

20 | 生鲜速递：HTTP的缓存控制

2019-07-12 Chrono

《透视HTTP协议》

课程介绍 >



讲述：Chrono

时长 11:10 大小 12.79M



缓存（Cache）是计算机领域里的一个重要概念，是优化系统性能的利器。

由于链路漫长，网络时延不可控，浏览器使用 HTTP 获取资源的成本较高。所以，非常有必要把“来之不易”的数据缓存起来，下次再请求的时候尽可能地复用。这样，就可以避免多次请求 - 应答的通信成本，节约网络带宽，也可以加快响应速度。

试想一下，如果有几十 K 甚至几十 M 的数据，不是从网络而是从本地磁盘获取，那将是多么大的一笔节省，免去多少等待的时间。

领资料

实际上，HTTP 传输的每一个环节基本上都会有缓存，非常复杂。

基于“请求 - 应答”模式的特点，可以大致分为客户端缓存和服务端缓存，因为服务器端缓存经常与代理服务“混搭”在一起，所以今天我先讲客户端——也就是浏览器的缓存。



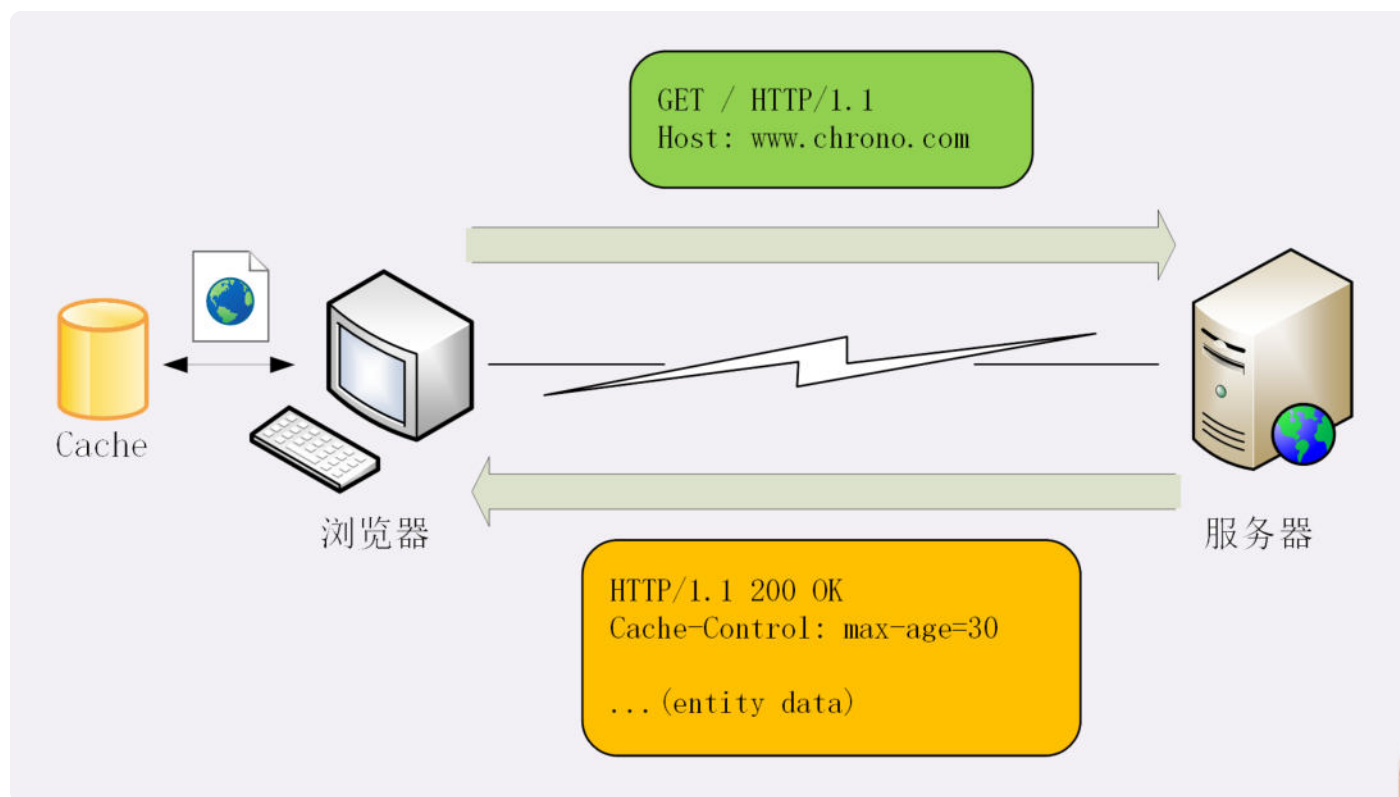
服务器的缓存控制

为了更好地说明缓存的运行机制，下面我用“生鲜速递”作为比喻，看看缓存是如何工作的。

夏天到了，天气很热。你想吃西瓜消暑，于是打开冰箱，但很不巧，冰箱是空的。不过没事，现在物流很发达，给生鲜超市打个电话，不一会儿，就给你送来一个 8 斤的沙瓤大西瓜，上面还贴着标签：“保鲜期 5 天”。好了，你把它放进冰箱，想吃的时候随时拿出来。

