#）
   id 选择器指定具有 ID 的元素的样式。Id 选择器是我们最常用的 css 选择器之一。id 选择器的优势是精准，高优先级(优先级基数为 0100，高于 class 的 0010)，作为 javascript 脚本钩子的不二选择，同样缺点也很明显优先级过高，重用性差（只能有一个 Id）。兼容浏览器：IE6+、Firefox、Chrome、Safari、Opera

   代码示例：
   选择 id 为 container 的元素

```
#container {
        width: 960px;
        margin: auto;
}
```

4. Class 选择器（.）
   class 选择器是指定类的所有元素的样式。class 选择器与 id 选择器的不同是 class 选择器能作用于期望样式化的一组元素。兼容浏览器：IE6+、Firefox、Chrome、Safari、Opera

   代码示例：
   选择 class 为 error 的元素

```
.error {
        color: red;
 }
```

以上四种是大家最常见，也是最常使用的选择器。接下来我们来看一看关系选择器。

关系选择符

关系选择符包括：后代选择器（E F）、直接子元素选择器（E > F）、相邻选择器（E + F）、兄弟选择器（E ~ F）、并列选择器（E, F）

1. 后代选择器（E F）
   这也是我们最常用的一种选择器——后代选择器。用于选取 E 元素下子元素 F，要留意的点是，这种方式的选择器将选取其下所有匹配的子元素，无视层级，所以有的情况是不宜使用的，比如代码去掉 li 下的所有 a 的下划线，但 li 里面还有个 ul，我们不希望 ul 下的 li 的 a 去掉下划线。使用此后代选择器的时候要考虑是否希望某样式对所有子孙元素都起作用。这种后代选择器还有个作用，就是创建类似命名空间的作用。兼容浏览器：IE6+、Firefox、Chrome、Safari、Opera
   代码示例：
   选择 li 下面的 a 标签，不限层级

```
li a {
        text-decoration: none;
    }
```

2. 直接子元素选择器（E>F）
   子选择器。与后代选择器 X Y 不同的是，子选择器只对 X 下的直接子级 Y 起作用，理论上来讲 X > Y 是值得提倡选择器，作用范围明确。兼容浏览器：IE7+、Firefox、Chrome、Safari、Opera。

   代码示例：
   选择 id 为 container 下面直接子元素 ul

```
#container>ul {
        border: 1px solid black;
    }
```

3. 相邻选择器（E+F）
   相邻选择器用于选取第一个指定的元素(E)之后(不是内部)紧跟的元素(F)。
   在下面的结构中：

```
<div>
    <span>123</span>
</div>
<p>ppp</p>
<button>btn</button>
```

在这样的结构中，可以通过相邻选择器，选择 p 标签， div + p 通过 div 找到近邻 div 的兄弟标签 P 标签。但是 div +button 就不能被选择，因为不是近邻的兄弟元素。

代码示例：
选择 div 近邻的 p 标签

```
div+p {
        color: red;
    }
```

4. 兄弟选择器（E~F）

E~F 选择器，也称兄弟选择器，匹配出现在 E 后面的 F。注意这里的 E 和 F 这两种元素必须具有相同的父元素，但 F 不必紧跟在 E 的后面。

在下面的结构中：

```
<div>
    <span>123</span>
</div>
<span>1</span>
<p>p1</p>
<p>p2</p>
```

我们利用 div ~ p 可以选择与 div 同级的所有 p 元素，可以不连续，可以不挨着，但是必须同级。

代码示例：

与 div 同级的所有 p 标签选中

```
div~p {
        color: red;
    }
```

5. 并列选择器（E, F）
并列选择器，用于多个元素共享同一样式。利于样式代码的复用。和层级没有关系，任意多个元素可以通过并列选择器进行选择。

代码示例:

```
div, p {
        color: red;
    }
```

属性选择符

属性选择符根据元素的属性进行选择，元素的属性包括但不限于（id, class, data-, href,src,）等等，自定义属性也是可以的。权重值为 0010。对于属性选择器加上了部分正则的匹配，可以选择更多情况下的元素，接下来我们看看。

1. [attr] 基本的属性选择器
   [attr] 选择器用于选取带有指定属性的元素。这里的 attr，可以是我们知道一些属性，id class 之类的，也可以是我们自定义的属性。
   代码示例：

```
/* 选择具有 id 属性的元素 */
  [id] {
      color: red;
  }
  /* 选择具有 class 属性的元素 */
  [class] {
      color: green;
  }
  /* 选择具有 src 属性的元素 */
  [src] {
      color: blue
  }
```

