

浏览器：一个浏览器是如何工作的（阶段三）

winter 2019-02-14



你好，我是 winter。

在上一节课中，我已经讲了浏览器的 DOM 构建过程，但是这个构建的 DOM，实际上信息是不全的，它只有节点和属性，不包含任何的样式信息。

我们这一节课就来讲讲：浏览器是如何把 CSS 规则应用到节点上，并给这棵朴素的 DOM 树添加上 CSS 属性的。

整体过程

首先我们还是要感性地理解一下这个过程。

首先 CSS 选择器这个名称，可能会给你带来一定的误解，觉得好像 CSS 规则是 DOM 树构建好了以后，再进行选择并给它添加样式的。实际上，这个过程并不是这样的。

我们回忆一下我们在浏览器第一课讲的内容，浏览器会尽量流式处理整个过程。我们上一节课构建 DOM 的过程是：从父到子，从先到后，一个一个节点构造，并且挂载到 DOM

树上的，那么在这个过程中，我们是否能同步把 CSS 属性计算出来呢？

答案是肯定的。

在这个过程中，我们依次拿到上一步构造好的元素，去检查它匹配到了哪些规则，再根据规则的优先级，做覆盖和调整。所以，从这个角度看，所谓的选择器，应该被理解成“匹配器”才更合适。

我在 CSS 语法部分，已经总结了选择器的各种符号，这里再把它列出来，我们回顾一下。

空格: 后代，选中它的子节点和所有子节点的后代节点。

>: 子代，选中它的子节点。

+ : 直接后继选择器，选中它的下一个相邻节点。

~: 后继，选中它之后所有的相邻节点。

||: 列，选中表格中的一列。

关于选择器的知识，我会在 CSS 的部分继续讲解。这里我们主要介绍浏览器是如何实现这些规则的。

不知道你有没有发现，这里的选择器有个特点，那就是选择器的出现顺序，必定跟构建 DOM 树的顺序一致。这是一个 CSS 设计的原则，即保证选择器在 DOM 树构建到当前节点时，已经可以准确判断是否匹配，不需要后续节点信息。

也就是说，未来也不可能会出现“父元素选择器”这种东西，因为父元素选择器要求根据当前节点的子节点，来判断当前节点是否被选中，而父节点会先于子节点构建。

理解了 CSS 构建的大概过程，我们下面来看看具体的操作。


首先，我们必须把 CSS 规则做一下处理。作为一门语言，CSS 需要先经过词法分析和语法分析，变成计算机能够理解的结构。

这部分具体的做法属于编译原理的内容，这里就不做赘述了。我们这里假设 CSS 已经被解析成了一棵可用的抽象语法树。

我们在之前的 CSS 课程中已经介绍过 compound-selector 的概念，一个 compound-selector 是检查一个元素的规则，而一个复合型选择器，则是由数个 compound-selector 通过前面讲的符号连接起来的。

后代选择器“空格”

我们先来分析一下后代选择器，我们来一起看一个例子：


 复制代码

```
1 a#b .cls {
2     width: 100px;
3 }
```

可以把一个 CSS 选择器按照 compound-selector 来拆成数段，每当满足一段条件的时候，就前进一段。

比如，在上面的例子中，当我们找到了匹配 a#b 的元素时，我们才会开始检查它所有的子代是否匹配 .cls。

除了前进一段的情况，我们还需要处理后退的情况，比如，我们这样一段代码：

 复制代码

```
1 <a id=b>
2     <span>1<span>
3     <span class=cls>2<span>
4 </a>
5 <span class=cls>3<span>
```

当遇到 时，必须使得规则 a#b .cls 回退一步，这样第三个 span 才不会被选中。后代选择器的作用范围是父节点的所有子节点，因此规则是在匹配到本标签的结束标签时回退。

后继选择器“~”

接下来我们看下后继选择器，跟后代选择器不同的地方是，后继选择器只作用于一层，我们来看一个例子：

```
1 .cls~* {  
2     border:solid 1px green;  
3 }  
4 <div>  
5 <span>1<span>  
6 <span class=cls>2<span>  
7 <span>  
8     3  
9     <span>4</span>  
10 <span>  
11 <span>5</span>  
12 </div>
```

这里 .cls 选中了 span 2 然后 span 3 是它的后继，但是 span 3 的子节点 span 4 并不应该被选中，而 span 5 也是它的后继，因此应该被选中。

按照 DOM 树的构造顺序，4 在 3 和 5 中间，我们就没有办法像前面讲的后代选择器一样通过激活或者关闭规则来实现匹配。

但是这里有个非常方便的思路，就是给选择器的激活，带上一个条件：父元素。

注意，这里后继选择器，当前半段的 .cls 匹配成功时，后续 * 所匹配的所有元素的父元素都已经确定了（后继节点和当前节点父元素相同是充分必要条件）。在我们的例子中，那个 div 就是后继节点的父元素。

子代选择器“>”