在这个场景里，“生鲜超市”就是 Web 服务器，“你”就是浏览器，“冰箱”就是浏览器内部的缓存。整个流程翻译成 HTTP 就是：

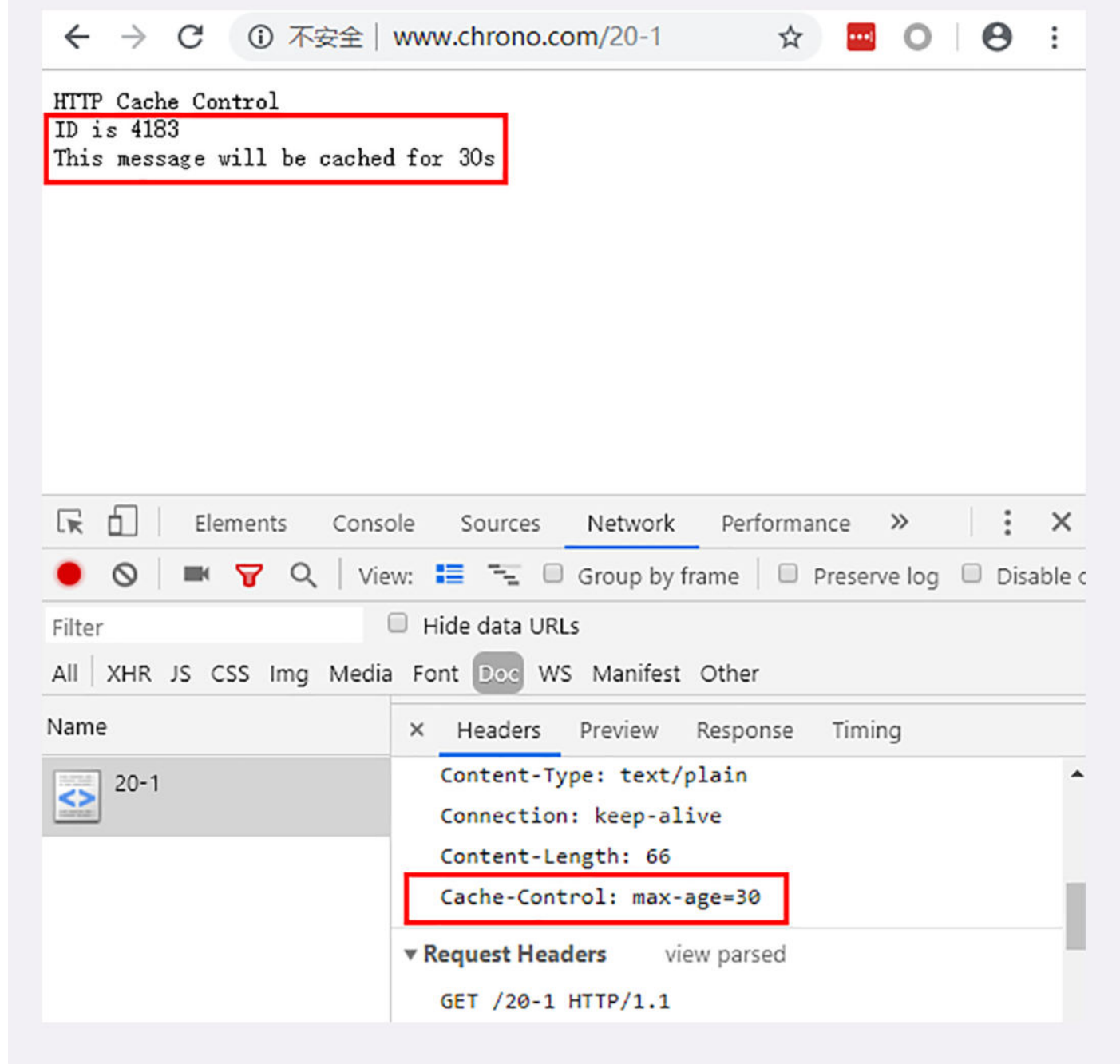
1. 浏览器发现缓存无数据，于是发送请求，向服务器获取资源；
2. 服务器响应请求，返回资源，同时标记资源的有效期；
3. 浏览器缓存资源，等待下次重用。



你可以访问实验环境的 URI “/20-1”，看看具体的请求 - 应答过程。

领资料





服务器标记资源有效期使用的头字段是“**Cache-Control**”，里面的值“**max-age=30**”就是资源的有效时间，相当于告诉浏览器，“这个页面只能缓存 30 秒，之后就算是过期，不能用。”

你可能要问了，让浏览器直接缓存数据就好了，为什么要加个有效期呢？

这是因为网络上的数据随时都在变化，不能保证它稍后的一段时间还是原来的样子。就像生鲜超市给你快递的西瓜，只有 5 天的保鲜期，过了这个期限最好还是别吃，不然可能会闹肚子。

“Cache-Control”字段里的“max-age”和上一讲里 Cookie 有点像，都是标记资源的有效期。

领资料



但我必须提醒你注意，这里的 max-age 是“生存时间”（又叫“新鲜度”“缓存寿命”，类似 TTL，Time-To-Live），时间的计算起点是响应报文的创建时刻（即 Date 字段，也就是离开服务器的时刻），而不是客户端收到报文的时刻，也就是说包含了在链路传输过程中所有节点所停留的时间。

比如，服务器设定“max-age=5”，但因为网络质量很糟糕，等浏览器收到响应报文已经过去了 4 秒，那么这个资源在客户端就最多能够再存 1 秒钟，之后就会失效。

“max-age”是 HTTP 缓存控制最常用的属性，此外在响应报文里还可以用其他的属性来更精确地指示浏览器应该如何使用缓存：

- no-store：不允许缓存，用于某些变化非常频繁的数据，例如秒杀页面；
- no-cache：它的字面含义容易与 no-store 搞混，实际的意思并不是不允许缓存，而是可以缓存，但在使用之前必须要去服务器验证是否过期，是否有最新的版本；
- must-revalidate：又是一个和 no-cache 相似的词，它的意思是如果缓存不过期就可以继续使用，但过期了如果还想用就必须去服务器验证。

