CSS选择器

本词条由"科普中国"科学百科词条编写与应用工作项目 审核 。

要使用css对HTML页面中的元素实现一对一，一对多或者多对一的控制，这就需要用到CSS选择器。

HTML页面中的元素就是通过CSS选择器进行控制的。

目录

简介

什么是选择器呢?

每一条css样式定义由两部分组成，形式如下： [code] 选择器{样式} [/code] 在{}之前的部分就是"选择器"。 "选择器"指明了{}中的"样式"的作用对象，也就是"样式"作用于网页中的哪些元素

1.1 类别选择器

类选择器根据类名来选择

前面以"."来标志，如:

.demoDiv{

color:#FF0000;

}

在HTML中，元素可以定义一个class的属性。

如:

这个区域字体颜色为红色

同时，我们可以再定义一个元素:

这个段落字体颜色为红色

最后，用浏览器浏览，我们可以发现所有class为demoDiv的元素都应用了这个样式。包括了页面中的div元素和p元素。

上例我们给两个元素都定义了class，但如果有很多个元素都会应用这个元素，那得一个个的定义元素，就会造成页面重复的代码太多，这种现象称为"多类症"。

我们可以改成这样来定义。

这个区域字体颜色为红色

同时，我们可以再定义一个元素:

这个段落字体颜色为红色

这样，我们就只是定义了一个类，同时把样式应用到了所有的元素当中。

1.2 标签选择器

播报

一个完整的HTML页面是有很多不同的标签组成，而标签选择器，则是决定哪些标签

采用相应的CSS样式，(在大环境中你可能处于不同的位置，但是不管怎么样，你总

是穿着同一套衣服，这件衣服就是由标签选择器事先给你限定好的，不管走到哪里

都是这身衣服)比如，在style.css文件中对p标签样式的声明如下：

p{

font-size:12px;

background:#900;

color:090;

}

复制代码则页面中所有p标签的背景都是#900(红色)，文字大小均是12px，颜色为#090(绿色)

，这在后期维护中，如果想改变整个网站中p标签背景的颜色，只需要修改

background属性就可以了，就这么容易！

1.3 ID选择器

播报

ID 选择器可以为标有特定 ID 的 HTML 元素指定特定的样式。 根据元素ID来选择元素，具有唯一性，这意味着同一id在同一文档页面中只能出现一次，例如，你将一个元素的id取值为”navi”，那么在同一页面你就不能再将其他元素id取名为”navi”了。尽管你会发现即便你把几个元素都命名成相同的id名字，css选择器还是会把这些元素都选中应用样式（如class选择器那样），对于css选择器，id属性的唯一性似乎不存在。然而，对于js而言，它只会选择具有相同id名字元素中的第一个。出于一个好的编程习惯，同一个id不要在页面中出现第二次。

前面以”#”号来标志，在样式里面可以这样定义：

#demoDiv{

color:#FF0000;

}

这里代表id为demoDiv的元素的设置它的字体颜色为红色。

我们在页面上定义一个元素把它的ID定义为demoDiv，如：

这个区域字体颜色为红色

用浏览器浏览，我们可以看到因为区域内的颜色变成了红色

再定义一个区域

这个区域没有定义颜色

用浏览器浏览，与预期的一样，区域没有应用样式，所以区域中的字体颜色还是默认的颜色黑色。

1.4 后代选择器

播报

后代选择器也称为包含选择器，用来选择特定元素或元素组的后代，将对父元素的选择放在前面，对子元素的选择放在后面，中间加一个空格分开。后代选择器中的元素不仅仅只能有两个，对于多层祖先后代关系，可以有多个空格加以分开，如id为a、b、c的三个元素，则后代选择器可以写成#a #b #c{}的形式，只要对祖先元素的选择在后代元素之前、中间以空格分开即可。

如：

<style>

.father.child{

color:#0000CC;

}

</style>

黑色

<label class="child">蓝色

也是蓝色

</label>

这里我们定义了所有class属性为father的元素下面的class属性为child的颜色为蓝色。后代选择器是一种很有用的选择器，使用后代选择器可以更加精确的定位元素。

