目录

3 解读雅虎35条军规（下）

更新时间：2019-11-26 09:48:50

> 能够生存下来的物种,并不是那些最强壮的,也不是那些最聪明的,而是那些对变化作出快速反应的。
>
> ——达尔文

上一节我们讲解了雅虎军规当中的 Image、CSS、Cookie 以及 Server 这个四个方面,那么这一节我们继续将剩下的 JavaScript、Mobile、以及 Content 这三方面讲解完。

1.JavaScript

Put Scripts at the Bottom(把脚本放到底部)

脚本引起的问题是它们阻塞了并行下载。HTTP1.1 规范建议浏览器每个域名下不要并行下载超过2个组件。如果你的图片分散在不同服务器，那么你能并行下载多个图片。但当脚本在下载的时候，浏览器不会再下载其它文件，即使在不同域名下。

有些情况下把脚本移动到底部并不简单。比如，脚本中用了 document.write 来插入内容，它就不能被移动到底部。另外有可能有作用域问题。但大多数情况，有方法可以解决这些问题。

一个替代建议是使用异步脚本。defer 属性表明脚本不包含 document.write，是提示浏览器继续渲染的线索。不幸的是，Firefox 不支持。如果脚本能异步，那么也就可以移动到底部，这样可以大大加快网页运行速度。

> Tips：这里要注意JavaScript是会阻塞浏览器运行的，所以脚本文件尽量放到页面的最下面。

Make JavaScript and CSS External(使用外部JS和CSS)

目录

真实世界中使用外部文件一般会加快页面，因为 JS 和 CSS 文件被浏览器缓存了。内联的 JS 和CSS 怎在每次 HTML 文档下载时都被下载。内联减少了http请求，但增加了HTML文档大小。另一方面，如果 JS 和 CSS 被缓存了，那么 HTML 文档可以减小大小而不增加 HTTP 请求。

核心因素，就是 JS 和 CSS 被缓存相对于 HTML 文档被请求的频率。尽管这个因素很难被量化，但可以用不同的指标来计算。如果网站用户每个会话打开了多个页面，许多页面重复使用相同的 JS 和CSS，那么有很大可能用外部 JS 和 CSS 更好。

许多网站用这些指标计算后在中间位置。对这些网站来说，最佳方案还是用外部 JS 和 CSS 文件。唯一例外是内连更被主页偏爱，如http://www.yahoo.com/。主页每个会话可能只会打开少量甚至一个页面，这时候内联可能更快。

> Tips：这里我们就需要实际业务的需要选择内联还是外联。

Minify JavaScript and CSS(压缩JS和CSS)

压缩就是删除代码中不必要的字符来减小文件大小，从而提高加载速度。当代码压缩时，注释删除，不需要的空格（空白，换行，tab）也被删除。

混淆是对代码可选的优化。它比压缩更复杂，并且可能产生 bug。在对美国 top10 网站的调查，压缩可减小 21%，而混淆可减小 25%。

除了外部脚本和样式，内联的脚本和样式同样应该被压缩。

Remove Duplicate Scripts(删除重复的脚本)

在页面中引入相同的脚本两次会伤害性能。可能超出你的预料，美国 top10 网站的 2 家有重复脚本引入。两个主要因素造成同一页面引入相同脚本：团队规模和脚本数量。当确实引入重复脚本，会发出不必要的http请求和浪费js执行时间。

发出不必要的 http 请求发生在 IE 而不是 Firefox。在 IE，如果外部脚本引入两次且没有缓存，它会发出2个请求。即使脚本被缓存，刷新时也会发出额外请求。

除了增加http请求，时间被浪费在执行脚本多次上。不管IE还是Firefox都会执行多次。

一种避免多次引入脚本的方法是在模板系统实现一个脚本管理模块。

> Tips：其实这里简单剖析一下，和我们现在使用 React 和 Vue 框架的时候经常会用到的性能优化方法按需加载非常相似，那么对应到 Angular 当中就是按需注入。

Minimize DOM Access(最小化DOM访问)

用 JS 访问 DOM 元素是缓慢的，所以为了响应更好的页面，你应该：

- 避免用 JS 实现固定布局

> Tips：能用CSS搞定的事情，就尽量使用CSS，使用JS操作DOM对性能开销非常大。

Develop Smart Event Handlers(开发聪明的事件处理)

