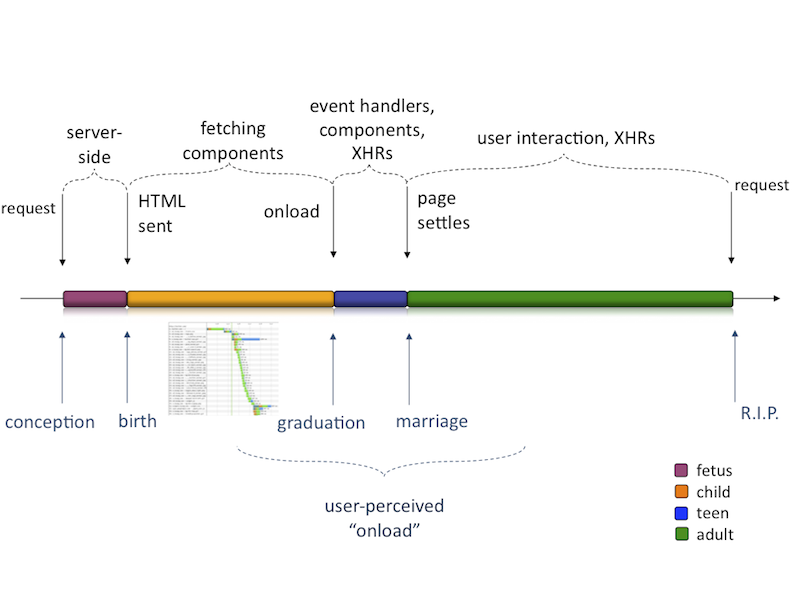
Yahoo前端优化十四条军规

相信互联网已经越来越成为人们生活中不可或缺的一部分。Ajax，flex 等等富客户端的应用使得人们越加“幸福”地体验着许多原先只能在 C/S 实 现的功能。

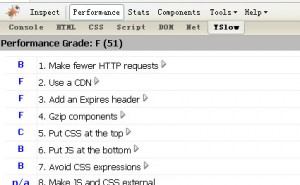
比如 Google 机会已经把最基本的 office 应用都搬到了互联网上。当然便利的同时毫无疑问的也使页面的速度越来越慢。自己是做前端开发的，在性能方面，根据 Yahoo 的调查，后台只占 5%，而前端高达 95% 之多，其中有 88% 的东西是可以优化的。



以上是一张 web2.0 页面的生命周期图。工程师很形象地讲它分成了“怀孕，出生，毕业，结婚”四个阶段。如果在我们点击网页链接的时候能够意识到 这个过程而不是简单的请求 - 响应的话，我们便可以挖掘出很多细节上可以提升性能的东西。今天听了淘宝小马哥的一个对 yahoo 开发团队对 web 性能研究的 一个讲座，感觉收获很大，想在 blog 上做个分享。

相信很多人都听过优化网站性能的 14 条规则。更多的信息可见 developer.yahoo.com

[](http://images.51cto.com/files/uploadimg/20120712/1059491.jpg)

[](http://images.51cto.com/files/uploadimg/20120712/1059492.jpg)

**第一条、尽可能的减少 HTTP 的请求数 （Make Fewer HTTP Requests ）**

http 请求是要开销的，想办法减少请求数自然可以提高网页速度。常用的方法，合并 css，js（将一个页面中的 css 和 js 文件分别合并）以及 Image maps 和 css sprites 等。当然或许将 css，js 文件拆分多个是因为 css 结构，共用等方面的考虑。阿里巴巴中文站当时的做法是开发时依然分开开发，然后在后台 对 js，css 进行合并，这样对于浏览器来说依然是一个请求，但是开发时仍然能还原成多个，方便管理和重复引用。yahoo 甚至建议将首页的 css 和 js 直接写在页面文件里面，而不是外部引用。因为首页的访问量太大了，这么做也可以减少两个请求数。而事实上国内的很多门户都是这么做的。