听的有点糊涂吧。没关系，我拿生鲜速递来举例说明一下：

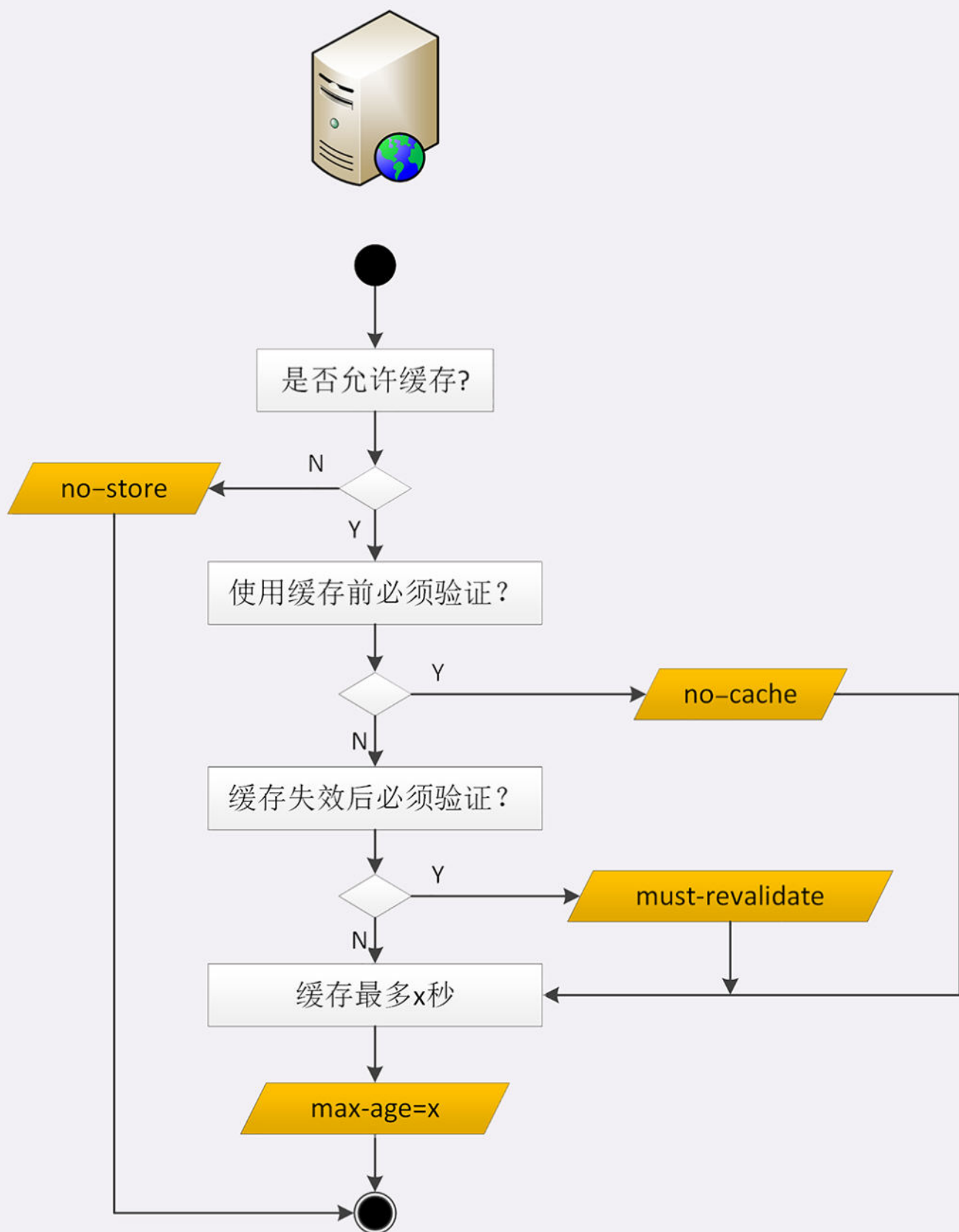
- no-store：买来的西瓜不允许放进冰箱，要么立刻吃，要么立刻扔掉；
- no-cache：可以放进冰箱，但吃之前必须问超市有没有更新鲜的，有就吃超市里的；
- must-revalidate：可以放进冰箱，保鲜期内可以吃，过期了就要问超市让不让吃。

你看，这超市管的还真多啊，西瓜到了家里怎么吃还得听他。不过没办法，在 HTTP 协议里服务器就是这样的“霸气”。

我把服务器的缓存控制策略画了一个流程图，对照着它你就可以在今后的后台开发里明确“Cache-Control”的用法了。

领资料





领资料

客户端的缓存控制

现在冰箱里已经有了“缓存”的西瓜，是不是可以直接开吃了呢？

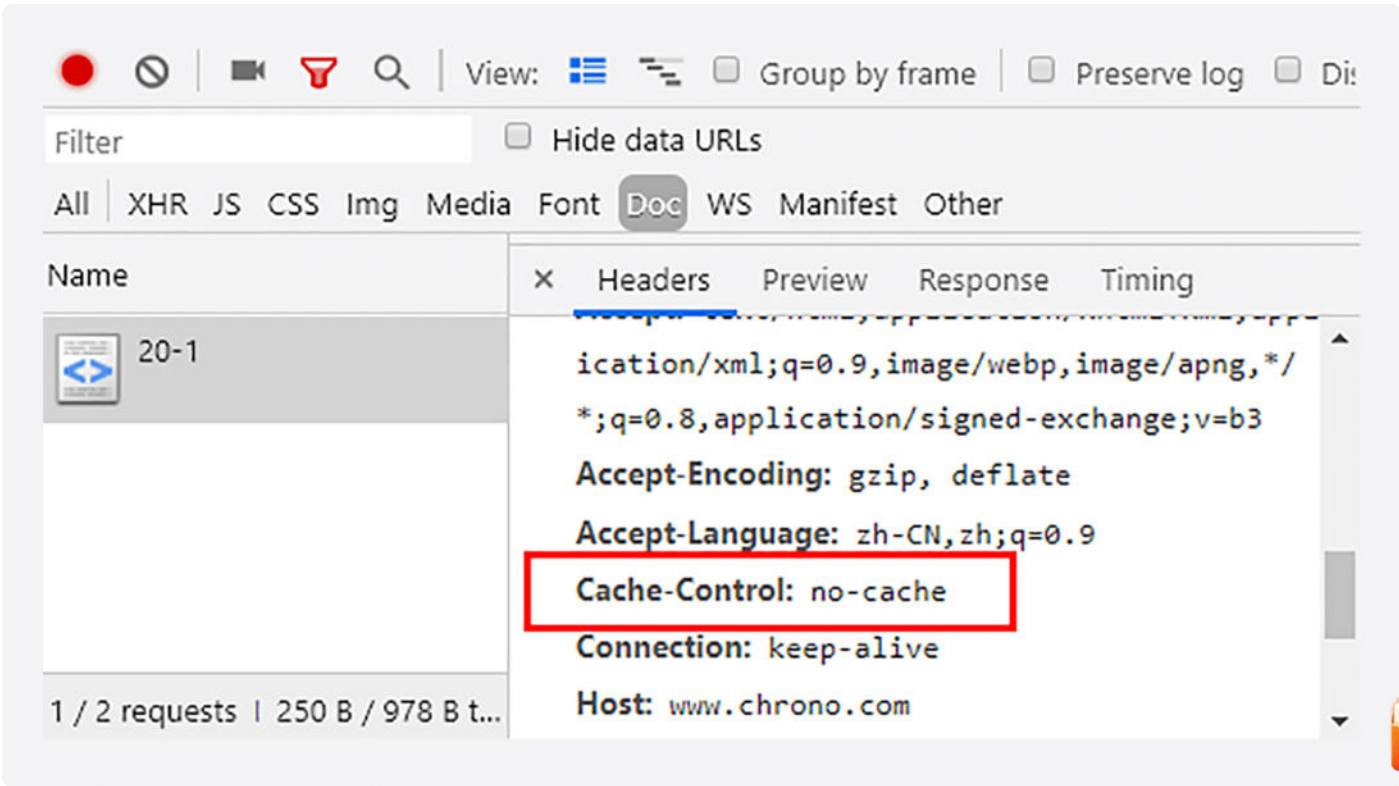
你可以在 Chrome 里点几次“刷新”按钮，估计你会失望，页面上的 ID 一直在变，根本不是缓存的结果，明明说缓存 30 秒，怎么就不起作用呢？