我们继续看，子代选择器是如何实现的。

实际上，有了前面讲的父元素这个约束思路，我们很容易实现子代选择器。区别仅仅是拿当前节点作为父元素，还是拿当前节点的父元素作为父元素。

```
1 div>.cls {  
2     border:solid 1px green;  
3 }  
4 <div>  
5 <span>1<span>  
6 <span class=cls>2<span>  
7 <span>  
8     3  
9     <span>4</span>
```

```
10 <span>  
11 <span>5</span>  
12 </div>
```

我们看这段代码，当 DOM 树构造到 div 时，匹配了 CSS 规则的第一段，因为是子代选择器，我们激活后面的 .cls 选择条件，并且指定父元素必须是当前 div。于是后续的构建 DOM 树构建过程中，span 2 就被选中了。

直接后继选择器“+”

直接后继选择器的实现是上述中最为简单的了，因为它只对唯一一个元素生效，所以不需要像前面几种一样反复激活和关闭规则。

一个最简单的思路是，我们可以把它当作检查元素自身的选择器来处理。即我们把 #id+.cls 都当做检查某一个元素的选择器。

另外的一种思路是：给后继选择器加上一个 flag，使它匹配一次后失效。你可以尝试一下，告诉我结果。

列选择器“||”


列选择器比较特别，它是专门针对表格的选择器，跟表格的模型建立相关，我们这里不详细讲了。

其它

我们不要忘记，CSS 选择器还支持逗号分隔，表示“或”的关系。这里最简单的实现是把逗号视为两条规则的一种简易写法。

比如：

```
1 a#b, .cls {  
2  
3 }
```

 复制代码

我们当作两条规则来处理：

[📄 复制代码](#)

```
1  a#b {  
2  
3  }
```

[📄 复制代码](#)

```
1  .cls {  
2  
3  }
```

还有一个情况，就是选择器可能有重合，这样，我们可以使用树形结构来进行一些合并，来提高效率：

[📄 复制代码](#)

```
1  #a .cls {  
2  
3  }  
4  
5  #a span {  
6  
7  }  
8  #a>span {  
9  
10 }
```

这里实际上可以把选择器构造成一棵树：

#a

- < 空格 >.cls
- < 空格 >span
- >span

需要注意的是，这里的树，必须要带上连接符。

结语

这一节我们讲解了 CSS 计算的过程。CSS 计算是把 CSS 规则应用到 DOM 树上，为 DOM 结构添加显示相关属性的过程。在这一节中，我们主要介绍了选择器的几种复合结构

应该如何实现。

在这一步骤之后，我们得到了一棵带有 CSS 属性的树，为我们后续打下了基础。

最后留一个问题，你认为 CSS 语法解析成什么结构，最适合我们进行 CSS 计算。



精选留言(34)



coma

之前选择器是从右往左匹配，好像跟这里的说法不一致，是理解错了吗？

2019-02-14

2

70



以勒

css的渲染过程：

1. 流式渲染，每生成一个dom节点，便立刻去匹配相应的css规则
2. dom节点的生成顺序是 从父-> 子。 css的渲染顺序也是 从 父节点-> 子节点
- 3.混合选择器 在激活的时候父元素已经确定好了 。 后代选择器，所有元素都是父元素，并 选中自己的子元素。 子元素 选择器，只有父元素为当前节点的 dom 元素会被选中。 在父元素 结束标签时，关闭选择器。

2019-02-14

2

20



啊柴

老师好，文中后代选择器是说先检查父级，匹配后再检查子级，以前学习看到比较多的一种说法是从关键选择器开始匹配，然后左移查找选择器的祖先元素，实践中也一直把最后一个选择器权重加高。请问这是从开始就学错了，还是我对两种说法理解有偏差呢？

作者回复: 浏览器编写本身有不同的思路，但是按关键选择器这个思路，据我所知还没有实现。

2019-02-14



17



南半边翅膀

dom树构建与cssom构建有先后关系吗？css计算与dom树流式构建同步进行是不是意味着dom树流式构建之前cssom已经构建完成呢？

2019-02-14



15



一步

老师，对于这个问题，我也有疑问：
dom树构建与cssom构建有先后关系吗？css计算与dom树流式构建同步进行是不是意味着dom树流式构建之前cssom已经构建完成呢？

作者回复: 我这里说的确实有点歧义，cssom是有rule部分和view部分的，rule部分是在dom开始之前就构件完成的，而view部分是跟着dom同步构建的。

2019-03-01



8



flow

请问DOM去匹配css rule的时候是不是必须先等页面的css都下载完成后才会去匹配的？否则如果DOM匹配的时候对应的css还没下载完，后面怎么得到正确的css？

