21种优化CSS和加快网站速度的方法，看看你知道几种?

web前端开发 12月3日

来源 | https://github.com/qq449245884/xiaozhi/issues/28

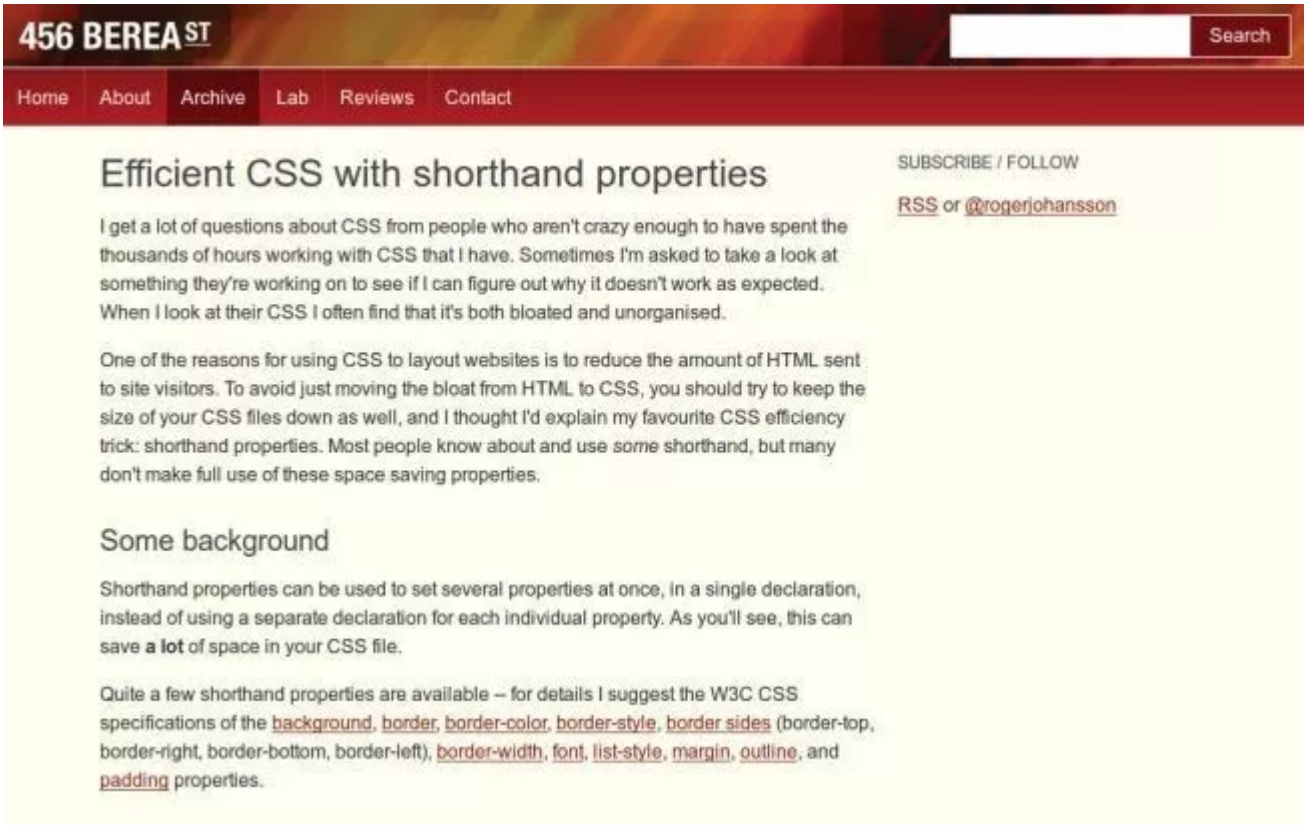
CSS 必须通过一个相对复杂的管道，就像 HTML 和 JavaScript一样，浏览器必须从服务器下载文件，然后进行解析并将其应用于DOM。由于优化程度极高，这个过程通常非常快——对于不基于框架的小型 web 项目，CSS通常只占总资源消耗的一小部分。

框架打破了这种平衡。包括一个 JavaScript GUI 堆栈，如 jQuery UI，可以观察 CSS, JS 和 HTML大小逐渐的变大。通常，开发人员最后才会感到压力，当他们用一个强大的 8 核工作站后面，使用 T3 internet 时，没有人关心速度，这随着延迟或 cpu 受限设备的出现而改变。