有时候页面看起来不那么响应（响应速度慢），是因为绑定到不同元素的大量事件处理函数执行太多次。这是为什么使用*事件委托*是一种更好的解决方法。

另外，你不必等到 onload 事件来开始处理 DOM 树，DOMContentLoaded 更快。大多时候你需要的只是想访问的元素已在 DOM 树中，所以你不必等到所有图片也被下载。

> Tips：
>
> - onLoad是的在页面所有文件加载完成后执行
> - DomContentLoad是Dom加载完成后执行，不必等待样式脚本和图片加载

2. Mobile

Keep Components under 25K(保持组件小于25K)

这个限制与 iPhone 不缓存大于 25K 的组件相关。

Pack Components into a Multipart Document(组件打包到一个多部分文档)

将组件打包到多部分文档就像带有附件的电子邮件，它可以帮助您通过一个 HTTP 请求获取多个组件（请记住：HTTP 请求很昂贵）。使用此技术时，首先检查用户代理是否支持它(iPhone 不支持)。

3. Content

Make Fewer HTTP Requests(最小化 http 请求)

到终端用户的响应时间 80% 花在前端：大部分用于下载组件 js/css/image/flash 等等。减少组件数就是减少渲染页面所需的 http 请求数。这是更快页面的关键。

减少组件数的一个方法就是简化页面设计。保持富内容的页面且能减少 http 请求，有以下几个技术：

Combined files (合并文件)，如合并 js，合并 css 都能减少请求数。如果页面间脚本和样式差异很大，合并会更具挑战性,同时也这样也可以减少发布所需要的时间。

CSS Sprites。雪碧图可以合并多个背景图片，通过 background-image 和 background-position 来显示不同部分。

Image maps (雪碧图)。合并多个图片到一个图片，一般用于如导航条。由于定义坐标的枯燥和易错，一般不推荐。

减少请求数是为第一次访问页面的用户提高性能的最重要的指导。

> Tips：Inline images 这里我们一些矢量图标转换为base64编码，然后直接内嵌到HTML文件或者CSS文件当中，减少http请求。

Reduce DNS Lookups(减少DNS查询)

就像电话簿，你在浏览器地址栏输入网址，通过 DNS 查询得到网站真实 IP。

DNS 查询被缓存来提高性能。这种缓存可能发生在特定的缓存服务器（ISP/local area network维护），或者用户的计算机。DNS 信息留存在操作系统 DNS 缓存中（在windows中就是 DNS Client Serve ）。大多浏览器有自己的缓存，独立于操作系统缓存。只要浏览器在自己的缓存里有某条DNS 记录，它就不会向操作系统发 DNS 解析请求。

IE默认缓存 DNS 记录30分钟，FireFox 默认缓存1分钟。

当客户端的 DNS 缓存是空的，DNS 查找次数等于页面中的唯一域名数。

减少DNS请求数可能会减少并行下载数。避免 DNS 查找减少响应时间，但减少并行下载数可能会增加响应时间。指导原则是组件可以分散在至少2个但不多于4个的不同域名。这是这两者的一个平衡点。

Avoid Redirects(避免重定向)

跳转用301或302状态码来达成。一个301响应 http 头的例子：

```
HTTP/1.1 301 Moved Permanently
Location: http://example.com/newuri
Content-Type: text/html
```

浏览器自动跳转到 Location 指定的路径。跳转所需的所有信息都在 http 头，所以 http 主体一般是空的。301 302响应一般不会被缓存，除非有额外的头部信息，比如 Expires 或 Cache-Control 指定要缓存。meta 刷新标签或 JavaScript 也可以跳转，但如果真要跳转，3xx跳转更好，主要是保证返回键可用。

要记住的主要事情是重定向会降低用户体验。在用户和 HTML 文档之间插入重定向会延迟页面中的所有内容，因为页面中的任何内容都无法呈现，并且在 HTML 文档到达之前不会开始下载任何组件。