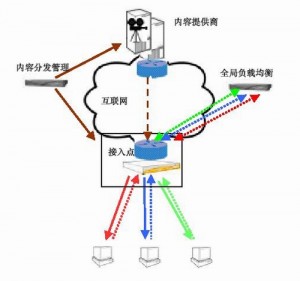
而 css sprites 是指只用将页面上的背景图合并成一张，然后通过 css 的 background-position 属性定义不过的值来取他的背景。淘宝和阿里巴巴中文站目前都是这样做的。有兴趣的可以看下淘宝和阿里巴巴的背景图。

<http://www.csssprites.com/> 这是个工具网站，它可以自动将你上传的图片合并并给出对应的 background-position 坐标。并将结果以 png 和 gif 的格式输出。

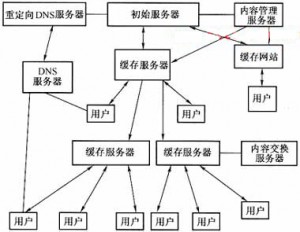
**第二条、使用 CDN（内容分发网络）: Use a Content Delivery Network**

说实话，对于 CDN 这一块自己并不是很了解，简单地讲，通过在现有的 Internet 中增加一层新的网络架构，将网站的内容发布到最接近用户的 cache 服务器内，通过 DNS 负载均衡的技术，判断用户来源就近访问 cache 服务器取得所需的内容，杭州的用户访问近杭州服务器上的内容，北京的访问 近北京服务器上的内容。这样可以有效减少数据在网络上传输的时间，提高速度。更详细地内容大家可以参考百度百科上对于 CDN 的解释。Yahoo! 把静态内容分布到 CDN 减少了用户影响时间 20% 或更多。

CDN 技术示意图：



CDN 组网示意图：



**第三条、 添加 Expire/Cache-Control 头：Add an Expires Header**

现在越来越多的图片，脚本，css，flash 被嵌入到页面中，当我们访问他们的时候势必会做许多次的 http 请求。其实我们可以通过设置 Expires header 来缓存这些文件。Expire 其实就是通过 header 报文来指定特定类型的文件在览器中的缓存时间。大多数的图片，flash 在发布后都是不需要经常修改的，做了缓存以后这样浏览器以后就不需要再从服务器下载这些文件而是而直接从缓存中读取，这样再次访问页面的速度会大大加快。一个典型的 HTTP 1.1 协议返回的头信息：

|  |
| --- |
| HTTP/1.1 200 OK Date: Fri, 30 Oct 1998 13:19:41 GMT Server: Apache/1.3.3 (Unix) Cache-Control: max-age=3600, must-revalidate Expires: Fri, 30 Oct 1998 14:19:41 GMT Last-Modified: Mon, 29 Jun 1998 02:28:12 GMT ETag: “3e86-410-3596fbbc” Content-Length: 1040 Content-Type: text/html |

其中通过服务器端脚本设置 Cache-Control 和 Expires 可以完成。

如，在 php 中设置 30 天后过期：

1. pHeader("Cache-Control: must-revalidate"); $offset = 60 \* 60 \* 24 \* 30; $ExpStr = "Expires: " . gmdate("D, d M Y H:i:s", time() + $offset) . " GMT"; Header($ExpStr);

也可以通过配置服务器本身完成，这些偶就不是很清楚了，呵呵。想了解跟多的朋友可以参考<http://www.web-caching.com/>

据我了解，目前阿里巴巴中文站的 Expires 过期时间是 30 天。不过期间也有过问题，特别是对于脚本过期时间的设置还是应该仔细考虑下，不然相应的脚本功能更新后客户端可能要过很长一段时间才能“感知”到这样的变化。所以，哪些应该缓存，哪些不该缓存还是应该仔细斟酌一番。

**第四条、启用 Gzip 压缩：Gzip Components**

Gzip 的思想就是把文件先在服务器端进行压缩，然后再传输。这样可以显著减少文件传输的大小。传输完毕后浏览器会 重新对压缩过的内容进行解压缩，并执行。目前的浏览器都能“良好”地支持 gzip。不仅浏览器可以识别，而且各大“爬虫”也同样可以识别，各位 seoer 可以放下心了。而且 gzip 的压缩比例非常大，一般压缩率为 85%，就是 说服务器端 100K 的页面可以压缩到 25K 左右再发送到客户端。具体的 Gzip 压缩原理大家可以参考 csdn 上的《gzip 压缩算法》 这篇文章。雅虎特别强调， 所有的文本内容都应该被 gzip 压缩: html (php), js, css, xml, txt… 这一点我们网站做得不错，是一个 A。以前我们的首页也并不是 A，因为首页上还有很多广告代码投放的 js，这些广告代码拥有者的网站的 js 没有经过 gzip 压缩，也会拖累我们网站。

