

CSS语法：除了属性和选择器，你还需要知道这些带@的规则

winter 2019-02-07



你好，我是 winter。

今天我们进入 CSS 的学习。CSS 是前端工程师几乎每天都要用的技术了，不过 CSS 的学习资料却是最糟糕的，这是因为 CSS 并没有像 HTML 和 JavaScript 那样的一份标准文档。

如果我们到 W3C 的网站上搜索看看，可以得到一些信息：

[🔗 https://www.w3.org/TR/?title=css](https://www.w3.org/TR/?title=css)

在这里，我们一共看到了 98 份 CSS 相关的标准，它们各自从一些角度规定了 CSS 的特性。

这里我们暂且去掉 Working Draft 状态的标准，可以得到 22 份候选标准和 6 份推荐标准。

既然我们的专栏内容强调去系统性学习 CSS，于是，面对这 22+6 份标准，我们就又需要一条线索，才能把这些离散的标准组织成易于理解和记忆的形式。

在这样的需求下，我找到的线索就是 CSS 语法，任何 CSS 的特性都必须通过一定的语法结构表达出来，所以语法可以帮助我们发现大多数 CSS 特性。

CSS 语法的最新标准，你可以戳这里查看：

🔗 <https://www.w3.org/TR/css-syntax-3/>

这篇文档的阅读体验其实是非常糟糕的，它对 CSS 语法的描述使用了类似 LL 语法分析的伪代码，而且没有描述任何具体的规则。

这里你就不必自己去阅读了，我来把其中一些有用的关键信息抽取出来描述一下，我们一起来看看。

我们拿到这份标准可以看到，去除空格、HTML 注释等无效信息，**CSS 的顶层样式表由两种规则组成的规则列表构成，一种被称为 at-rule，也就是 at 规则，另一种是 qualified rule，也就是普通规则。**

at-rule 由一个 @ 关键字和后续的一个区块组成，如果没有区块，则以分号结束。这些 at-rule 在开发中使用机会远远小于普通的规则，所以它的大部分内容，你可能会感觉很陌生。

这些 at 规则正是掌握 CSS 的一些高级特性所必须的内容。qualified rule 则是指普通的 CSS 规则，也就是我们所熟识的，由选择器和属性指定构成的规则。

at 规则

好了，现在我们已经知道了，CSS 语法的整体结构，接下来我们要做的是个体力活，从所有的 CSS 标准里找到所有可能的 at-rule（不用谢，我已经帮你找好了，如果页面定位不准，你可以打开页面搜索关键字）。

@charset : 🔗 <https://www.w3.org/TR/css-syntax-3/>

@import : 🔗 <https://www.w3.org/TR/css-cascade-4/>

@media : <https://www.w3.org/TR/css3-conditional/>

@page : <https://www.w3.org/TR/css-page-3/>

@counter-style : <https://www.w3.org/TR/css-counter-styles-3>

@keyframes : <https://www.w3.org/TR/css-animations-1/>

@fontface : <https://www.w3.org/TR/css-fonts-3/>

@supports : <https://www.w3.org/TR/css3-conditional/>


@namespace : <https://www.w3.org/TR/css-namespaces-3/>

这里的每一种 @规则背后，都是一组 CSS 的知识。在我们的课程中，有些会重点介绍，不过，为了先给你建立起一个整体的认知，我们这里会给所有的 @规则提供一些简单的例子和介绍。

@charset

@charset 用于提示 CSS 文件使用的字符编码方式，它如果被使用，必须出现在最前面。这个规则只在给出语法解析阶段前使用，并不影响页面上的展示效果。

```
1 @charset "utf-8";
```

 复制代码


@import

@import 用于引入一个 CSS 文件，除了 @charset 规则不会被引入，@import 可以引入另一个文件的全部内容。

```
1 @import "mystyle.css";  
2 @import url("mystyle.css");
```

 复制代码


```
1 @import [ <url> | <string> ]  
2       [ supports( [ <supports-condition> | <declaration> ] ) ]?  
3       <media-query-list>? ;
```

 复制代码

通过代码，我们可以看出，import 还支持 supports 和 media query 形式。

@media


media 就是大名鼎鼎的 media query 使用的规则了，它能够对设备的类型进行一些判断。在 media 的区块内，是普通规则列表。

 复制代码

```
1 @media print {  
2     body { font-size: 10pt }  
3 }
```

@page


page 用于分页媒体访问网页时的表现设置，页面是一种特殊的盒模型结构，除了页面本身，还可以设置它周围的盒。

 复制代码

```
1 @page {  
2     size: 8.5in 11in;  
3     margin: 10%;  
4  
5     @top-left {  
6         content: "Hamlet";  
7     }  
8     @top-right {  
9         content: "Page " counter(page);  
10    }  
11 }
```