2. 属性选择器的加强版
   a. [attr=value]。属性选择器，选择元素具有 attr 属性并且值为 value。
   代码示例：

```
  /* 选择具有 class 属性为 content 的元素 */
  [class="content"] {
      color: blue;
  }
```

   b. [attr~=value]。属性选择器。用于选择属性值包含一个指定单词的元素。属性选择器中的波浪线符号可以让我们匹配属性值中用空格分隔的多个值中的一个。
   代码示例：

```
  /* 选择具有 class 属性中有 wrap 的元素，注意不是包含 wrapper 里面就不可以， wrap host 这样的类名结构是可以的 */
   [class~="wrap"] {
     color: blue;
   }
```

   c. [attr^=value]。选择器匹配元素属性值带指定的值开始的元素。类似于正则中的规则，匹配属性 attr 的值以 value 开始的元素。
   代码示例：

```
/*选择具有 href 属性并且，href 属性以 http 开头的元素。*/
 [href^="http"] {
   color: red;
 }
```

d. [attr$=value]。选择器匹配元素属性值带指定的值结尾的元素。同样类似于正则中的规则，匹配属性 attr 的值以 value 结尾的元素。
代码示例：

```
/*选择具有 href 属性并且，href 属性以.com 结尾的元素。*/
[href$=".com"] {
     color: red;
 }
```

e. [attr*=value] 选择器匹配元素属性值包含指定值的元素。匹配 attr 属性中包含 value 的的元素。

```
/*选择具有 href 属性并且，href 属性中存在 baidu 的元素。*/
[href*="baidu"] {
         color: red;
}
```

f. [attr|=value] 选择器用于选择以指定值开头的属性值的元素。注意：该值是整个单词，空格逗号分开的不可以，中划线分开的可以算为整个单词。
代码示例：

```
/*选择具有 class 属性并且 class 属性中以 my 开头的元素。并且 my 作为整个单词*/
[class|="my"] {
   color: red;
 }
 <div class="my-content">淘宝</div>
```

伪类选择符

伪类选择符的内容就很多了。比如大家之前接触的 hover。接下来我们将能一起打出“组合拳”的选择器放到一起来说。

1. :link, :hover, :visited, :active 选择器。

:link 向未访问的链接添加特殊的样式。注意: :link 选择器对已经访问的链接没有样式。

:visited 向访问过的链接添加特殊的样式。
:hover 在鼠标移到链接上时添加的特殊样式。
:active 向活动的链接添加特殊的样式。当你点击一个链接时它变成活动链接。当被点击时触发。
:link 选择器设置了未访问过的页面链接样式， :visited 选择器设置访问过的页面链接的样式， :hover 选择器当有鼠标悬停在其上的链接样式。:active 选

择器设置当你点击链接时的样式。
注意：为了产生预期的效果，在 CSS 定义中，:hover 必须位于 :link 和 :visited 之后
代码示例：

```
a:link {
    color: red;
}
a:hover {
    color: gray;
}
a:active {
    color: green;
}
a:visited {
    color: yellow;
}
```

这里要注意，上面四个除了 hover 以外的伪类选择器，仅仅针对 a 标签去使用。Hover 可以对任意元素使用。

第一个表示已经被点击（触发 visited），第二个可以被点击（触发 link，默认也是 link 样式），第三个被点击时触发（active）

2. :first-child, :last-child, :only-child, :nth-child(), :nth-last-child()选择器
上面这些选择器用的不多，但是大家要了解。这五个选择器，先注重位置，在看类型。兼容浏览器：IE9+、Firefox、Chrome、Safari。接下来我们挨个看一看该如何使用？该注意什么？
   a. :first-child
   :first-child 选择器用于选取属于其父元素的首个子元素的指定选择器。
   这里要说明的是：
   p:first-child 这个选择器的含义是：**p 是它父级元素下面的第一个元素。**并不是 p 元素下面的第一个元素。还要注意：**这里的不仅强调第一个出现，而且必须是第一个元素。**
   代码示例：
   Html:

```
<p>body 下面的第 1 个 p 元素</p>
<p>body 下面的第 2 个 p 元素</p>
<div>
    <p>div 下面的第 1 个 p 元素</p>
</div>
```

```
<div>
    <span>第一个 span 元素</span>
    <p>第一个 p 元素</p>
</div>

CSS:
 p:first-child {
 background-color: lime;
}
```