其实不止服务器可以发“Cache-Control”头，浏览器也可以发“Cache-Control”，也就是说请求 - 应答的双方都可以用这个字段进行缓存控制，互相协商缓存的使用策略。

当你点“刷新”按钮的时候，浏览器会在请求头里加一个“**Cache-Control: max-age=0**”。因为 max-age 是“生存时间”，max-age=0 的意思就是“我要一个最最新鲜的西瓜”，而本地缓存里的数据至少保存了几秒钟，所以浏览器就不会使用缓存，而是向服务器发请求。服务器看到 max-age=0，也就会用一个最新生成的报文回应浏览器。

Ctrl+F5 的“强制刷新”又是什么样的呢？


它其实是发了一个“**Cache-Control: no-cache**”，含义和“max-age=0”基本一样，就看后台的服务器怎么理解，通常两者的效果是相同的。



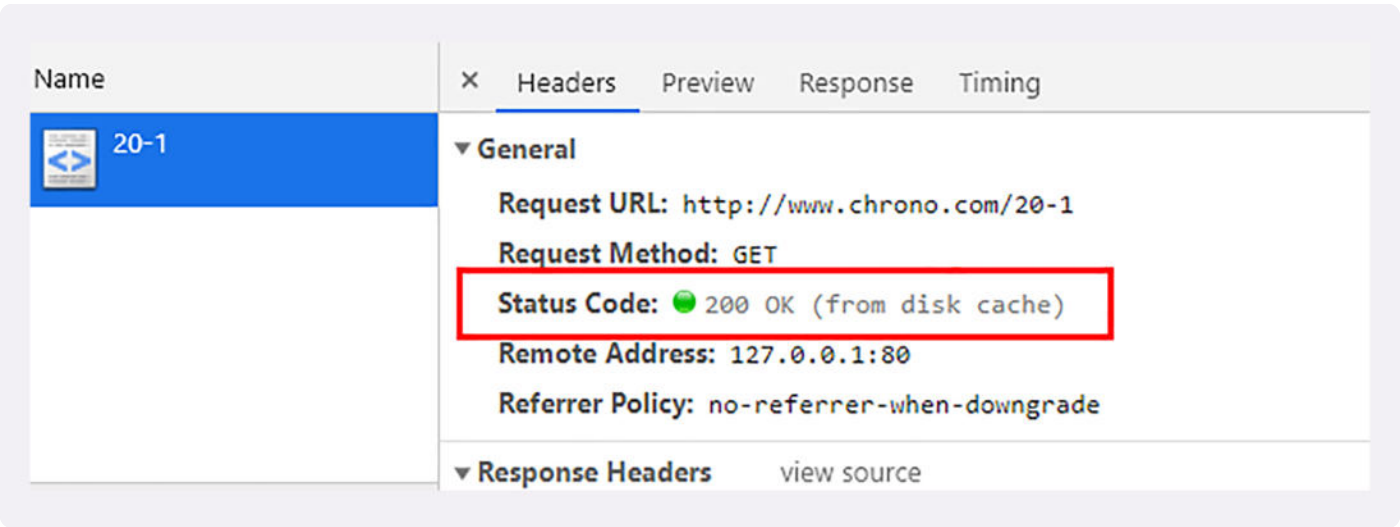
那么，浏览器的缓存究竟什么时候才能生效呢？

别着急，试着点一下浏览器的“前进”“后退”按钮，再看开发者工具，你就会惊喜地发现“from disk cache”的字样，意思是没有发送网络请求，而是读取的磁盘上的缓存。

另外，如果用 [第 18 讲](#) 里的重定向跳转功能，也可以发现浏览器使用了缓存：

 复制代码

```
1 http://www.chrono.com/18-1?dst=20-1
```



这几个操作与刷新有什么区别呢？

其实也很简单，在“前进”“后退”“跳转”这些重定向动作中浏览器不会“夹带私货”，只用最基本的请求头，没有“Cache-Control”，所以就会检查缓存，直接利用之前的资源，不再进行网络通信。