@ counter-style


counter-style 产生一种数据，用于定义列表项的表现。

 复制代码

```
1 @counter-style triangle {  
2     system: cyclic;  
3     symbols: ▶;  
4     suffix: " ";  
5 }
```

@ key-frames


keyframes 产生一种数据，用于定义动画关键帧。

 复制代码

```
1 @keyframes diagonal-slide {  
2  
3   from {  
4     left: 0;  
5     top: 0;  
6   }  
7  
8   to {  
9     left: 100px;  
10    top: 100px;  
11  }  
12  
13 }
```

@ fontface

fontface 用于定义一种字体，icon font 技术就是利用这个特性来实现的。

 复制代码

```
1 @font-face {  
2   font-family: Gentium;  
3   src: url(http://example.com/fonts/Gentium.woff);  
4 }  
5  
6 p { font-family: Gentium, serif; }
```

@ support

support 检查环境的特性，它与 media 比较类似。

@ namespace

用于跟 XML 命名空间配合的一个规则，表示内部的 CSS 选择器全都带上特定命名空间。

@ viewport

用于设置视口的一些特性，不过兼容性目前不是很好，多数时候被 HTML 的 meta 代替。

其它

除了以上这些，还有些目前不太推荐使用的 at 规则。

@color-profile 是 SVG1.0 引入的 CSS 特性，但是实现状况不怎么好。

@document 还没讨论清楚，被推迟到了 CSS4 中。

@font-feature-values 。

普通规则

接下来我们进入 qualified rule，也就是普通规则的部分，看看这里有什么需要我们记住的内容。

qualified rule 主要是由选择器和声明区块构成。声明区块又由属性和值构成。我在下面的列表中，介绍了这部分语法的组成要点。

普通规则

- 选择器
- 声明列表

属性

值

值的类型

函数

选择器

我们先来看看选择器，它有一份独立的标准，我们可以参考这个网址：

🔗 <https://www.w3.org/TR/selectors-4/>

这份标准不在我们前面的过滤条件中，它属于 CSS 和 HTML 共用的标准。

关于选择器的叠加规则等知识我们后文会专门的一节课程来讲，这里我们就从语法的角度介绍一下选择器。

在选择器标准的最后，附有一张选择器的语法表，从这份语法表，我们可以理清楚记忆选择器的思路。

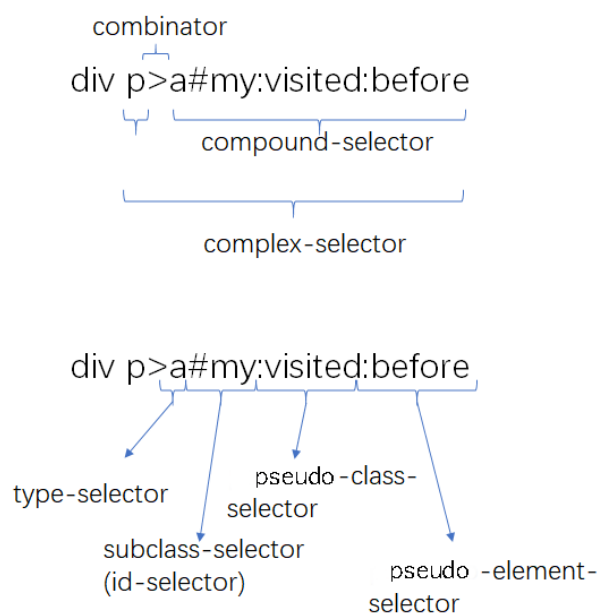
我们从语法结构可以看出，任何选择器，都是由几个符号结构连接的：空格、大于号、加号、波浪线、双竖线，这里需要注意一下，空格，即为后代选择器的优先级较低。

然后对每一个选择器来说，如果它不是伪元素的话，由几个可选的部分组成，标签类型选择器，id、class、属性和伪类，它们中只要出现一个，就构成了选择器。

如果它是伪元素，则在这个结构之后追加伪元素。只有伪类可以出现在伪元素之后。我在下面用一个列表（不太严谨地）整理了选择器的语法结构：

- complex-selector
 - combinator
 - 空格
 - >
 - +
 - ~
 - ||
 - compound-selector
 - type-selector
 - subclass-selector
 - id
 - class
 - attribute
 - pseudo-class
 - pseudo-element