1.5 子选择器

播报

请注意这个选择器与后代选择器的区别，子选择器（child selector）仅是指它的直接后代，或者你可以理解为作用于子元素的第一个后代。而后代选择器是作用于所有子后代元素。后代选择器通过空格来进行选择，而子选择器是通过“>”进行选择，我们看下面的代码：

Example Source Code

CSS：

#links a {color:red;}

#links > a {color:blue;}

HTML：

Div+CSS教程>

CSS布局实例

CSS2.0教程

我们将会看到第一个链接元素“Div+CSS教程”会显示成蓝色，而其它两个元素会显示成红色。当然，或许你的浏览器并不支持这样的CSS选择符。我们在一开始也强调了不太兼容的现状。

子选择器（>）和后代选择器（空格）的区别：都表示“祖先-后代”的关系，但是>必须是“爸爸>儿子”，而空格不仅可以是“爸爸儿子”，还能是“爷爷儿”、“太爷爷儿子”。

1.6 伪类选择器

播报

有时候还会需要用文档以外的其他条件来应用元素的样式，比如鼠标悬停等。这时候我们就需要用到伪类了。以下是链接应用的伪类定义。

a:link{

color:#999999;

}

a:visited{

color:#FFFF00;

}

a:hover{

color:#006600;

}

/* IE不支持，用Firefox浏览可以看到效果 */

input:focus{

background:# E0F1F5;

}

Link表示链接在没有被点击时的样式。Visited表示链接已经被访问时的样式。Hover表示当鼠标悬停在链接上面时的样式。

伪类不仅可以应用在链接标签中，也可以应用在一些表单元素中，但表单元素的应用IE不支持，所以一般伪类都只会被应用在链接的样式上。

1.7 通用选择器

播报

通用选择器用*来表示。例如：

*{

font-size: 12px;

}

表示所有的元素的字体大小都是12px；同时通用选择器还可以和后代选择器组合。

例如：

p *{

……

}

表示所有p元素后代的所有元素都应用这个样式。但是与后代选择器的搭配容易出现浏览器不能解析的情况，比如像这样子：

所有的文本都被定义成红色

所有这个段落里面的子标签都会被定义成蓝色

所有这个段落里面的子标签都会被定义成蓝色

所有这个段落里面的子标签都会被定义成蓝色

所有这个段落里面的子标签都会被定义成蓝色

这个例子里面p标签里面嵌套了一个p标签，这个时候样式可能会出现与预期结果不相同的结果。

1.8 群组选择器

播报

当几个元素样式属性一样时，可以共同调用一个声明，元素之间用逗号分隔。如：

p, td, li {

line-height:20px;

color:#c00;

}

#main p, #sider span {

color:#000;

line-height:26px;

}

.#main p span {

color:#f60;

}

.text1 h1,#sider h3,.art_title h2 {

font-weight:100;

}

使用群组选择器，将会大大的简化CSS代码，将具有多个相同属性的元素，合并群组进行选择，定义同样的CSS属性，这大大的提高了编码效率，同时也减少了CSS文件的体积。

1.9 相邻同胞选择器

播报

我们除了上面的子选择器与后代选择器，我们可能还希望找到兄弟两个当中的一个，如一个标题h1元素后面紧跟了两个段落p元素，我们想定位第一个段落p元素，对它应用样式。我们就可以使用相邻同胞选择器。看下面的代码：

Example Source Code CSS

h1 + p {color:blue}

HTML

<h1>一个非常专业的CSS站点</h1>

Div+CSS教程中，介绍了很多关于CSS网页布局的知识。

CSS布局实例中，有很多与CSS布局有关的案例。

我们将会看到第一个段落“Div+CSS教程中，介绍了很多关于CSS网页布局的知识。”文字颜色将会是蓝色。而第二段则不受此CSS样式的影响。

+和~的区别：类似上面一个，两者都表示兄弟关系，但是+必须是“大哥+二哥”，~还能是“大哥~三弟”、“二哥~四妹”