优化CSS需要一个多维的方法。虽然手工编写的代码可以使用各种技术进行简化，但是手工检查框架代码是低效的。在这些情况下，使用自动化的简化会产生更好的结果。

下面的步骤将带我们进入 CSS 优化的世界。并不是每一个都可以直接应用到你的项目中，但是一定要记住它们。

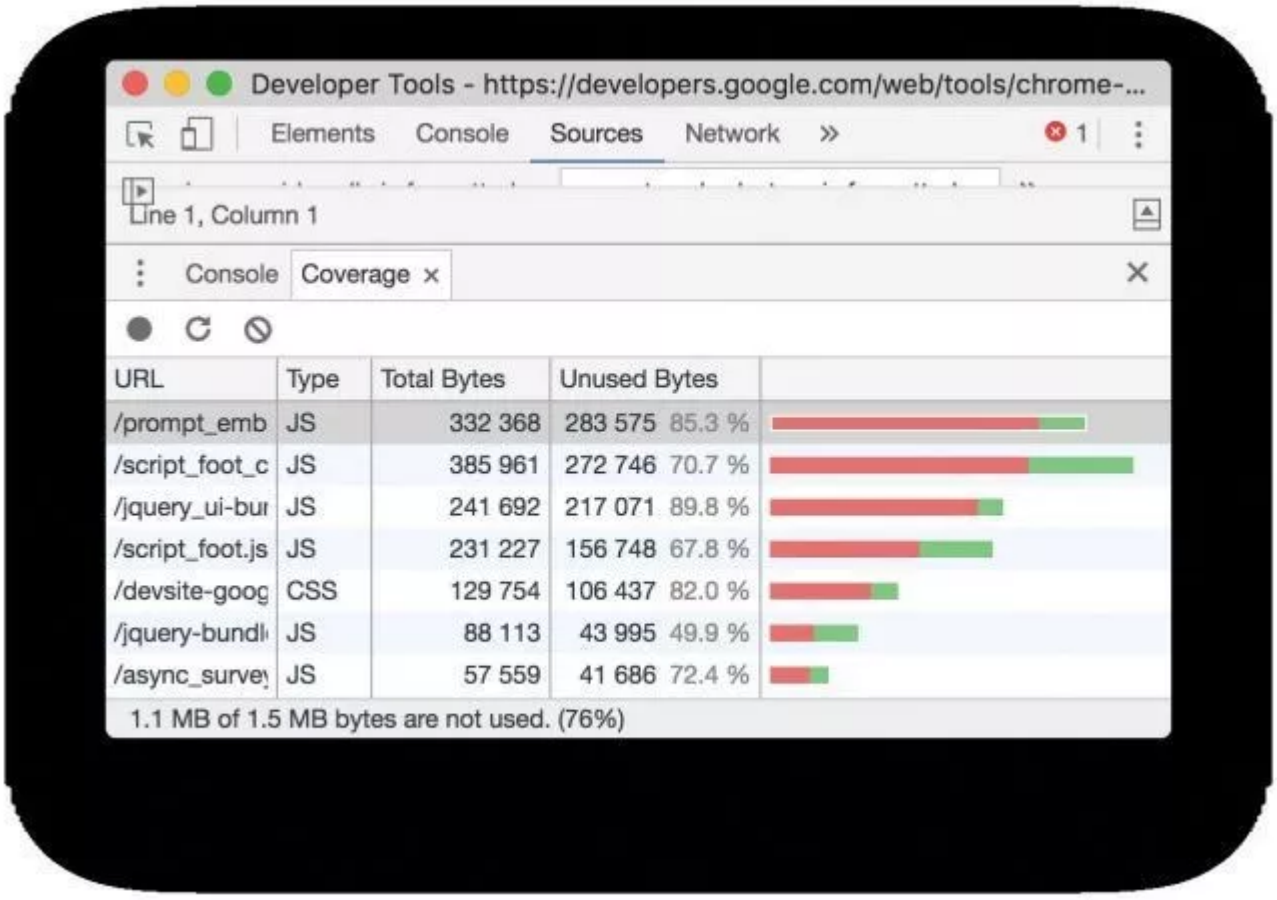
01. 使用简写



使用缩写语句，如下面所示的 `margin` 声明，可以从根本上减小 CSS 文件的大小。在 google 上搜索 `CSS Shorthand` 可以找到许多其他的速记形式。

