12 CSS规范：写出一套优雅的 CSS 代码

更新时间：2019-07-04 11:04:34

> 自信和希望是青年的特权。
>
> ——大仲马

这一章到这里就快要结束了，这章的最后，我们来聊一下 CSS 的规范。我们要知道，代码主要是给人看的，其次才是给机器来运行的。所以写代码时候还是要有个规矩，尽量写出一手漂亮的代码。我们这一节要介绍的就是在做样式开发时要遵守的一些规范。这些规范里有些是为了让代码更简洁易懂，也就是为了让人看起来舒服；还有些是为了提高代码的效率，为了提高代码在机器上的运行质量。

文件引用规范

先说加载的规范，这个规范主要是为了提高**页面加载速度**或者是**首屏的速度**。

1. **CSS 文件或样式在 head 标签中引用**。页面的渲染需要 CSS，所以尽量早的让 CSS 文件加载出来。
2. **JS 文件要放在 body 标签尾部**。页面里加载和运行 JS 都会阻塞页面的渲染过程，所以把 JS 放在尾部可以加快首屏显示的速度，但对整个页面完成加载的时间没什么影响。
3. **使用压缩后的文件**。线上使用的静态文件，尽量都是压缩好的，CSS 使用 .min.css 形式，JS 使用 .min.js 形式，这样可以减少文件的体积，从而减少下载的时间。
4. **减少 import 方式引用 css 文件**。import 方式引入的 CSS 文件要等原 CSS 文件加载并解析后才会去请求，会拖慢 CSS 文件的加载速度。

选择器的书写规范

CSS 选择器在使用的时候尽量遵从以下规范：

一、选择器命名规范

- 选择器都用小写字母。

- 选择器的命名要使用英语，这是编程的通用语言。
- 选择器的名字尽量简短，但要有实际意义，不能为了文件的体积而忽略代码的可读性。
- 逗号后要有空格，一般在选择器的逗号后面加一个空格，这样不至于看起来拥挤。
- 使用-分割，如果有需要多个单词的选择器，几个单词中间用-分隔，比如 page-wrap、btn-small 等。

二、选择器使用规范

- CSS 的属性中需要引号的地方使用单引号，HTML 中的属性使用双引号。
- CSS 选择器在使用的时候，要把样式限定在某个 HTML 区域里生效。这样可以防止不同区域的元素互相影响
- 选择器合并。有相同样式的多个选择器，使用分组选择器，可以减小文件的体积。
- 尽量不使用通配选择器或标签选择器。这两种选择器效率比较低，尽量使用类选择器来代替，只有在需要改变元素默认属性的时候再使用。
- 最右侧的选择器尽量精确。选择器中最后一位的选择器尽量使用类选择器这种比较精确的选择器，因为选择器的读取是由右至左，最右边的选择器会先去遍历，如果最后使用了标签选择器，那么查找样式的消耗就会增多。
- 选择器的嵌套不宜太长。选择器在读取的时候都是一层层的去查找的，所以使用太长的选择器也会增加查找的消耗。
- 在可以的情况下用子代选择器代替后代选择器。子代选择器只需要做一层的查找，而后代选择器需要一直查找到根节点，所以子代选择器的效率会更高一点。
- 使用样式继承。对于可以继承的样式，尽量在父节点加入样式，而不要给每一个子节点都加样式。

属性的书写规范

一、使用缩写

在 CSS 中有很多属性或属性值可以缩写，在能用缩写的地方尽量使用缩写。

1、**属性的缩写**。CSS 中有些属性是可以合并的，如:

```
margin-top: 10px;
margin-bottom: 0;
margin-left: 5px;
margin-right: 5px;
```

上面这几组 margin 相关的属性占了四条样式，我们可以使用一条 margin 属性代替这四个方向的 margin：

```
margin: 10px 5px 0 5px;
```

一般带有方向的属性，缩写的时候各个方向的值都是按着“上 右 下 左”的顺序写的。另外如果四个方向值一样，可以直接用一个值代替四个方向；如果左右方向的值一样，则可以省略最后一个左侧的值。上面这条缩写也可以写成：

```
margin: 10px 5px 0;
```

