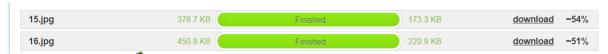
1.图片压缩对比

通过jpg压缩后



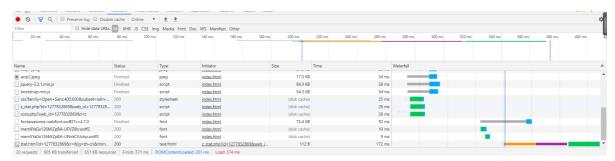
通过png压缩后

15.png	378.7 KB	Finished	173.3 KB	download -54%
16.png	450.8 KB	Finished	220.9 KB	download -51%

结果看来是相同的..

2.某公司页面压缩优化

在压缩以前先来看看最先开始未经压缩的效果



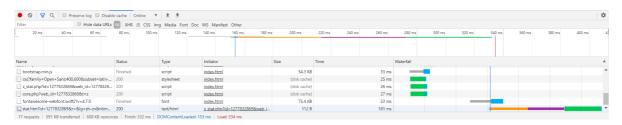
应该可以很清楚的看到DOMContentLoaded数值为201ms,而Load数值为374ms

下面我们来看看文字压缩后的

首先是只对css进行压缩后的,下图是我对style.css进行的压缩

```
😈 index.html
  README.md •
                 style.min.css X
                                                  😈 inde 🖏
mobileWeb 〉 实验5 〉 demo 〉 🧧 style.min.css 〉 😭 .top-right img
      *{outline:none}html,body{height:100%}body{font-family:'Op
      Sans',sans-serif;font-size:13px;line-height:1.6;
      color: ■ #8c8c8c; background-color: ■ #fff}h1,h2,h3,h4,h5,h6
      {font-family:'Open Sans',sans-serif}a{color: □#404040}a:h
      {color: ■#404040;transition-property:all;transition-durat
      3s;transition-timing-function:linear}a.none:hover
      {text-decoration:none}.site-main>section:first-child
      {display:flex;flex-direction:column}.top
      {background-color: ■#404040; padding:5px 0; color: ■#fff}.
      {margin:0}.top ul{margin:0;padding:0}.top li i{color: ■#f
      top li a{color: #fff}.top li a:hover{text-decoration:nor
      top li a:hover,.top li a:hover i{color: ■#005FA6;
      transition-property:all;transition-duration:0.3s;
      transition-timing-function:linear}.navbar
      {-moz-border-radius:0;-webkit-border-radius:0;
      border-radius:0;margin-bottom:0}.navbar .container
      {position:relative}.navbar-default{-moz-transition:all 0.
      ease-out;-o-transition:all 0.3s ease-out;
      -webkit-transition:all 0.3s ease-out; transition:all 0.3s
      ease-out;width:100%;border:none;border-bottom:1px solid
       #e7e7e7;background-color: #fff}.navbar-default .
```

现在我们把这个css引入并查看一下结果



应该可以很清楚的看到DOMContentLoaded数值为153ms,而Load数值为334ms,性能有了明显的提升。

下面再将HTML进行压缩。

这是我对HTML进行的压缩



下面打开浏览器查看一下



应该可以很清楚的看到DOMContentLoaded数值为145ms,而Load数值为190 ms,load速度和最初相比将近节省了一半的时间,速度得到了大幅度的提升。

速度提升的原因就是字符的减小吧,文件大小也明显的下降。

o index.html	2020/4/3 15:18	Chrome HTML D	12 KB
indexMin.html	2020/4/3 15:10	Chrome HTML D	8 KB
🔊 style.css	2020/3/27 9:32	层叠样式表文档	15 KB
🔬 style.min.css	2020/4/3 14:59	层叠样式表文档	13 KB

下面进行http请求的优化

首先先做的就是合并文字资源,减少下载频率



应该可以很清楚的看到**DOMContentLoaded**数值为139ms,而**Load**数值为 211 ms ,减少了发出的HTTP 请求的数量,又快了一些。

接下来就是图片合并将图片制作为精灵图。效果如下图:



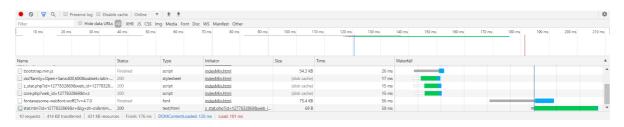
再通过css修改



应该可以很清楚的看到DOMContentLoaded数值为135ms,而Load数值为184 ms,明显的又快了些速度提升的原因就是请求数量的减小,请求数量的减少,大大缩短了时间,上次实验报告中也有提到,文件资源下载的过程中很多时间都是Stalled等待时间,减少请求次数等待时间也大大的缩短了很多。

最后是压缩图片资源把所有用到的图片像(1.压缩图片对比)中的方式进行压缩,这里就不再一张一张放截 图了,md好麻烦啊我这里保存了一大串截图了...

话不多说放结果



应该可以很清楚的看到DOMContentLoaded数值为120ms,而Load数值为181 ms,明显的又快了些这次的时间的减少原因很明显是因为,图片资源大小的减少,从而加快了加载的速度。

Over.