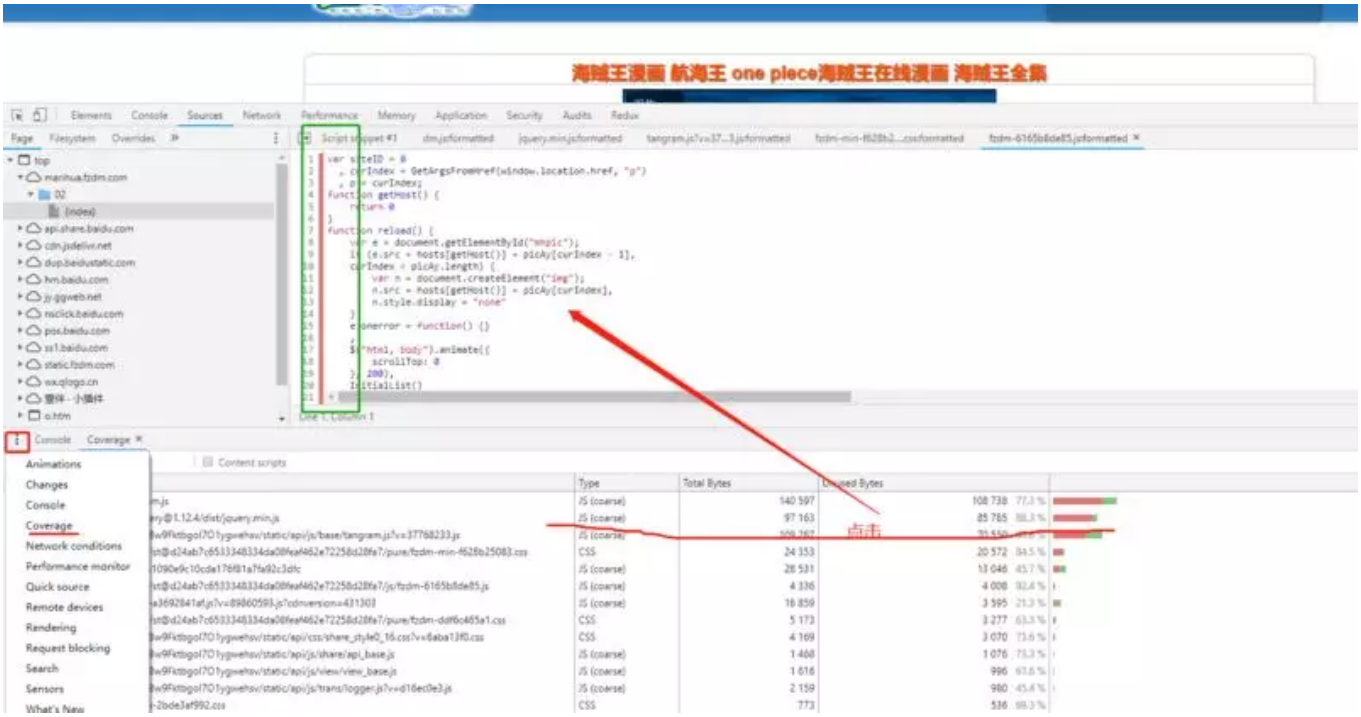
```
p { margin-top: 1px;
    margin-right: 2px;
    margin-bottom:  3px;
    margin-left: 4px; }
p { margin: 1px 2px 3px 4px; }
```