作者回复: head中的css是要下载完的，body中放CSS的话，会重新计算。

2019-02-16



2

8





嗯喊我杰哥

<也就是说，未来也不可能会出现“父元素选择器”这种东西，因为父元素选择器要求根据当前节点的子节点，来判断当前节点是否被选中，而父节点会先于子节点构建。>

这句话后半句好难理解，有大神能通俗地解释一下吗

作者回复: 意思是DOM树构造到父节点的时候，还不知道它有没有子节点，所以算不出来CSS。

2019-02-14



7



乃乎

CSS 没有父选择器那里讲得太好了，这个原因不能更合理

作者回复: 其实Selector Level 4里面已经打破这个规则了，现在还是draft，也不知道最后能不能过。

要是真过了 webkit这块差不多得重写……

2019-02-26



5



flow

看到老师一个分享会的视频，说到DOM匹配css rule的时候是从右往左匹配的，为什么跟文章提到的后代选择器的匹配顺序相反呢？

作者回复: 哈哈，是这样的没错，但是你看，新加了这么多运算符，所以思路也会有不一样。

其实浏览器实现是比较懒的，直接每条规则都检查了，没做合并优化。

2019-02-16



4



CC

第一反应猜测 CSS 语法应该解析成对象（Object），然后根据 DOM 的树形结构，CSS 也会生成自己的树形结构。

查找验证后发现，这被称为 CSSOM (CSS Object Model) 。

举个 CSSOM 的例子：

```
body {font-size: 16px;}
h1 {font-size: 2rem;}
.orange {color: orange;}
div {margin: 1rem 0;}
div p {padding-bottom: 1rem;}
```

从根结点 body 开始，树形结构如下：

```
* body {font-size: 16px;}
  * h1 {font-size: 2rem;}
  * .orange {color: orange;}
  * div {margin: 1rem 0;}
    * div p {padding-bottom: 1rem;}
```

不知道这样理解是否准确？

作者回复: 哈哈 CSSOM不是这样子的，CSSOM主要是DOM结构上的盒的描述，它基本上是依附于DOM树的。

你说的这个东西，比较接近的是CSS的语法树。

2019-02-14



4



我发现关于CSS选择从右到左开始匹配的理论网上还是有很多相关资料的，<https://www.sitepoint.com/optimizing-css-id-selectors-and-other-myths/>

看起来应该还是有点道理？不过这个东西浏览器引擎应该也会不断优化，不同版本不同引擎都可能存在差异

2019-03-07



3



Young!

老师，请问一下您说的「后退，前进」的含义，不太理解

a#b.cls 这个选择器中，分成 a#b 和 .cls，当 DOM 树构造到 这个节点时，css 匹配 a#b，

并且「前进」看是否自带是否有 class="cls" 的元素，这里可能会有指针这个抽象，指向 .cls，当 DOM 树构建到 这里时，css 选择器指针「后退」到 a#b 这里，然后继续构造 DOM 树，而且这个选择器已经被构造所谓的 css 树之类的，只是指针会有回退和前进的时候

可以这样理解吗

2019-03-13



2



Dylan-Tseng

请问老师，有一点不是很明白，如果是在构建dom树的同时，就开始构建css，带空格选择子元素的时候，css怎么知道dom树有没有构建完成，是否有没有子元素，是怎么做匹配的？

作者回复: 并不管有没有子元素，只是让这条规则前进一格。

2019-02-20



2



Jaykey

空格，和后代选择器有什么不一样吗？

2019-06-19



1

1



啊咩

|| 列选择器 是最新的标准出的吗？ 我在w3school和 菜鸟教程 都找不到这个选择器

作者回复: Selector Level 3里的，不过不太重要，不用太在意。

2019-02-22



1



么么

按照文章的描述，css的匹配好像是从左到右匹配的，但是网上有很多文章说的是从右往左进行匹配的，这样的规则可以优化重复选择计算。那这种说法与文章上的是否右冲突？

作者回复: 是没错，浏览器多数实现是从右往左匹配的，这个说法早年我还设计过实验，但是现在我觉得这是比较偷懒的做法，实际上关于”实现“并没有一定的规定，我这里讲的也仅仅是一种思路。

2019-02-19



1



胡永

css选择器这里面最重要的一条规则就是没有父选择器，根据流式处理的dom规则，这样就节省了很多的重新计算

2019-09-04



令狐洋葱

老师，DOM 和 CSSOM 都是流式构建的，按照您的讲解，我理解是 CSSOM 必须先解析好，才能和 DOM 做样式上的融合，然后流式绘制已经解析好的 DOM，单纯HTML的解析我认为时间是可以忽略不计的，遇到 script 标签才会导致 DOM 解析有停顿，这时候页面才会做渲染，所以我理解上的流式渲染是分批的，遇到 js 就立即渲染之前融合好的布局，不知道这样理解流式渲染是否正确？还望老师指正，谢谢。

作者回复: 不是分批的，也跟JS无关。

2019-05-30



小王

处理后退的情况那里没太理解，当找到匹配了a#b的元素才开始检查它的所有子代是否匹配.cls。那第三个span节点不是a#b元素的子节点，它本身就不会被选中的。为什么文中提到“必须使得规则a#b .cls后退一步，这样第三个span才不会被选中”

2019-05-08



关山楂

我还是比较认同，从左往右匹配的规则，这样就像老师讲的可以在构建dom树的同时来进行匹配css规则，相当于同时构建渲染树了，而不必等到dom构建完毕在进行css的规则匹配，进行构建渲染树，虽然从右向左匹配对于复杂的选择器更优，但是这里面浪费了等待dom构建完毕才能使用的css匹配规

则。而且相对于同为id等单一选择器而言，明显从左向右更具有优势！这是我的一点想法，望老师指正！

2019-05-08