以上三点大多属于服务器端的内容，本人也是粗浅地了解而已。说得不对的地方有待各位指正。

**第五条、将 css 放在页面最上面 （ Put Stylesheets at the Top）**

将 css 放在页面最上面，这是为什么？因为 ie，firefox 等浏览器在 css 全部传输完全之前不会去渲染任何的东西。理由诚如小马哥说得那样很简单。css，全称 Cascading Style Sheets (层叠样式表单)。层叠即意味这后面的 css 可以覆盖前面的 css，级别高的 css 可以覆盖级别低的 css。在 [css 之！important] 这篇文章的最下面曾简单地提到过这层级关系，这里我们只需要知道 css 可以被覆盖的。既然前面的可以被覆盖，浏览器在他完全加载完毕之后再去渲染无疑也是合情合理的很多浏览器下，如 IE，把样式表放在页面的底部的问题在于它禁止了网页内容的顺序显示。浏览器阻止显示以免重画页面元素，那用户只能看到空白页了。Firefox 不会阻止显示，但这意味着当样式表下载后，有些页面元素可能需要重画，这导致闪烁问题。所以我们应该尽快让 css 加载完毕

顺着这层意思，如果我们再细究的话，其实还有可以优化的地方。比如本站上面包含的两个 css 文件，

**第六条、将 script 放在页面最下面 （Put Scripts at the Bottom ）**

将脚本放在页面最下面的目的有那么两点：

1、 因为防止 script 脚本的执行阻塞页面的下载。在页面 loading 的过程中，当浏览器读到 js 执行语句的时候一定会把它全部解释完毕后在会接下来读下 面的内容。不信你可以写一个 js 死循环看看页面下面的东西还会不会出来。（setTimeout 和 setInterval 的执行有点类似于多线程，在相应的响应时间之前也会继续下面的内容渲染。）浏览器这么做的逻辑是因为 js 随时可能执 行 location.href 或是其他可能完全中断此页面过程的函数，即如此，当然得等他执行完毕之后再加载咯。所以放在页面最后，可以有效减少页面可 视元素的加载时间。

2、脚本引起的第二个问题是它阻塞并行下载数量。HTTP/1.1 规范建议浏览器每个主机的并行下载数不超过 2 个（IE 只能为 2 个，其他浏览器如 ff 等都是默认设置为 2 个，不过新出的 ie8 可以达 6 个）。因此如果您把图像文件分布到多台机器的话，您可以达到超过 2 个的并行下载。但是当脚本文件下载时，浏览器不会启动其他的并行下载。

当然对各个网站来说，把脚本都放到页面底部加载的可行性还是值得商榷的。就比如阿里巴巴中文站的页面。很多地方有内联的 js，页面的显示严重依赖于此，我承认这和无侵入脚本的理念相差甚远，但是很多“历史遗留问题”却不是那么容易解决的。

**第七条、避免在 CSS 中使用 Expressions （Avoid CSS Expressions ）**

不过这样就多了两层无意义的嵌套，肯定不好。还需要一个更好的办法。

**第八条、把 javascript 和 css 都放到外部文件中 （Make JavaScript and CSS External ）**

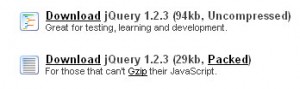
这点我想还是很容易理解的。不仅从性能优化上会这么做，用代码易于维护的角度看也应该这么做。把 css 和 js 写在页面内容可以减少 2 次请求，但也增 大了页面的大小。如果已经对 css 和 js 做了缓存，那也就没有 2 次多余的 http 请求了。当然，我在前面中也说过，有些特殊的页面开发人员还是会选择内联 的 css 和 js 文件。