1.10 属性选择器

播报

您可以用判断html标签的某个属性是否存在的方法来定义css

属性选择器，是根据元素的属性来匹配的，其属性可以是标准属性也可以是自定义属性；!ie6，0 0 1 0

当然，也可以同时匹配多个属性；

[attr]

[title] {margin-left: 10px}

//选择具有 title 属性的所有元素；

[attr=val]

[title = 'this'] {margin-right: 10px}

//选择属性 title 的值等于 this 的所有元素

[attr^=val]

[title ^= 'this'] {margin-left: 15px}

//选择属性title的值以this开头的所有元素

[attr$=val]

[title $= 'this'] {margin-right: 15px}

//选择属性title的值以this结尾的所有元素

[attr*=val]

[title *= 'this'] {margin: 10px}

//选择属性title 的值包含 this 的所有元素

[attr~=val]

[title ~= 'this'] {margin-top: 10px}

//选择属性 title 的值包含一个以空格分隔的词为 this 的所有元素，即 title 的值里必须要有 this 这个单词并且this要与其他单词之间有空格分隔

[attr|=val]

[title |= 'this'] {margin-bottom: 10px}

//选择属性 title 的值等于this，或值以 this- 开头的所有元素

1.11 伪元素选择器

播报

所有伪元素选择器都必须放在出现该伪元素的选择器的最后面，也就是说伪元素选择器不能跟任何派生选择器，如：p:first-letter em {} 这就是不合法的，ie6不支持

:first-letter，设置块元素首字母样式，行内元素转换成块元素和行内块元素也支持；

div p:first-letter {font-size: 20px}

//选择div元素里所有的p元素的第一个字母或汉字，如果把块元素转换成行内元素则就不支持了；

:first-line，设置第一个文本行样式；

.box .main:first-line {color: #f00}

//只有部分属性允许first-line：所有font属性、color、所有background属性、word-spacing、letter-spacing、text-decoration、vertical-align、text-transform、line-height

:before，设置之前的样式，可以插入生成的内容，并设置其样式；

body:before {content: 'The Start:'; display: block}

//在body元素前插入文本内容'The Start:'，并设置其为块元素

:after，设置之后的样式，可以插入生成的内容，并设置其样式；

body:after {content: 'The End.'; display: block}

//在body元素最后插入文本内容'The End.'，并设置其为块元素

1.12 结构性伪类选择器

播报

HTML CODE:

1.

2. First span

3. First p

4. First divStrong text

5. Second p

6. Second div Second spanThird span

7.

结构性伪类选择器的冒号前边可以跟一个其他选择器做为限定；

带括号的选择器，里面一定要有参数；

匹配子元素，同时也会匹配孙子元素，因为子元素是孙子元素的父元素；

下面的 !lte8是指IE8一下浏览器不支持，包括IE8也不支持

:first-of-type，选择相对父元素里同类型子元素中的第一个，!lte8

.box :first-of-type {color: #f00}

//匹配2.3.4以及4里面的strong和6里面的第一个span，因为这个span是6里的第一个span子元素

.box .ft:first-of-type {color: #ff0}

//匹配3和4里面的strong，因为3是box里面的第一个p且class=”ft”，而4里只有一个strong且class=”ft”，而5和6虽然class=”ft”但是他们相对于box的同类型中不是第一个出现的；

:last-of-type，选择相对父元素里同类型子元素中的最后一个，!lte8

.box :last-of-type {color: #f00}

//匹配2.5.6以及4里的strong和6里的最后一个span

:only-of-type，选择相对父元素里同类型子元素中只有一个的元素，!lte8

.box :only-of-type {color: #f00}

//匹配2以及4里的strong，类为box里同类型元素只有一个的只有span

.box .ft:only-of-type {color: #f00}

//只匹配4里的strong

:only-child，选择的元素相对于其父元素是唯一的子元素，!lte8

.box :only-child {color: #f00}

//只匹配4里的strong

:nth-child(n)，选择其父元素的第n个子元素或多个子元素，索引从1开始，当n用于表达式时索引从0开始!lte8

.box :nth-child(3) {color: #f00}

//匹配第三个子元素即这里的4

.box :nth-child(odd) {color: #f00} 等价于 .box :nth-child(2n + 1) {color: #f00}

//匹配奇数即这里的2.4.6以及4里的strong和6里的第一个span

.box :nth-child(even) {color: #f00} 等价于 .box :nth-child(2n + 2) {color: #f00}和.box :nth-child(2n)