2、**颜色的缩写**。在使用十六进制颜色的时候，如果 rgb 三个颜色位置中，每两位的颜色值相同，可以把六位的颜色写成三位。如：

```
color: #22ffcc;
```

就可以写成：

```
color: #2fc;
```

这两种写法是等效的，但要注意的是如果需要兼容低版本 IE 浏览器，还是要用六位的颜色值。

3、**数字的缩写**。在 CSS 中如果整数部分是 0 的小数，可以忽略小数点前面的 0；如果属性值是 0，则可以忽略属性值的单位。如:

```
font-size: 0.8rem;
padding: 0px;
```

这两条属性就可以做简写：

```
font-size: .8rem;
padding: 0;
```

二、属性顺序的规范

理论上，CSS 的属性是一条一条解析执行的。这种情况下，就要把能确定大小和位置的属性写在前面，把对布局没什么影响的属性写在后面，避免返工。一般说的使用顺序如下:

```
1. 位置属性 (position, top, right, z-index, display, float等)
2. 大小 (width, height, padding, margin)
3. 文字系列 (font, line-height, letter-spacing, color- text-align等)
4. 背景 (background, border等)
5. 其他 (animation, transition等)
```

但是这条规范仅供参考，不强制要求。因为我觉得稍微有点脑子的浏览器也不会去自上而下的逐条执行，最起码也要把这个代码块读完，对属性做个排序再去解析执行，而实际情况应该会比我说的更聪明一些。所以这一条规范只作为建议。

三、属性使用的规范

1. 不大面积的使用 gif 图片。显示 gif 图片的消耗比较大，所以一个页面里不要大面积使用 gif 图片。
2. 尽量不要对图片进行缩放，这也是一个高消耗的操作。
3. 减少高消耗属性的使用 box-shadow/border-radius/filter/ 透明度 /:nth-child 等
4. 动画里使用 3D 属性代替一般属性，如使用 transform、scale 等代替原始的 width、height、margin 等，因为这些 CSS3 的属性可以调用 GPU 进行渲染，会减少资源的消耗并提高动画的流畅度。

注释规范

在代码里还有比较特殊的一部分就是注释。这部分内容不参与代码的运行，只是为了提供给开发者看。我们对注释有如下要求：

一、文件头注释。

在文件头部加上注释是为了记录文件的创建者、创建时间、最后更改者和更改时间。这样在一个项目组里，如果遇到开发上的问题，可以直接根据文件头的注释找到这个文件所属人和操作时间。一个 CSS 文件头的注释可以按如下格式:

```
/*
 * @Author: Rosen
 * @Date: 2019-04-18 20:09:21
 * @Last Modified by: Rosen
 * @Last Modified time: 2019-05-05 10:21:21
 */
```

二、普通注释。

在业务里也需要注释，这种注释我们用标准的注释圈起来就行，最好是在注释的文本两边留下个空格，这样不会显得拥挤。如下：

```
/* 头部导航 */
.nav-top{
    background: #ccc;
}
```

我们在书写样式的时候，对于关键位置一定要随手加注释，免得过一阵自己写过的代码都不知道是干什么用的了。

CSS-Reset

最后要介绍一下 CSS-Reset。在浏览器里每个元素都有默认的样式，也就是你不给它任何样式也是可以显示的。但问题是，不同的浏览器中默认样式是不一致的，并且这些样式也可能和我们的业务代码有冲突。因此我们在开发的时候就要把这些默认样式的差异抹平，用到的方式就是 CSS-Reset。

CSS-Reset 的做法其实就是自己写一组 CSS 样式，把浏览器的默认样式覆盖掉。这样即可以抹平浏览器的差异，还可以自己定义元素的默认样式。CSS-Reset 是一组公共的样式，所以在它里面使用标签选择器还是比较合适的（也有用通用选择器的，效率更低，不推荐）。下面放上我们这个项目的 CSS-Reset 文件：