**第九条、减少 DNS 查询 (Reduce DNS Lookups)**

在 Internet 上域名与 IP 地址之间是一一对应的，域名（kuqin.com）很好记，但计算机不认识，计算机之间的“相认”还要转成 ip 地址。在网络 上每台计算机都对应有一个独立的 ip 地址。在域名和 ip 地址之间的转换工作称为域名解析，也称 DNS 查询。一次 DNS 的解析过程会消耗 20-120 毫秒的 时间, 在 dns 查询结束之前，浏览器不会下载该域名下的任何东西。所以减少 dns 查询的时间可以加快页面的加载速度。yahoo 的建议一个页面所包含的域 名数尽量控制在 2-4 个。这就需要对页面整体有一个很好的规划。目前我们这点做的不好，很多打点的广告投放系统拖累了我们。

**第十条、压缩 JavaScript 和 CSS (Minify JavaScript )**

压缩 js 和 css 的左右很显然，减少页面字节数。容量小页面加载速度自然也就快。而且压缩除了减少体积以外还可以起到一定的保护左右。这点我们做得不错。常用的压缩工具有 JsMin、YUI compressor 等。另外像 http://dean.edwards.name/packer/ 还给我们提供了一个非常方便的在线压缩工具。你可以在 jQuery 的网页看到压缩过的 js 文件和没有压缩过的 js 文件的容量差别：



当然，压缩带来的一个弊端就是代码的可读性没了。相信很多做前端的朋友都遇到过这个问题：看 Google 的效果很酷，可是去看他的源代码却是一大堆 挤在一起的字符，连函数名都是替换过的，汗死！自己的代码也这样岂不是对维护非常不方便。所有阿里巴巴中文站目前采用的做法是在 js 和 css 发布的时候在 服务器端进行压缩。这样在我们很方便地维护自己的代码。

**第十一条、避免重定向 (Avoid Redirects )**

不久前在 ieblog 上看到过《Internet Explorer and Connection Limits》这篇文章，比如 当你输入 http://www.kuqin.com/ 的时候服务器会自动产生一个 301 服务器转向 http://www.kuqin.com/ ，你看浏览器的地址栏就能看出来。这种重定向自然也是需要消耗时间的。当然这只是一个例子，发生重定向的原因还有很多，但是不变的是每增加一次重定向就会增加一次 web 请求，所以因该尽量减少。

**第十二条、移除重复的脚本 (Remove Duplicate Scripts )**

这点我想不说也知道，不仅是从性能上考虑，代码规范上看也是这样。但是不得不承认，很多时候我们会因为图一时之快而加上一些或许是重复的代码。或许一个统一的 css 框架和 js 框架可以比较好的解决我们的问题。小猪的观点很对，不仅是要做到不重复，更是要做到可重用。

**第十三条、配置实体标签（ETags） (Configure ETags )**

这点我也不懂，呵呵。在 inforQ 上找到一篇解释得比较详细的说明《使用 ETags 减少 Web 应用带宽和负载》，有兴趣的同学可以去看看。

**第十四条、使 AJAX 缓存 (Make Ajax Cacheable )**

ajax 还要去缓存？做 ajax 请求的时候往往还要增加一个时间戳去避免他缓存。It’s important to remember that “asynchronous” does not imply “instantaneous”.（记住“异步”不是“瞬间”这一点很重要）。记住，即使 AJAX 是动态产生的而且只对一个用户起作用，他们依然可以被缓存。

原文链接：<http://www.itokit.com/2012/0704/74572.html>

1. [使用 jQuery 的 9 个误区](http://developer.51cto.com/art/201207/346751.htm)
2. [JavaScript+CSS3 写的 Web 版纸牌](http://developer.51cto.com/art/201206/343916.htm)
3. [如何优化你的 JS 代码](http://developer.51cto.com/art/201206/343167.htm)
4. [jQ、Yahoo API 和 HTML 5 开发天气预报应用](http://developer.51cto.com/art/201206/343171.htm)
5. [HTML 5 开发实战案例之网易微博](http://developer.51cto.com/art/201206/343145.htm)