//匹配偶数即这里的3.5以及6里的第二个span

.box :nth-child(n + 1) {color: #f00}

//匹配 n + 1开始的所有子元素即.box里所有的子元素以及子孙元素，因为这里n是从1开始的即：

n = 0 ----> n + 1 = 0 + 1 = 1，即这里的2

n = 1 ----> n + 1 = 1 + 1 = 2，即这里的3

... ...

n = 4 ----> n + 1 = 4 + 1 = 5，即这里的6

:nth-last-child(n)，跟:nth-child(n)使用类似，只是索引是从最后开始往前数，!lte8

.box :nth-last-child(3) {color: #f00}

//匹配倒数第三个子元素即这里的4

:nth-of-type(n)，选择父元素的第n个或多个同类型的子元素，!lte8

.box :nth-of-type(2) {color: #f00}

//匹配5和6以及6里面的第二个span

:nth-last-of-type(n)，同上，只是从最后开始往前数，!lte8

.box :nth-last-of-type(2) {color: #f00}

//匹配3和4以及6里面的第一个span

:first-child，选择父元素里的第一个子元素，!ie6

.box :first-child {color: #f00}

//匹配2和4里的strong以及6里的第一个span

:last-child，选择父元素里的最后一个子元素，!lte8

.box :last-child {color: #f00}

//匹配6和6里的最后一个span以及4里的strong

:root，选择文档的根元素，在HTML中就是指<html>标签，!Ie8

:empty，选择没有任何内容的元素，那怕是有一个空格也不行，!Ie8

table td:empty {background-color: #ffc}

//匹配表格里没有内容的td

:target，选择当前活动的元素，指锚点点击后跳转到的那一个元素，!Ie8

:not(selector)，选择排除selector以外的其他所有元素，!Ie8

.box *:not(div) {background-color: #ffc}

//选择box里除div以外的所有后代元素，如果div里有其他非div元素，也会选择上，如上的HTML CODE就会选择上div里面的span和strong

1.13 UI元素状态伪类选择器

播报

:enabled，指定元素处于可用状态时的样式，一般用于input，select和textarea

:disabled，指定元素处于不可用状态时的样式，一般用于input，select和textarea

:read-only，指定元素为只读状态时的样式，FF为-moz-read-only，一般用于input和textarea

:read-write，指定元素为只可写状态时的样式，FF为-moz-read-write，一般用于input和textarea

:checked，指定元素被选中状态时的样式，FF为-moz-checked一般用于checkbox和radio

:default，指定元素默认选中的样式，一般用于checkbox和radio

:indeterminate，指定默认一组单选或复选都没被选中的样式，只要有一个被选中则样式被取消，一般用于checkbox和radio

::selection，指定元素处理选中状态时的样式，可用于所有元素，上面的几个基本上只用于表单元素；!Ie8

FF为::-moz-selection，不能用群组选择器来写；

::selection {background-color: #ffc; color: #fff}

::-moz-selection {background-color: #ffc; color: #fff}

注意事项

播报

由于对CSS的解释是自上而下的，对于一个元素的相同属性描述，放在下面的会覆盖掉位于上面的属性描述，因此我们在对元素的选择中一定要注意书写顺序，如：

a:visited {color: #00FF00; text-decoration: none}

a:hover {color: #FF00FF; text-decoration: underline}

采用这样的书写顺序，无论链接有没有被访问过，只要当鼠标移到链接上，链接都会变成蓝色并有下划线。但是，如果采用下面的书写顺序：

a:hover {color: #FF00FF; text-decoration: underline}

a:visited {color: #00FF00; text-decoration: none}

如果链接被访问过，则当你鼠标移到链接上时不会变成蓝色并有下划线，依然保持绿色。

[1]

参考资料

1. W3school，css3选择器参考手册 ．w3school[引用日期2014-03-20]