这个过程你也可以用 Wireshark 抓包，看看是否真的没有向服务器发请求。

条件请求

浏览器用“Cache-Control”做缓存控制只能是刷新数据，不能很好地利用缓存数据，又因为缓存会失效，使用前还必须要去服务器验证是否是最新版。

那么该怎么做呢？

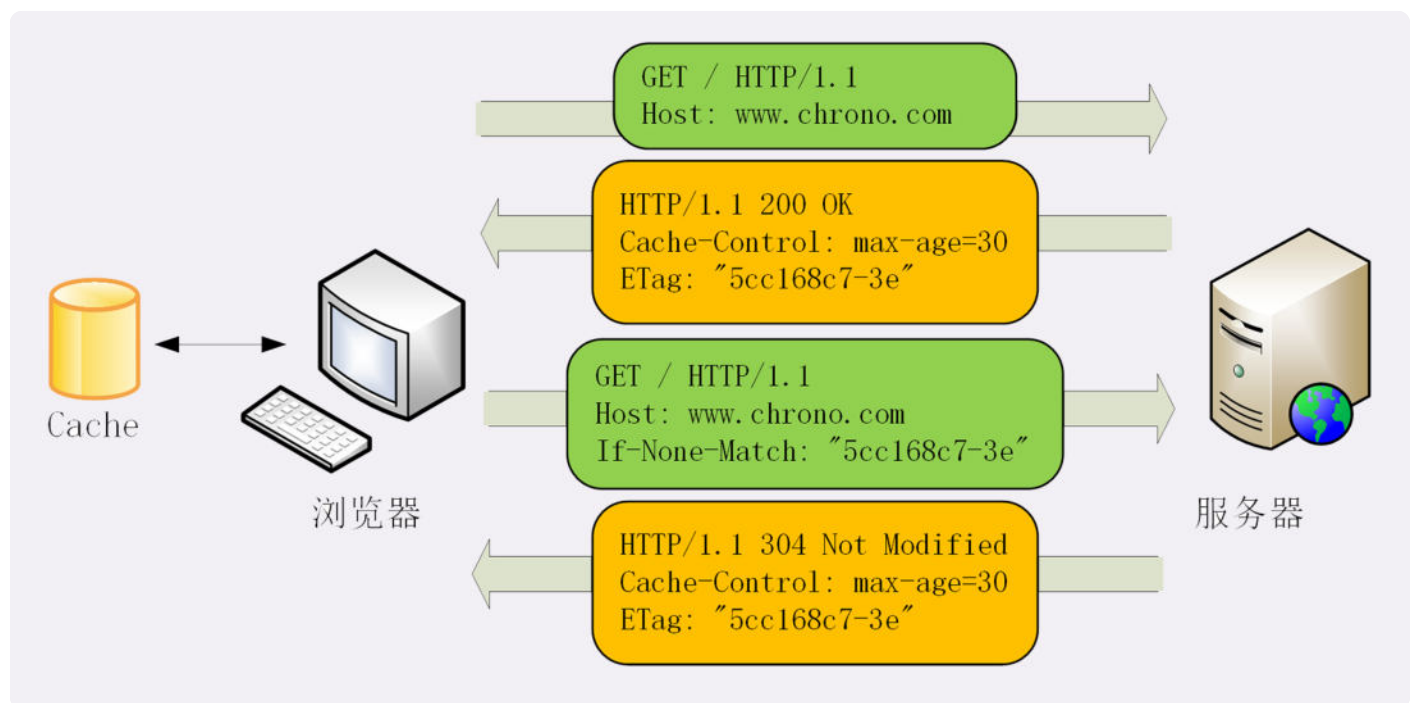
浏览器可以用两个连续请求组成“验证动作”：先是一个 HEAD，获取资源的修改时间等元信息，然后与缓存数据比较，如果没有改动就使用缓存，节省网络流量，否则就再发一个 GET 请求，获取最新的版本。



但这样的两个请求网络成本太高了，所以 HTTP 协议就定义了一系列“if”开头的“**条件请求**”字段，专门用来检查验证资源是否过期，把两个请求才能完成的工作合并在一个请求里做。而且，验证的责任也交给服务器，浏览器只需“坐享其成”。

条件请求一共有 5 个头字段，我们最常用的是“**if-Modified-Since**”和“**If-None-Match**”这两个。需要第一次的响应报文预先提供“**Last-modified**”和“**ETag**”，然后第二次请求时就可以带上缓存里的原值，验证资源是否是最新的。

如果资源没有变，服务器就回应一个“**304 Not Modified**”，表示缓存依然有效，浏览器就可以更新一下有效期，然后放心大胆地使用缓存了。



“Last-modified”很好理解，就是文件的最后修改时间。ETag 是什么呢？

ETag 是“实体标签”（Entity Tag）的缩写，**是资源的一个唯一标识**，主要是用来解决修改时间无法准确区分文件变化的问题。

比如，一个文件在一秒内修改了多次，但因为修改时间是秒级，所以这一秒内的新版本无法区分。

再比如，一个文件定期更新，但有时会是同样的内容，实际上没有变化，用修改时间就会误以为发生了变化，传送给浏览器就会浪费带宽。



使用 ETag 就可以精确地识别资源的变动情况，让浏览器能够更有效地利用缓存。

ETag 还有“强”“弱”之分。

强 ETag 要求资源在字节级别必须完全相符，弱 ETag 在值前有个“W/”标记，只要求资源在语义上没有变化，但内部可能会有部分发生了改变（例如 HTML 里的标签顺序调整，或者多了几个空格）。

还是拿生鲜速递做比喻最容易理解：

你打电话给超市，“我这个西瓜是 3 天前买的，还有最新的吗？”。超市看了一下库存，说：“没有啊，我这里都是 3 天前的。”于是你就知道了，再让超市送货也没用，还是吃冰箱里的西瓜吧。这就是“**if-Modified-Since**”和“**Last-modified**”。