02. 查找并删除未使用的 CSS

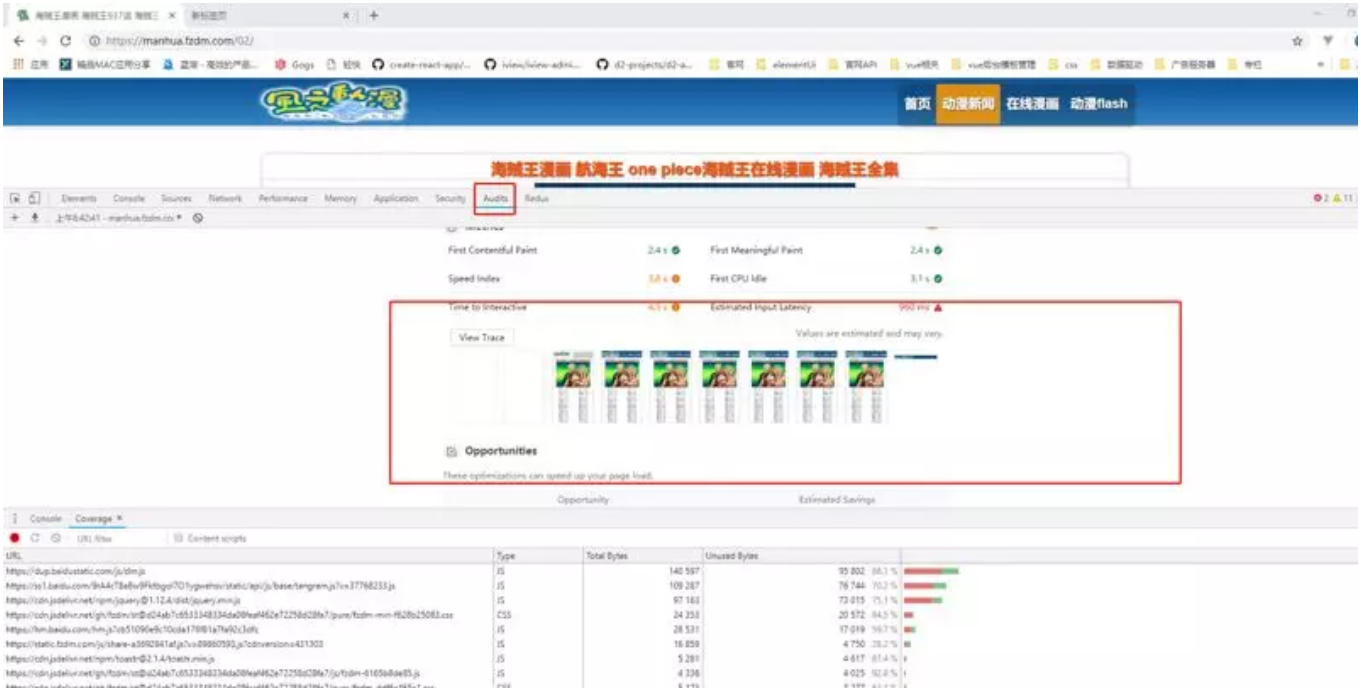


删除不必要的部分 CSS，j显然会加快网页的加载速度。谷歌的Chrome浏览器有这种开箱即用的功能。只需转到查看>开发人员>开发人员工具，并在最近的版本中打开Sources选项卡，然后打开命令菜单。然后，选择Show Coverage，在Coverage analysis窗口中高亮显示当前页面上未使用的代码，让您大开眼界。

打开谷歌浏览器开发都工具，在 `Conlse` 旁边更多选择 `Coverage`，就可以看到未使用的 CSS, 点击对应的项，高亮显示当前页面上未使用的代码，让你大开眼界：



03. 以更便捷的方式做到这一点



在逐行分析中导航并不一定便捷，使用谷歌浏览器的 Audits 就可以快速帮我们分析，使用方式，打开开发者工具，点击 `Audits` 栏位，点击 **Run audits** 后就开始分析结果。

04. 注意这些问题

请记住，对 CSS 的自动分析总是会导致错误。用压缩后的 CSS 文件替换 未压缩CSS文件之后，对整个网站进行彻底的测试——没有人知道优化器会导致什么错误。

05.内联关键 CSS

加载外部样式表需要花费时间，这是由于延迟造成的——因此，可以把最关键的代码位放在 `head` 中。但是，请确保不要做得过火，记住，执行维护任务的人员也必须读取代码。

```
<html>
  <head>
    <style>
      .blue{color:blue;}
</style>
    </head>
  <body>
    <div class="blue">
      Hello, world!
    </div>
```

06.允许反并行解析

`@import` 将 CSS 样式方便添加代码中。遗憾的是，这些好处并不是没有代价的:由于 `@import` 可以嵌套，因此无法并行解析它们。更并行的方法是使用一系列 标记，浏览器可以立即获取这些标记。

```
@import url("a.css");
@import url("b.css");
@import url("c.css");
<link rel="stylesheet" href="a.css">
<link rel="stylesheet" href="b.css">
<link rel="stylesheet" href="c.css">
```

07. 用 CSS 替换图片

几年前，一套半透明的 png 在网站上创建半透明效果是司空见惯的。现在，CSS过 滤器提供了一种节省资源的替代方法。例如，以下这个代码片段可以确保所讨论的图片显示为其自身的灰度版本。

```
img {
    -webkit-filter: grayscale(100%);
    /* old safari */
    filter: grayscale(100%);
}
```

08.使用颜色快捷方式

常识告诉我们，六位数的颜色描述符是表达颜色最有效的方式。事实并非如此——在某些情况下，速记描述或颜色名称可以更短。

```
target { background-color: #ffffff; }
target { background: #fff; }
```