我们在这里可以参考一个示例图：



(语法结构分析示例)

看完了选择器，我们继续来看看声明部分的语法。

声明：属性和值

声明部分是一个由“属性: 值”组成的序列。

属性是由中划线、下划线、字母等组成的标识符，CSS 还支持使用反斜杠转义。我们需要注意的：属性不允许使用连续的两个中划线开头，这样的属性会被认为是 CSS 变量。

在 [CSS Variables 标准](#) 中，以双中划线开头的属性被当作变量，与之配合的则是 `var` 函数：

```
1 :root {
2   --main-color: #06c;
3   --accent-color: #006;
4 }
5 /* The rest of the CSS file */
6 #foo h1 {
7   color: var(--main-color);
8 }
```

[复制代码](#)

值的部分，主要 [在标准 CSS Values and Unit](#)，根据每个 CSS 属性可以取到不同的值，这里的值可能是字符串、标识符。

CSS 属性值可能是以下类型。

CSS 范围的关键字：initial，unset，inherit，任何属性都可以的关键字。

字符串：比如 content 属性。

URL：使用 url() 函数的 URL 值。

整数 / 实数：比如 flex 属性。

维度：单位的整数 / 实数，比如 width 属性。

百分比：大部分维度都支持。

颜色：比如 background-color 属性。

图片：比如 background-image 属性。

2D 位置：比如 background-position 属性。

函数：来自函数的值，比如 transform 属性。

这里我们要重点介绍一下函数。一些属性会要求产生函数类型的值，比如 easing-function 会要求 cubic-bezier() 函数的值：

CSS 支持一批特定的计算型函数：

calc()

max()

min()

clamp()

toggle()

attr()

calc() 函数是基本的表达式计算，它支持加减乘除四则运算。在针对维度进行计算时，calc() 函数允许不同单位混合运算，这非常的有用。

例如：

```
1 section {  
2   float: left;  
3   margin: 1em; border: solid 1px;  
4   width: calc(100%/3 - 2*1em - 2*1px);  
5 }
```

max()、**min()** 和 **clamp()** 则是一些比较大小的函数，**max()** 表示取两数中较大的一个，**min()** 表示取两数之中较小的一个，**clamp()** 则是给一个值限定一个范围，超出范围外则使用范围的最大或者最小值。

toggle() 函数在规则选中多于一个元素时生效，它会在几个值之间来回切换，比如我们要让一个列表项的样式圆点和方点间隔出现，可以使用下面代码：

```
1 ul { list-style-type: toggle(circle, square); }
```

attr() 函数允许 CSS 接受属性值的控制。

总结

在这一部分，我们介绍了 CSS 语法的总体结构，CSS 的语法总体结构是由两种规则列表构成，一种是 at 规则，另一种是普通规则。

在 at 规则中，我举了 13 个以上的例子，并逐个进行了简单的介绍。而在普通规则的部分，我介绍了选择器和声明区块是普通规则的主要组成部分。

并且，我给出了一个（不太严谨）的选择器语法结构，声明区块则由属性和值构成，这一部分我们重点介绍了函数。

从整体上去掌握内容，再去定位到单个细节，这对于我们学习 CSS 有非常重要的提示作用。

最后，给你留一个思考问题，CSS 的函数有很多，本文也提到了不少，请你也一起查阅资料，试着总结一下，你能找到多少种 CSS 函数？

精选留言(33)



CC

在网站上搜索了一下，发现 css 函数有不少，尤其是近三年，增加的函数几乎超过过去的总和。

按照 winter 老师提到「知识完备性」的思路，尝试整理了一下 CSS 函数。

按照功能，分成以下 5 个类别（可能并不完全准确）：

1. 图片

* filter

- * blur()
- * brightness()
- * contrast()
- * drop-shadow()
- * grayscale()
- * hue_rotate()
- * invert()
- * opacity()
- * saturate()
- * sepia()

* cross-fade()

* element()

* image-set()

* imagefunction()

2. 图形绘制

* conic-gradient()

* linear-gradient()

* radial-gradient()

* repeating-linear-gradient()

* repeating-radial-gradient()

* shape()

3. 布局

* calc()

- * clamp()
- * fit-content()
- * max()
- * min()
- * minmax()
- * repeat()

4. 变形/动画

- * transform
 - * matrix()
 - * matrix3d()
 - * perspective()
 - * rotate()
 - * rotate3d()
 - * rotateX()
 - * rotateY()
 - * rotateZ()
 - * scale()
 - * scale3d()
 - * scaleX()
 - * scaleY()
 - * scaleZ()
 - * skew()
 - * skewX()
 - * skewY()
 - * translate()
 - * translate3d()
 - * translateX()
 - * translateY()
 - * translateZ()