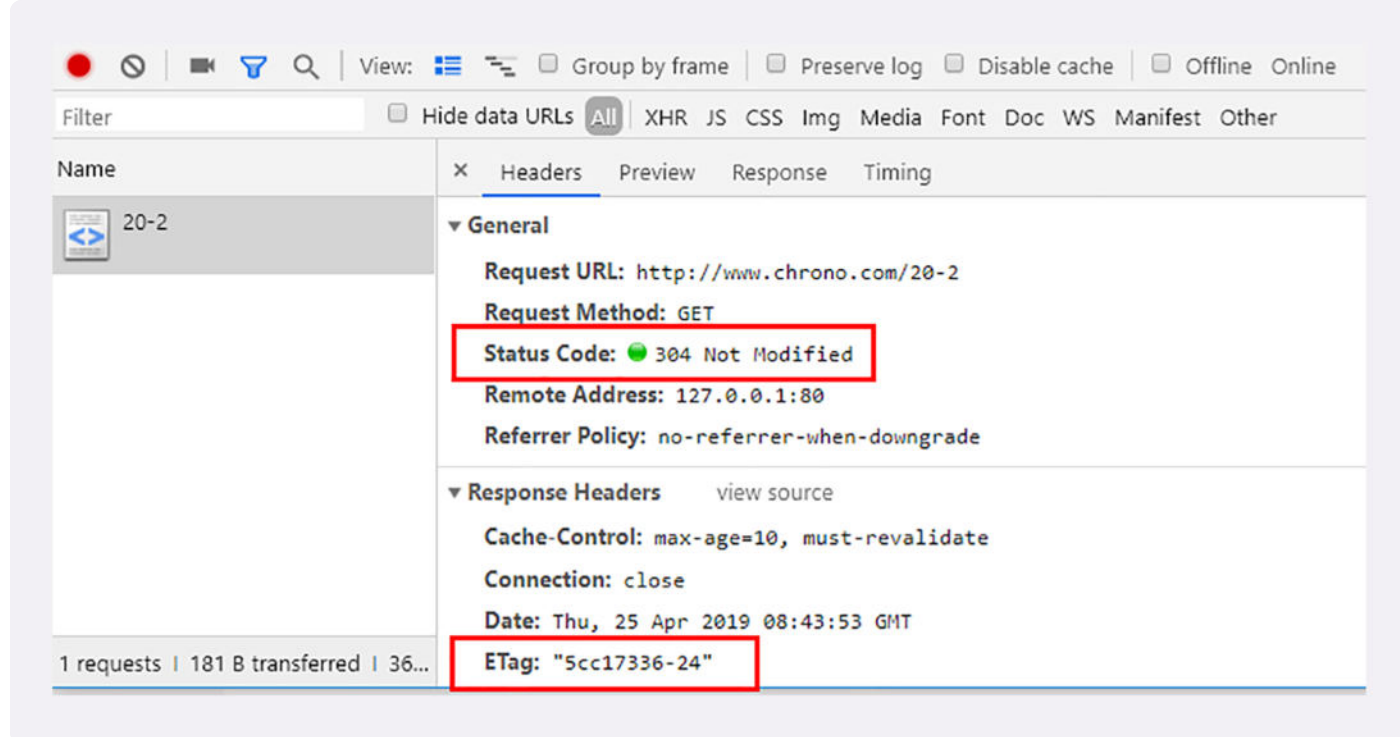
但你还是想要最新的，就又打电话：“有不是沙瓤的西瓜吗？”，超市告诉你都是沙瓤的（Match），于是你还是只能吃冰箱里的沙瓤西瓜。这就是“**If-None-Match**”和“**弱 ETag**”。

第三次打电话，你说“有不是 8 斤的沙瓤西瓜吗？”，这回超市给了你满意的答复：“有个 10 斤的沙瓤西瓜”。于是，你就扔掉了冰箱里的存货，让超市重新送了一个新的大西瓜。这就是“**If-None-Match**”和“**强 ETag**”。

再来看看实验环境的 URI “/20-2”。它为资源增加了 ETag 字段，刷新页面时浏览器就会同时发送缓存控制头“max-age=0”和条件请求头“If-None-Match”，如果缓存有效服务器就会返回 304：

领资料





条件请求里其他的三个头字段是“`If-Unmodified-Since`”“`If-Match`”和“`If-Range`”，其实只要你掌握了“`if-Modified-Since`”和“`If-None-Match`”，可以轻易地“举一反三”。

小结

今天我们学习了 HTTP 的缓存控制和条件请求，用好它们可以减少响应时间、节约网络流量，一起小结一下今天的内容吧：

1. 缓存是优化系统性能的重要手段，HTTP 传输的每一个环节中都可以有缓存；
2. 服务器使用“`Cache-Control`”设置缓存策略，常用的是“`max-age`”，表示资源的有效期；
3. 浏览器收到数据就会存入缓存，如果没过期就可以直接使用，过期就要去服务器验证是否仍然可用；
4. 验证资源是否失效需要使用“条件请求”，常用的是“`if-Modified-Since`”和“`If-None-Match`”，收到 304 就可以复用缓存里的资源；
5. 验证资源是否被修改的条件有两个：“`Last-modified`”和“`ETag`”，需要服务器预先在响应报文里设置，搭配条件请求使用；
6. 浏览器也可以发送“`Cache-Control`”字段，使用“`max-age=0`”或“`no_cache`”刷新数据。

HTTP 缓存看上去很复杂，但基本原理说白了就是一句话：“没有消息就是好消息”，“没有请求的请求，才是最快的请求。”



课下作业

1. Cache 和 Cookie 都是服务器发给客户端并存储的数据，你能比较一下两者的异同吗？
2. 即使有“Last-modified”和“ETag”，强制刷新（Ctrl+F5）也能够从服务器获取最新数据（返回 200 而不是 304），请你在实验环境里试一下，观察请求头和响应头，解释原因。

欢迎你把自己的学习体会写在留言区，与我和其他同学一起讨论。如果你觉得有所收获，也欢迎把文章分享给你的朋友。



课外小贴士

- 01 较早版本的 Chrome（66 之前）可以使用 URL “chrome://cache” 检查本地缓存，但因为存在安全隐患，现在已经不能用了。
- 02 “no-cache” 属性可以理解为 “max-age=0,must-revalidate”。
- 03 除了 “Cache-Control”，服务器也可以用 “Expires” 字段来标记资源的有效期，它的形式和 Cookie 的差不多，同样属于 “过时” 的属性，优先级低于 “Cache-Control”。还有一个历史遗留字段 “Pragma: no-cache”，它相当于 “Cache-Control: no-cache”，除非为了

领资料




兼容 HTTP/1.0 否则不建议使用。

04 如果响应报文里提供了“Last-modified”，但没有“Cache-Control”或“Expires”，浏览器会使用“启发”（Heuristic）算法计算一个缓存时间，在 RFC 里的建议是： $(Date - Last-modified) * 10\%$ 。