09. 删除不必要的零和单位

CSS 支持多种单位和数字格式。它们是一个值得感谢的优化目标——可以删除尾随和跟随的零，如下面的代码片段所示。此外，请记住，零始终是零，添加维度不会为包含的信息附带价值。

```
padding: 0.2em;
margin: 20.0em;
avalue: 0px;
padding: .2em;
margin: 20em;
avalue: 0;
```

10. 消除过多分号

这种优化需要谨慎，因为它会影响代码的更改。CSS的规范允许省略属性组中的最后一个分号。由于这种优化方法所节省的成本很小，所以我们主要针对那些正在开发自动优化的程序员说明这一点。

```
p {
. . .
  font-size: 1.33em
}
```

11.使用纹理图集

由于协议开销的原因，加载多个小图片的效率很低。CSS 精灵将一系列小图片组合成一个大的 PNG 文件，然后通过 CSS 规则将其分解。[TexturePacker][7] 等程序大大简化了创建过程。

```
.download {
  width:80px;
  height:31px;
  background-position: -160px -160px
}

.download:hover {
  width:80px;
  height:32px;
  background-position: -80px -160px
}
```

12. 省略 px

提高性能的一个简单方法是使用CSS标准的一个特性。为 0 的数值默认单位是 `px` —— 删除 `px` 可以为每个数字节省两个字节。

```
h2 {padding:0px; margin:0px;}
h2 {padding:0; margin:0}
```

13. 避免需要性能要求的属性

分析表明，一些标签比其他标签更昂贵。以下这些解析会影响性能—如果在没有必要的情况，尽量不要使用它们。

border-radius
box-shadow
transform
filter
:nth-child
position: fixed;

14. 删除空格

空格——考虑制表符、回车符和空格——使代码更容易阅读，但从解析器的角度看，它没有什么用处。在发布前删除它们，更好的方法是将此任务委托给 shell 脚本或类似的工具。

15. 删除注释

注释对编译器也没有任何作用。创建一个自定义解析器，以便在发布之前删除它们。这不仅节省了带宽，而且还确保攻击者和克隆者更难理解手头代码背后的思想。

16. 使用自动压缩

Yahoo 的用户体验团队创建了一个处理许多压缩任务的应用程序。它以 JAR 文件的形式发布，在这里可用，并且可以使用所选的JVM运行。

```
java -jar yuicompressor-x.y.z.jar
Usage: java -jar yuicompressor-x.y.z.jar
 [options] [input file]
Global Options
    -h, --help               Displays this
 information
    --type <js|css>          Specifies the
 type of the input file
```

17. 在 NPM 运行它

如果你希望将产品集成到 Node.JS 中，请访问 [npmjs.com/package/yuicompressor][8]。维护不良的存储库包含一组包装器文件和JavaScript API。

```
var compressor = require('yuicompressor');
 compressor.compress('/path/to/
file or String of JS', {
    //Compressor Options:
    charset: 'utf8',
    type: 'js',
```

18. 保持 Sass 的检查

虽然 CSS 选择器的性能不像几年前那么重要(请参阅参考资料)，但是像 Sass 这样的框架有时会产生非常复杂的代，不时查看输出文件，并考虑优化结果的方法。

19. 设置缓存

有句老话说，最快的文件永远不会通过网络发送。让浏览器缓存请求有效地实现这一点。遗憾的是，缓存头的设置必须在服务器上进行。充分上面讲的的两个 Chrome 工具，它们提供了一种快速分析更改结果的方法。

20. 打破缓存

设计人员通常不喜欢缓存，因为他们担心浏览器会缓存上次的样式表。解决这个问题的一个简单方法是包含带有文件名的标记。遗憾的是，由于一些代理拒绝缓存具有“动态”路径的文件，此步骤所附带的代码中概述的方案并不适用于所有地方。

```
<Link rel="stylesheet" href="style.css?v=1.2.3">
```

21. 不要忘记基础知识

优化CSS只是游戏的一部分。如果你的服务器不使用 HTTP/2 和 gzip 压缩，那么在数据传输期间会损失很多时间。幸运的是，解决这两个问题通常很简单。我们的示例显示了对常用

Apache 服务器的一些调整。如果您发现自己在一个不同的系统上，只需参考服务器文档即可。

```
pico /etc/httpd/conf/httpd.conf
AddOutputFilterByType DEFLATE text/html
AddOutputFilterByType DEFLATE text/css
```

End