5. 环境与元素

- * var()
- * env()
- * attr()

2019-02-07

 2

 272



Sevens 些粉

推荐一下《css世界》这本书，有理论基础也有实战应用和常遇坑，看了两章感觉不错。

2019-02-08

 1

 58



米斯特菠萝

我看到winter老师讲解这些冷门的知识，忽然意识到什么叫做精通？要精通就要抠这种细节，这样才能做到精通

做就要做精通，前端是一种手艺人

2019-02-09



29



文全

@import 用于引入一个 CSS 文件，除了 @charset 规则不会被引入，@import 可以引入另一个 JavaScript 文件的全部内容。这段写错了 应该是css 文件全部内容

2019-02-07



14



mimof9

试了一下 toggle这个函数 并没有效果。clac实测下来是有效果的。

2019-02-08



3

8



Aaaaaaaaaaayou

“只有伪类可以出现在伪元素之后”是不是写反了

2019-02-07



7



无痕

“||”这个选择器我怎么没搜到，是什么意思

2019-03-04



4



前端男孩

本来想发张思维导图的，但是貌似不支持发图，想想算了，也就整理了67个css函数，也不知道够不够。

2019-06-04



2



花骨朵

css函数：<https://www.w3cplus.com/css/css-functions.html>

2019-02-28



2



Rushan-Chen

@mimof9

文章的链接是CSS4 working draft状态的文档，是很新的文档。

看了下CSS3 Candidate Recommendation状态的文档，没有toggle()、min()、max()、clamp()，这几个函数应该是css4新加的，基本上浏览器都还不支持。

attr()虽然css3文档有，查了下，浏览器也都不支持。😂

等浏览器支持估计还要一段时间吧，现在先知道有这个东西就好，我是这样想的。

2019-02-13



2



南蓝

@ counter-style 只在火狐上有用

```
<style>
  @counter-style circled-alpha {
    system: fixed;
    symbols: "\7532""\4E59""\4E19""\4E01""\620A""\5DF1""\5E9A""\8F9B""\58EC""\7678";
    /* 甲 乙 丙 丁 戊 己 庚 辛 壬 癸 */
    suffix: " ";
  }

  ul li {
    list-style: circled-alpha;
  }
</style>
</head>

<body>
  <ul>
    <li>1</li>
```

```
<li>2</li>
<li>3</li>
<li>4</li>
<li>5</li>
</ul>
</body>
```

2019-02-12



2



饼

不少函数还没法用，目前还是var clac比较接近实用了

2019-06-05



1



王峰口

选择器还有一种情况就是*通配符，它不属于标签选择器，也没有优先级

2019-04-17



1



CSS Functions
Alphabetical list of CSS functions included in CSS3.

- attr()
- blur()
- brightness()
- calc()
- circle()
- contrast()
- counter()
- counters()
- cubic-bezier()
- drop-shadow()
- ellipse()
- grayscale()
- hsl()
- hsla()
- hue-rotate()
- hwb()
- image()

inset()
invert()
linear-gradient()
matrix()
matrix3d()
opacity()
perspective()
polygon()
radial-gradient()
repeating-linear-gradient()
repeating-radial-gradient()
rgb()
rgba()
rotate()
rotate3d()
rotateX()
rotateY()
rotateZ()
saturate()
sepia()
scale()
scale3d()
scaleX()
scaleY()
scaleZ()
skew()
skewX()
skewY()
symbols()
translate()
translate3d()
translateX()
translateY()
translateZ()
url()

2019-03-04



吴前端

toggle()函数试了下在google 火狐打开都没用呢显示无效属性值

作者回复: 移动端问题不大, 已经都是webkit了。
不过我还真没注意火狐没这个函数。

2019-02-19



胡永

Pseudo class selector

2019-02-12



hhk

css语法: at 规则 + 普通规则

普通规则: 选择器 + 声明区块

另外, margin 的读音好像读错了

2019-02-07



起而行

发现对于css体系还不太了解, 比如动画, 伪选择器等等。打算在这几天准备一下绘图, 布局, 伪选择器的知识, 为面试做准备

2020-03-19



小夜

@namespace svg url(http://www.w3.org/2000/svg)

请问这个 声明这个 namespace 的必要性在哪里, 在什么情况下会需要使用这个规则

作者回复: 这是选svg元素的时候的简写啊, 跟html的namespace对应的

2019-10-25



Jackzhoumine

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。

2019-09-14