05 每个 Web 服务器对 ETag 的计算方法都不一样，只要保证数据变化后值不一样就好，但复杂的计算会增加服务器的负担。Nginx 的算法是“修改时间 + 长度”，实际上和 Last-modified 基本等价。

分享给需要的人，Ta订阅超级会员，你将得 50 元

Ta单独购买本课程，你将得 20 元

 生成海报并分享

领资料

 赞 16

 提建议

© 版权归极客邦科技所有，未经许可不得传播售卖。页面已增加防盗追踪，如有侵权极客邦将依法追究其法律责任。



上一篇 19 | 让我知道你是谁：HTTP的Cookie机制

下一篇 21 | 良心中间商：HTTP的代理服务

JVM + NIO + Spring

各大厂面试题及知识点详解

限时免费



精选留言 (74)

写留言



Fstar

2019-07-13

Cache 和 Cookie 的相同点是：都会保存到浏览器中，并可以设置过期时间。

不同点：

1. Cookie 会随请求报文发送到服务器，而 Cache 不会，但可能会携带 if-Modified-Since（保存资源的最后修改时间）和 If-None-Match（保存资源唯一标识） 字段来验证资源是否过期。
2. Cookie 在浏览器可以通过脚本获取（如果 cookie 没有设置 HttpOnly），Cache 则无法在浏览器中获取（出于安全原因）。
3. Cookie 通过响应报文的 Set-Cookie 字段获得，Cache 则是位于 body 中。
4. 用途不同。Cookie 常用于身份识别，Cache 则是由浏览器管理，用于节省带宽和加快响应速度。
5. Cookie 的 max-age 是从浏览器拿到响应报文时开始计算的，而 Cache 的 max-age 是从响应报文的生成时间（Date 头字段）开始计算。

领资料

作者回复：总结的非常好。

第三点感觉有点问题，cache缓存的是完整的报文，不单单是body。



一步

2019-07-13

对于第二个问题：发现强制刷新后请求头中 没有了 If-None-Match，而且 Cache-Control: no-cache

是这个原因吗？

作者回复: 对，没有条件请求头，那么服务器就无法处理缓存，就只能返回最新的数据。

共 2 条评论 >

18



小鸟淫太

2019-07-12

1. cookie是方便进行身份识，cache是为了减少网络请求。
2. 强制刷新是因为请求头里的 If-Modified-Since 和 If-None-Match 会被清空所以会返回最新数据

作者回复: 回答正确，之前是我弄错了。

共 2 条评论 >

17



院长。

2019-07-15

老师我有几个问题想问一下：

1. F5刷新的时候，请求头加上"Cache-Control: max-age=0"，您文章里说，服务器用一个最新生成的报文回应浏览器，那这时候响应返回的应该是"200 OK"吗？为什么我在极客网页版的这个页面刷新后，有个叫"106804"的资源返回的是"304"，但是强制刷新是"200 ok"，产生的效果好像不同呀。这里是不是应该换一种方式说？感觉强制刷新说的有些简单了。
2. F5刷新发送的请求头是固定的吗？还是会根据浏览器不同而产生变化？
3. 200（from memory cache）和200(from disk cache)是针对内存和硬盘的，他们出现的场景分别是什么呢？
4. HTTP缓存有标准性的流程吗？比如说从我输入URL开始，到后续刷新或者强制刷新等？
- 5.对于"must-revalidate"我有疑问，本身存储机制不就是如果不过期的话可以继续使用，过期的话去请求服务器吗？那这个属性还有什么意义呢？
6. no-cache,no-store,max-age等属性可以共存吗？

问题有点多，因为网上资料质量参差不齐，解释有些也不全相同，所以在这里咨询下老师，希望老师可以解答一下，或者有推荐的讲述HTTP缓存的文章也可以，谢谢老师

作者回复: 1.强制刷新请求最新的资源，没有条件请求，所以不会有304，都是200。

2.每个浏览器可能会有不同，但基本的字段是一样的。

领资料



3.缓存的位置不一样，浏览器会分别存放到内存或者硬盘上，所以会显示来源不同。

4.http只规定了缓存的用法，具体如何存放如何使用就是客户端自己灵活实现了，怎么方便怎么来。

5.过期后去验证，如果服务器返回304，那么就可以继续重用缓存，而不用下载整个资源。

6.可以看一下流程图，不是所有的属性都能共存的。当然如果你要是都写上也不是不可以，那浏览器就会“精神错乱”了。

共 3 条评论 >

👍 15



DENG永青

2019-08-07

Etag的工作原理

Etag在服务器上生成后,客户端通过If-Match或者说If-None-Match这个条件判断请求来验证资源是否修改.我们常见的是使用If-None-Match.请求一个文件的流程可能如下:

新的请求

客户端发起HTTP GET请求一个文件(css ,image,js); 服务器处理请求,返回文件内容和一堆Header(包括Etag,例如"2e681a-6-5d044840"),http头状态码为200.

同一个用户第二次这个文件的请求

客户端在一次发起HTTP GET请求一个文件,注意这个时候客户端同时发送一个If-None-Match头,这个头中会包括上次这个文件的Etag(例如"2e681a-6-5d044840"),这时服务器判断发送过来的Etag和自己计算出来的Etag,因此If-None-Match为False,不返回200,返回304,客户端继续使用本地缓存;

注意.服务器又设置了Cache-Control:max-age和Expires时,会同时使用,也就是说在完全匹配If-Modified-Since和If-None-Match即检查完修改时间和Etag之后,服务器才能返回304.

