著作权归作者所有。

商业转载请联系作者获得授权,非商业转载请注明出处。

链接:http://caibaojian.com/browser-cache.html

来源:http://caibaojian.com

什么是浏览器缓存

浏览器缓存(Brower Caching)是浏览器在本地磁盘对用户最近请求过的文档进行存储，当访问者再次访问同一页面时，浏览器就可以直接从本地磁盘加载文档。

浏览器缓存的优点有：

减少了冗余的数据传输，节省了网费

减少了服务器的负担，大大提升了网站的性能

加快了客户端加载网页的速度

在前端开发面试中，浏览器缓存是web性能优化面试题中很重要的一个知识点，从而说明浏览器缓存是提升web性能的一大利器，但是浏览器缓存如果使用不当，也会产生很多问题，正所谓是，想说爱你，并不是很容易的事。所以，结合最近遇到的案例，本文对浏览器缓存相关的知识进行总结归纳，希望对读者有所帮助。

浏览器缓存的分类

浏览器缓存主要有两类：缓存协商和彻底缓存，也有称之为协商缓存和强缓存。

浏览器在第一次请求发生后，再次请求时：

浏览器会先获取该资源缓存的header信息，根据其中的expires和cahe-control判断是否命中强缓存，若命中则直接从缓存中获取资源，包括缓存的header信息，本次请求不会与服务器进行通信；

如果没有命中强缓存，浏览器会发送请求到服务器，该请求会携带第一次请求返回的有关缓存的header字段信息（Last-Modified/IF-Modified-Since、Etag/IF-None-Match）,由服务器根据请求中的相关header信息来对比结果是否命中协商缓存，若命中，则服务器返回新的响应header信息更新缓存中的对应header信息，但是并不返回资源内容，它会告知浏览器可以直接从缓存获取；否则返回最新的资源内容

**强缓存**

强缓存是利用http的返回头中的Expires或者Cache-Control两个字段来控制的，用来表示资源的缓存时间。

**Expires**

该字段是http1.0时的规范，它的值为一个绝对时间的GMT格式的时间字符串，比如Expires:Mon,18 Oct 2066 23:59:59 GMT。这个时间代表着这个资源的失效时间，在此时间之前，即命中缓存。这种方式有一个明显的缺点，由于失效时间是一个绝对时间，所以当服务器与客户端时间偏差较大时，就会导致缓存混乱。

**Cache-Control**

Cache-Control是http1.1时出现的header信息，主要是利用该字段的max-age值来进行判断，它是一个相对时间，例如Cache-Control:max-age=3600，代表着资源的有效期是3600秒。cache-control除了该字段外，还有下面几个比较常用的设置值：

1、no-cache：不使用本地缓存。需要使用缓存协商，先与服务器确认返回的响应是否被更改，如果之前的响应中存在ETag，那么请求的时候会与服务端验证，如果资源未被更改，则可以避免重新下载。

2、no-store：直接禁止游览器缓存数据，每次用户请求该资源，都会向服务器发送一个请求，每次都会下载完整的资源。

3、public：可以被所有的用户缓存，包括终端用户和CDN等中间代理服务器。

4、private：只能被终端用户的浏览器缓存，不允许CDN等中继缓存服务器对其缓存。

Cache-Control与Expires可以在服务端配置同时启用，同时启用的时候**Cache-Control优先级高**。

**协商缓存**

协商缓存就是由服务器来确定缓存资源是否可用，所以客户端与服务器端要通过某种标识来进行通信，从而让服务器判断请求资源是否可以缓存访问，这主要涉及到下面两组header字段，这两组搭档都是成对出现的，即第一次请求的响应头带上某个字段（Last-Modified或者Etag），则后续请求则会带上对应的请求字段（If-Modified-Since或者If-None-Match），若响应头没有Last-Modified或者Etag字段，则请求头也不会有对应的字段。

**Last-Modify/If-Modify-Since**

浏览器第一次请求一个资源的时候，服务器返回的header中会加上Last-Modify，Last-modify是一个时间标识该资源的最后修改时间，例如Last-Modify: Thu,31 Dec 2037 23:59:59 GMT。

当浏览器再次请求该资源时，request的请求头中会包含If-Modify-Since，该值为缓存之前返回的Last-Modify。服务器收到If-Modify-Since后，根据资源的最后修改时间判断是否命中缓存。

**如果命中缓存，则返回304，并且不会返回资源内容，并且不会返回Last-Modify。**

**ETag/If-None-Match**

与Last-Modify/If-Modify-Since不同的是，Etag/If-None-Match返回的是一个校验码。ETag可以保证每一个资源是唯一的，资源变化都会导致ETag变化。服务器根据浏览器上送的If-None-Match值来判断是否命中缓存。

与Last-Modified不一样的是，当服务器返回304 Not Modified的响应时，由于ETag重新生成过，response header中还会把这个ETag返回，即使这个ETag跟之前的没有变化。

**为什么要有Etag**

你可能会觉得使用Last-Modified已经足以让浏览器知道本地的缓存副本是否足够新，为什么还需要Etag呢？HTTP1.1中Etag的出现主要是为了解决几个Last-Modified比较难解决的问题：

一些文件也许会周期性的更改，但是他的内容并不改变(仅仅改变的修改时间)，这个时候我们并不希望客户端认为这个文件被修改了，而重新GET；

某些文件修改非常频繁，比如在秒以下的时间内进行修改，(比方说1s内修改了N次)，If-Modified-Since能检查到的粒度是s级的，这种修改无法判断(或者说UNIX记录MTIME只能精确到秒)；

某些服务器不能精确的得到文件的最后修改时间。

Last-Modified与ETag是可以一起使用的，服务器会优先验证ETag，一致的情况下，才会继续比对Last-Modified，最后才决定是否返回304。

强缓存与协商缓存的区别可以用下表来表示：



**实际问题分析**

代码更新到线上后用户浏览器不能自行更新，我们不能要求客户在系统更新后都进行一次缓存清理的操作。

到底该如何解决呢？

在资源请求的URL中增加一个参数，比如：js/mian.js?ver=0.7.1。这个参数是一个版本号，每一次部署的时候变更一下，当这个参数变化的时候，强缓存都会失效并重新加载。这样一来，静态资源，部署以后就需要重新加载。这样就比较完美的解决了问题。