# 页面优化

[**HTML页面加载和解析流程**](http://blog.csdn.net/xifeijian/article/details/10813339)   
1. 用户输入网址（假设是个html页面，并且是第一次访问），浏览器向服务器发出请求，服务器返回html文件；   
2. 浏览器开始载入html代码，发现＜head＞标签内有一个＜link＞标签引用外部CSS文件；   
3. 浏览器又发出CSS文件的请求，服务器返回这个CSS文件；   
4. 浏览器继续载入html中＜body＞部分的代码，并且CSS文件已经拿到手了，可以开始渲染页面了；   
5. 浏览器在代码中发现一个＜img＞标签引用了一张图片，向服务器发出请求。此时浏览器不会等到图片下载完，而是继续渲染后面的代码；   
6. 服务器返回图片文件，由于图片占用了一定面积，影响了后面段落的排布，因此浏览器需要回过头来重新渲染这部分代码；   
7. 浏览器发现了一个包含一行Javascript代码的＜script＞标签，赶快运行它；   
8. Javascript脚本执行了这条语句，它命令浏览器隐藏掉代码中的某个＜div＞ （style.display=”none”）。突然少了这么一个元素，浏览器不得不重新渲染这部分代码；   
9. 终于等到了＜/html＞的到来，浏览器泪流满面……   
10. 等等，还没完，用户点了一下界面中的“换肤”按钮，Javascript让浏览器换了一下＜link＞标签的CSS路径；   
11. 浏览器召集了在座的各位＜div＞＜span＞＜ul＞＜li＞们，“大伙儿收拾收拾行李，咱得重新来过……”，浏览器向服务器请求了新的CSS文件，重新渲染页面。

js阻塞其他资源的加载的原因是：浏览器为了防止js修改DOM树，需要重新构建DOM树的情况出现；

### ****编写CSS时应该注意：****

CSS选择符是从右到左进行匹配的。从右到左！所以，#nav li 我们以为这是一条很简单的规则，秒秒钟就能匹配到想要的元素，但是，但是，但是，是从右往左匹配啊，所以，会去找所有的li，然后再去确定它的父元素是不是#nav。，因此，写css的时候需要注意：

1. dom深度尽量浅。
2. 减少inline javascript、css的数量。
3. 使用现代合法的css属性。
4. 不要为id选择器指定类名或是标签，因为id可以唯一确定一个元素。
5. 避免后代选择符，尽量使用子选择符。原因：子元素匹配符的概率要大于后代元素匹配符。后代选择符;#tp p{} 子选择符：#tp>p{}
6. 避免使用通配符，举一个例子，.mod .hd \*{font-size:14px;} 根据匹配顺序,将首先匹配通配符,也就是说先匹配出通配符,然后匹配.hd（就是要对dom树上的所有节点进行遍历他的父级元素）,然后匹配.mod,这样的性能耗费可想而知.

**2、使用浏览器缓存**

        对一个网站而言，CSS、javascript、logo、图标这些静态资源文件更新的频率都比较低，而这些文件又几乎是每次http请求都需要的，如果将这些文件缓存在浏览器中，可以极好的改善性能。通过设置http头中的cache-control和expires的属性，可设定浏览器缓存，缓存时间可以是数天，甚至是几个月。

        在某些时候，静态资源文件变化需要及时应用到客户端浏览器，这种情况，可通过改变文件名实现，即更新javascript文件并不是更新javascript文件内容，而是生成一个新的JS文件并更新HTML文件中的引用。  
        使用浏览器缓存策略的网站在更新静态资源时，应采用逐量更新的方法，比如需要更新10个图标文件，不宜把10个文件一次全部更新，而是应该一个文件一个文件逐步更新，并有一定的间隔时间，以免用户浏览器忽然大量缓存失效，集中更新缓存，造成服务器负载骤增、网络堵塞的情况。