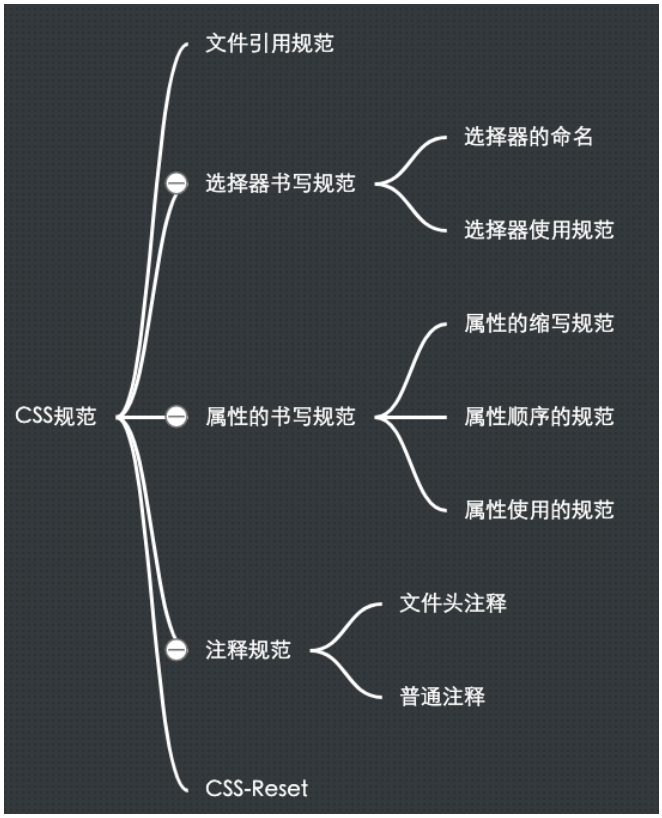
```
/* 去掉所有元素的内外边距 */
html, body, div, span,
h1, h2, h3, h4, h5, h6, p, pre,
a, img,  ul, li, form, label, input,
table, tbody, tfoot, thead, tr, th, td,
audio, video {
 margin: 0;
 padding: 0;
}
/* 统一全局字体 */
body {
 font-family: -apple-system-font,BlinkMacSystemFont,"Helvetica Neue","PingFang SC","Hiragino Sans GB","Microsoft YaHei UI","Microsoft YaHei",Arial,sa
ns-serif
}
/* 列表元素去掉默认的列表样式 */
ol, ul {
 list-style: none;
}
/* Table元素的边框折叠 */
table {
 border-collapse: collapse;
 border-spacing: 0;
}
/* 去掉默认的下划线 */
a{
 text-decoration: none;
}
/* 去掉input自带的边缘效果和背景颜色 */
input{
 outline: none;
 background: none;
}
```

这个 CSS-Reset 文件里是重置了一些比较常用标签里容易有问题的属性。网上也有那种特别全的 reset 文件，但是里面很多标签我们都用不到，所以还我通常是使用自定义的 reset 文件，有新的标签样式需要重置的时候再把它加进这个文件里。

小结

这一节里规定了一些常用的规范。里面有些规范是对代码的运行有帮助的，比如可以提高加载速度、首屏显示速度等。而也有些规范，比如 CSS 里空格的使用等，这些规范是为了让整体的风格统一，对代码的执行没有什么影响。

这一节的内容结构如下:



实际开发中有很多同学，尤其是新人，不太注重代码规范，觉得代码能跑起来就行了。代码不规范通常不会影响运行效果，但是一旦出现由不规范代码导致的问题就很难发现。这里要么没坑，要么就是大坑，所以一般踩过几次也就记住写代码的时候要规矩一些了。另外代码的规范程度也能反映出一个开发人员的素养和水平，我在面试别人的时候如果看到七扭八歪的代码，一般就会提高面试的难度了，我觉得应该也有不少面试官会和我一样。所以希望同学们从一开始就有一个好的编码习惯，让别人看的舒服，自己也能少踩坑。

我们这一节的内容就到这，至此这一章的内容也都讲完了。下一节我们把这一章的内容再重新过一遍，做个总结。

11 对比分析 CSS屏幕适配方案
13 本章总结