结果：

body下面的第1个p元素

body下面的第2个p元素

div下面的第1个p元素

第一个span元素

第一个p元素

上面例子中的最后下面的 p 是第一次出现，但是不是父级下面的第一个元素。

b. :last-child

:last-child 选择器用来匹配父元素中最后一个子元素。与 first-child 的方式的一致，但是强调最后一个。

代码示例：

同样是上面的 html 结构。我们把 CSS 改一下

```
p:last-child {
  background-color: lime;
}
```

结果：

body下面的第1个p元素

body下面的第2个p元素

div下面的第1个p元素

第一个span元素

第一个p元素

这里强调的是：p 元素作为父级元素的直接子元素，并且是最后一个元素才可以被选中。

c. :only-child

:only-child 选择器匹配属于父元素中唯一子元素的元素。

p:only-child 表示 父元素中有且只有一个元素，并且为 p 时选中。
代码示例：

HTML：
```
<p>body 下面的第 1 个 p 元素</p>
 <div>
  <p>div 下面的第 1 个 p 元素</p>
 </div>
 <div>
        <span>div2 第一个 span 元素</span>
        <p>div2 第一个 p 元素</p>
 </div>
```
CSS：
```
p:only-child {
  background-color: lime;
 }
```
结果：

body下面的第1个p元素

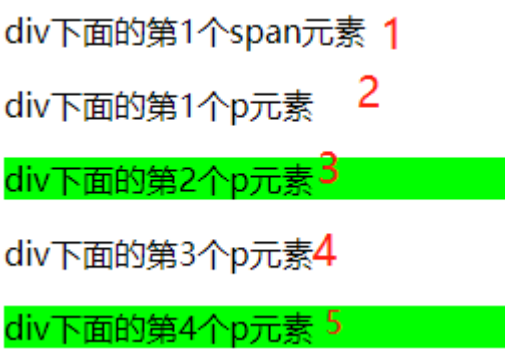
div下面的第1个p元素

div2第一个span元素

div2第一个p元素

d. :nth-child()
nth-child(n) 选择器匹配父元素中的第 n 个子元素 n 可以是一个数字，一个关键字（odd, even），或者一个公式(2n + 1)。这里的 n 从 1 开始计数。该选择器匹配同类型中的第 n 个同级兄弟元素。:nth-child(1) 等同于：first-child 。
代码示例：
HTML：
```
<div>
        <span>div 下面的第 1 个 span 元素</span>
        <p>div 下面的第 1 个 p 元素</p>
        <p>div 下面的第 2 个 p 元素</p>
        <p>div 下面的第 3 个 p 元素</p>
        <p>div 下面的第 4 个 p 元素</p>
 </div>
```
CSS：
```
p:nth-child(2n-1){
  background-color: lime;
 }
```

结果：



我们的公式是 2n-1 和 odd 是一样的，意思是选择当前子元素中的奇数个 P 元素，这里的奇数是在所有子元素中奇数，所以就出现了图上的选择情况。

e. :nth-last-child()

:nth-last-of-type(n)选择器匹配同类型中的倒数第 n 个同级兄弟元素。与上面的 nth-child 用法一致，只不过 nth-last-child 是倒数。同样 n 可以是具体数字，可以是公式，也可以是一个关键字。

3. :first-of-type, :last-of-type, :only-of-type, :nth-of-type(), :nth-last-of-type()

接下来的这五个伪元素选择器，和上面五个是对应的。但是，比上面那 5 个更常用，更好用。所以同学们要掌握。这一类选择器，先看类型在看位置，更容易去理解。兼容浏览器：IE9+、Firefox、Chrome、Safari。

a. :first-of-type

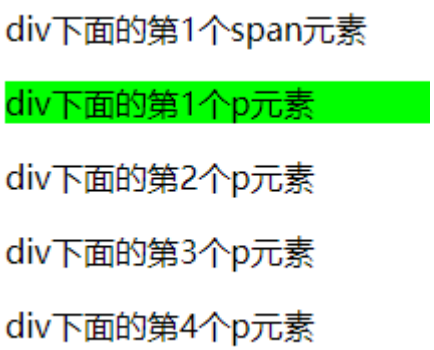
:first-of-type 选择器匹配元素其父级是特定类型的第一个子元素。概念感觉差不多。其实差别是很大的。这一系列的选择器，多了一个 type，这个 type 就很能说明问题。这里面强调的是，该类型的子元素的第一个，跟有多少子元素无关，只要是这个类型并且第一次出现就可以被选中。