最浪费的跳转之一发生在url尾部斜杠（/）缺失。比如http://astrology.yahoo.com/astrology会301跳转到http://astrology.yahoo.com/astrology/。这可以被Apache等服务器修复，如果您使用的是Apache处理程序，则可以使用Aliasor mod_rewrite或DirectorySlash指令在Apache中修复此问题。

Make Ajax Cacheable(让Ajax可缓存)

提高ajax性能的最重要的方法是让响应被缓存，即在 Add an Expires or a Cache-Control Header中讨论的 Expires 。其它方法是：

- gzip 组件
- 减少 DNS 查找
- 压缩 JS
- 避免跳转
- 设置 ETags

Post-load Components(延迟加载组件)

再看看你的页面然后问问自己，“什么是页面初始化必须的？”。剩下的内容和组件可以延迟。

JavaScript 是理想的（延迟）候选者，可以切分到 onload 事件之前和之后。比如拖放的 js 库可以延迟，因为拖动必须在页面初始化之后。其它可延迟的包括隐藏的内容，折叠起来的图片等等。

> Tips：所有文章中提到的组件泛指各种文件，如：HTML、CSS、JavaScript、图片、视频、音频等等。

Preload Components(预加载组件)

预加载看起来与延迟加载相反，但它的确有个不同的目标。通过预加载你可以利用浏览器的空闲时间来请求你将来会用到的组件。这样当用户访问下一个页面时，你会有更多的组件已经在缓存中，这样会极大加快页面加载。

有几种预加载类型：

- 无条件预加载：一旦 onload 触发，你立即获取另外的组件。比如谷歌会在主页这样加载搜索结果页面用到的雪碧图。
- 有条件预加载：基于用户动作，你推测用户下一步会去哪里并加载相应组件。
- 预期的预加载：在发布重新设计的网站前提前加载。在旧网页预加载新网页的部分组件，那么切换到新网页时就不会是没有任何缓存了。

Reduce the Number of DOM Elements(减少dom数)

一个复杂的页面意味着更多的内容要下载，以及更慢的 dom 访问。比如在有 500dom 数量的页面添加事件处理就和有 5000dom 数量的不同。

如果你的页面 dom 元素很多，那么意味着你可能需要删除无用的内容和标签来优化。

Split Components Across Domains(把组件分散到不同的域名)

把组件分散到不同的域名允许你最大化并行下载数。由于 DNS 查询的副作用，最佳的不同域名数是2-4。

Minimize the Number of iframes(最小化iframe的数量)

<iframe>优点：

- 帮助解决缓慢的第三方内容的加载，如广告和徽章 安全沙盒
- 并行下载脚本

<iframe>缺点：

- 即使空的也消耗（资源和时间）
- 阻塞了页面的onload
- 非语义化（标签）

No 404s(不要404)

http 请求是昂贵的，所以发出 http 请求但获得没用的响应（如404）是完全不必要的，并且会降低用户体验。

一些网站会有特别的 404 页面提高用户体验，但这仍然会浪费服务器资源。特别坏的是当链接指向外部 js 但却得到 404 结果。这样首先会降低（占用）并行下载数，其次浏览器可能会把 404 响应体当作 js 来解析，试图从里面找出可用的东西。

小结

到此，我们把雅虎军规当中的所有内容都过了一遍。虽然随着 Vue、React、Angular 这些框架的流行，雅虎军规当中一些优化的方法已不在适用，但是，更多的时候，我们是学习这些方法背后的思想，而不是去死记硬背某个方法，技术总是在更迭，只有掌握方法背后的思想，才能够立于不败之地。

← 2 解读雅虎35条军规（上） 4 你要不要看这些优化指标? →

精选留言 4

欢迎在这里发表留言，作者筛选后可公开显示

__innocence

使用base64的图片会减少HTTP请求，但同时也会增加文件大小吧。base64比原图大了将近1/3，所以一般的大体积图片还是不使用base64

0 回复 2019-11-21

K21vin

请问没有404的话，url又没命中路由，是需要重定向跳转到别的页面而不是返回404页面吗?

0 回复 2019-10-22

XYR001

目录

dguanjie 回复 **XYR001**

我也有这个疑问

回复 2019-09-25 21:02:34

撒科打诨

看了现在更新的内容了，老师讲的很细致，期待以后的内容

0 回复 2019-07-19

千学不如一看，千看不如一练