作者回复: 写的非常详细，点赞。



👍 12



Marvin

2019-08-04

我有一个问题，就拿咱们极客时间的网页来说，会请求一个Id-00001.ts的文件，响应头中指示了cache-control: max-age= 7200，要一个小时才过期，那么为什么每次刷新都是304，像这种情况不应该直接200 cacahe from disk才对么？为什么明明没有过期还要去服务器协商呢？

领资料



作者回复: 刷新时发的是条件请求, 不是普通的请求, 所以就必须返回304, 告诉浏览器内容没有过期, 可以继续用缓存。

普通请求才会直接检查缓存, 然后是200 cacahe from disk。

共 2 条评论 >

👍 9



啦啦啦

2019-07-12

老师, nocache, 每次使用前都需要去浏览器问一下有没有过期, 这不也是一次请求吗? 那不和没有缓存一个意思吗

作者回复: 不一样, 如果服务器返回304, 是一个很小的报文, 这样浏览器就可以直接重用缓存里的数据, 可以节约传输带宽。

nostore每次都会传输完整的报文, 成本高。



👍 8



Leon

2019-07-21

小贴士的nginx计算etag我贴下测试logngx_sprintf(etag->value.data, ""%xT-%xO"", r->headers_out.last_modified_time, r->headers_out.content_length_n)相信大家看到这里更清晰明了

作者回复: great。



👍 5



Khirye

2019-11-25

Hi, 我对缓存控制策略这张流程图有一些疑惑, must-revalidate是指缓存过期之后, 必须要向服务器验证缓存, 这一步应该是在图中“缓存最多x秒”这个判断之前的吧? 因为只有缓存超过了max-age的期限, 才会进入“must-revalidate的判断”这一步吧? 烦请解惑, 谢谢!

作者回复: 这张图是“服务器”的缓存策略, 也就是说服务器应该如何设置资源的缓存参数, 并不是客户端判断缓存的流程。

只要不是no-store就必然会设置max-age, 所以must-revalidate是max-age的一个附加条件。

领资料





4

**游鱼**

2020-03-20

老师，开发提测后，有时需要清缓存才是最新的，强刷都不管用，这个有解决办法吗

作者回复: 看缓存是在哪里了，因为http的传输链路很长，可能在某个节点的缓存时间长，强制刷新不生效，需要具体分析，不好直接给解决方案。

共 2 条评论 >



3

**Maske**

2020-06-16

1.cache的作用为定义浏览器对静态文件如何进行缓存控制，目的是为了有效利用可复用的资源，尽可能减少客户端的请求，优化用户体验减轻服务器响应压力。常用字段值就那么一些，并有各自的含义。cookie的作用是增加了http请求的状态性，让服务器‘认识’当前访问的用户是谁，字段key,value值都可以自定义，比较灵活。

2.看了下天猫首页的css, js文件，普通的刷新（F5）操作中，不会在请求头中包含cache-control、if-none-match, if-Modified-Since,刷新会命中缓存文件，属于强缓存。强制刷新（ctrl + f5）在请求头中附加了cache-control: no-cache，为协商缓存，相当于设置max-age=0;所以此时不会使用本地缓存，当前页面所有的请求均是如此

作者回复: good



2

**Larry**

2020-05-30

如果响应报文什么字段都不设置，单纯的返回数据，是不是也不会缓存？

作者回复: 按协议来说，没有规定，就取决于客户端，想缓存也可以。



2

领资料**walle**

2019-11-15

cache-control 中的 private 是如何识别的呢？是根据session吗还是什么方式开识别是私有缓存呢



作者回复: 缓存策略取决于服务器, 它认为这个缓存只能存放在客户端, 不能存放在代理上, 就设置private。

与session无关。



👍 2



WL

2019-07-12

请问老师弱ETag是服务器更新时自己判断本次的更新有没有语义的变化, 如果语义有变化就重新生成一个ETag, 如果没有变化不重新生成直接使用原来的, 请问是这样的流程吗?

作者回复: 强etag和etag的流程都是一样的, 只是计算的方式不同, (即判断是否发生变化的方式不同)。

你的理解正确。



👍 2



来自地狱的勇士

2019-07-12

老师, 既然Etag的算法比较复杂, 需要占用服务器资源, 那么, 实际上服务器会使用Etag吗? 看到有的资料说服务器很少会用到Etag, 这个说法正确吗?

作者回复: Nginx和Apache都有etag, 但算法不同, 但都不会用特别消耗计算资源的算法。

其他的web服务器就不太清楚了。



👍 2



路漫漫

2021-11-16

老师, 根据服务器的缓存控制那个图, 如果cache-control 设置了 no-cache 或 must-revalidate 那就 必须设置 max-age喽?

作者回复: 按照协议要求是这样的, 不然不知道应该缓存多长时间。

如果不提供max-age, 浏览器也可以估算一个时间, 但使用max-age还是最规范的写法。



👍 1

领资料





风宇殇

2021-03-10

这篇文章将缓存讲的比较容易理解。<https://mp.weixin.qq.com/s/cUqkG3NETmJbglDXfSf0tg>

作者回复: 挺好的文章, 欢迎多来这样的分享。



1



Joe

2020-09-18

老师你好, 如果使用的是强缓存, 比如Cache-Control: max-age=36000, 那么在有效期内服务器上的文件发生了改变, 客户端怎么才能及时获取最新的文件? 更改文件指纹是可以获取最新的文件吗? 如果可以这个请求流程是什么样的?

作者回复: 使用if系列的条件请求, 用时间戳或者etag, 发给服务器, 服务器来判断, 如果没变化就可以直接重用客户端的缓存, 否则就发回新的文件。

可以再仔细看看条件请求, 关键就是客户端带上一个小的验证信息, 让服务器检查。

共 2 条评论 >

1



旅途

2020-08-08

老师, 问一下 服务器返回304, 这个服务器是怎么返回的呢? 又没有走指定的查询数据是否改变的接口

作者回复: 304应该是走条件请求才返回的状态码, 如果没有发if系列头, 服务器应该返回200才对。

不过既然服务器这么做了, 就当它是200吧, 也许它内部有什么特殊的处理流程。



1



潇潇雨歇

2020-06-14

- 1、cookie主要用于保存会话状态, 会作为字段发送给服务端, 用于身份认证。而cache是整个资源, 也就是整个报文, 不作为字段, 但是要使用缓存需要设置相应字段。
- 2、强制刷新后: Cache-Control: no-cache, 且没有If-None-Match和If-Modified-Since字段, 这样就不能命中缓存了, 所以会返回200; 而普通刷新有这些字段, Cache-Control为max-age=0, 存在If-None-Match或If-Modified-Since字段, 根据情况使用强缓存协商缓存。

领资料



作者回复: good



1