代码示例：

CSS：

```
p:first-of-type{
 background-color: lime;
}
```

结果：



在这里 p 元素不是第一个直接子元素，但是当前子元素中第一个出现的 p 就可以。

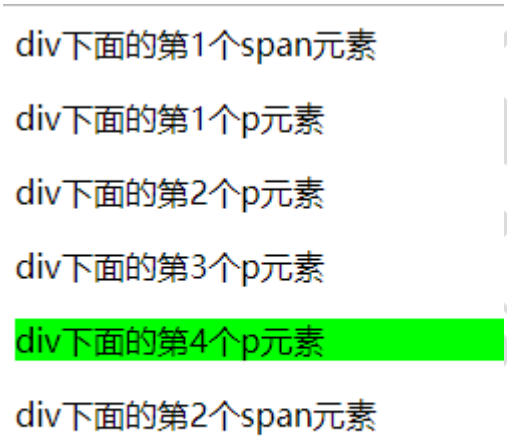
b. :last-of-type

:last-of-type 选择器匹配元素其父级是特定类型的最后一个子元素。同样，只要求最后出现就可以，不需要是最后一个子元素。

代码示例：

```
p:last-of-type{
 background-color: lime;
}
```

结果：



这个里面 p 是不最后一个元素，但是最后的 p 元素就可。

c. :only-of-type

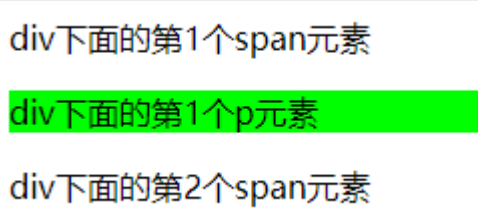
:only-of-type 选择器匹配属于同类型中唯一同级元素。父级元素中的子元素可以有多个，但是这个类型的元素只有一个就可以被选中。

代码示例：

CSS：

```
p:only-of-type{
 background-color: lime;
}
```

结果：



Div 下面有多个元素，但是 p 元素只有一个所以可以通过:only-of-type 选择出来。

d. :nth-of-type()

:nth-of-type(n)选择器匹配同类型中的第 n 个同级兄弟元素。同样 n 可以是一个数字，一个关键字，或者一个公式。:nth-of-type(1) 与 first-of-type

一样。

代码示例：

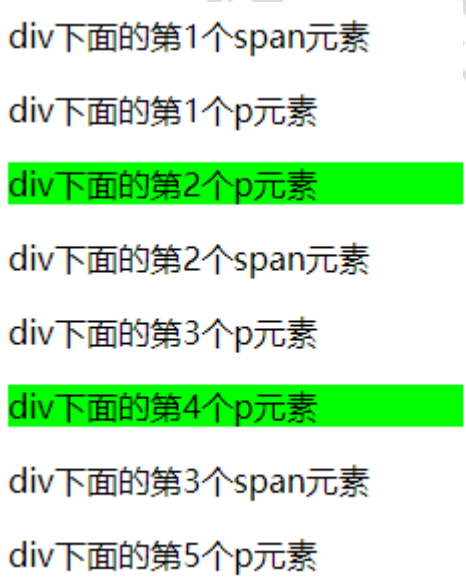
HTML:

```
<div>
 <span>div 下面的第 1 个 span 元素</span>
 <p>div 下面的第 1 个 p 元素</p>
 <p>div 下面的第 2 个 p 元素</p>
 <span>div 下面的第 2 个 span 元素</span>
 <p>div 下面的第 3 个 p 元素</p>
 <p>div 下面的第 4 个 p 元素</p>
 <span>div 下面的第 3 个 span 元素</span>
 <p>div 下面的第 5 个 p 元素</p>
</div>
```

CSS:

```
p:nth-of-type(2n){
 background-color: lime;
}
```

结果：



在这里，我们可以看到，只看 p 元素，然后选择偶数个 p 元素，和 p 元素在总子元素中的位置无关。

e. :nth-last-of-type()

:nth-last-of-type(n)选择器匹配同类型中的倒数第 n 个同级兄弟元素。与:nth-of-type() 选择器行为一致，不过是从后往前看，参数类型也是一致的。:nth-last-of-type(1) 与 last-of-type 是一样的。

4. 其他伪类选择器。

a. :not

:not(selector) 选择器匹配每个元素是不是指定的元素/选择器。这个 not 选

择器还是很常用。当我们只想在若干元素中筛选出来几个不满足条件的元素时，:not()选择器具有奇效。兼容浏览器：IE6+、Firefox、Chrome、Safari
代码示例：
HTML:

```
<p class="content">1</p>
<p class="content" id="p">2</p>
<p class="content">3</p>
<p class="content">4</p>
<p class="content">5</p>
<p class="content" id="p">6</p>
<p class="content">7</p>
<p class="content">8</p>
<p class="content">9</p>
<p class="content">10</p>
```

CSS:

```
p:not([id]) {
  background: lime;
}
```

结果

我们上面的操作是选择没有 id 属性的 p 元素。配合着之前的属性选择器。

b. :empty
:empty 选择器选择每个没有任何子级的元素（包括文本节点，空格也是文本节点）。但是注释节点可以有。
代码示例：

CSS：

```
p:empty {
 background: lime;
 height: 100px;
}
```

HTML:

```
<p><!-- demo --></p>
```

结果：

c. :checked

:checked 选择器匹配每个选中的输入元素(仅适用于单选按钮或复选框)。被选中时出现样式

代码示例：

```
input[type="checkbox"]:checked {
 width: 100px;
 height: 100px;
}
```

d. :root

:root 选择器用匹配文档的根元素。在 HTML 中根元素始终是 HTML 元素。但是在其他文档中就是其他文档的根元素。

代码示例：

: root {} 不需要其他元素。

e. :enabled，:disabled

:enabled 选择器匹配每个启用的的元素，:disabled 选择器匹配每个禁用的的元素（主要用于表单元素， 配合着 disabled 属性使用）。

代码示例：

HTML：

```
<input type="text">
<br>
<input type="text" disabled>
```

CSS：

```
input[type="text"]:enabled {
     background: #f20;
}
input[type="text"]:disabled {
     background: gray;
}
```

结果：

以上给大家展示了常用的伪元素选择器，总体看来还是很多的。但是有些需要了解即可，有些需要认真掌握。接下来看最后一部分伪类选择器。

伪类选择器

伪类选择器包含六个：`::first-letter`,`::first-line`, `::before`, `::after`, `::placeholder`, `::selection` 这六个选择器。这里面的 after 和 before 大家之前接触过，但是还是有一些是没有接触过的，我们来看看。

a. ::first-letter, ::fist-line

::first-letter 选择器用来指定元素第一个字母的样式。:first-line 选择器用来指定选择器第一行的样式。这两个选择器有一些限制，这个选择器仅适用于块级元素中。对于可操作样式也是有限制的。 可操作的属性有： font, color, Background, margin, padding, border, text-decoration, text-transform, ine-height, float, clear。

代码示例：

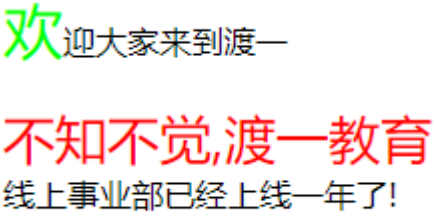
CSS：

```
p::first-letter {
    font-size: 30px;
    color: lime;
}
div::first-line {
    font-size: 26px;
    color: red;
}
```

HTML：

```
<p>欢迎大家来到渡一</p>
<div>不知不觉,渡一教育线上事业部已经上线一年了!</div>
```

结果：



b. ::before, ::after

这两个伪类与 content 结合用于在元素的前面或者后面追加内容。通常我们用于追加内容， 清除浮动。这个就不多说了。

c. ::placeholder, ::selection

这两个选择器，我们并不常用，而且还有很大的兼容性问题，很多浏览器并没有很好的支持。这里说给大家，大家作为了解内容即可。

::placeholder 伪元素用于控制表单输入框占位符的外观，它允许开发者/设计师改变文字占位符的样式，默认的文字占位符为浅灰色。

代码示例：

CSS：

```
input::-webkit-input-placeholder {
    color: #f20;
}
input:-ms-input-placeholder { /* IE10+ */
    color: #f20;
}
input:-moz-placeholder { /* Firefox4-18 */
    color: #f20;
}
input::-moz-placeholder { /* Firefox19+ */
    color: #f20;
}
```

HTML：

```
<input type="text" placeholder="占位符"/>
```

结果：

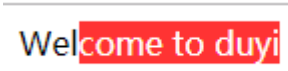
::selection 选择器匹配元素中被用户选中或处于高亮状态的部分。

::selection 只可以::selection 选择器应用于少数的 CSS 属性：color, background, cursor,和 outline

CSS：

```
::selection {
    color: white;
   background: red;
}
```

结果：



关于选择器的内容终于整理完了，这个过程仅仅是一个开始，只有我们在项目中

不断的去使用，去发现这些选择器的优势所在，最终达到熟练掌握，烂熟于心。这样你就成为真正的高手了。