

## 22 | 冷链周转：HTTP的缓存代理

2019-07-17 Chrono

《透视HTTP协议》

课程介绍 >



讲述：Chrono

时长 10:37 大小 12.16M



在 [第 20 讲](#) 中，我介绍了 HTTP 的缓存控制，[第 21 讲](#) 我介绍了 HTTP 的代理服务。那么，把这两者结合起来就是这节课所要说的“**缓存代理**”，也就是支持缓存控制的代理服务。

之前谈到缓存时，主要讲了客户端（浏览器）上的缓存控制，它能够减少响应时间、节约带宽，提升客户端的用户体验。

但 HTTP 传输链路上，不只是客户端有缓存，服务器上的缓存也是非常有价值的，可以让请求不必走完整个后续处理流程，“就近”获得响应结果。

特别是对于那些“读多写少”的数据，例如突发热点新闻、爆款商品的详情页，一秒钟内可能有成千上万次的请求。即使仅仅缓存数秒钟，也能够把巨大的访问流量挡在外面，让 RPS（request per second）降低好几个数量级，减轻应用服务器的并发压力，对性能的改善是非常显著的。



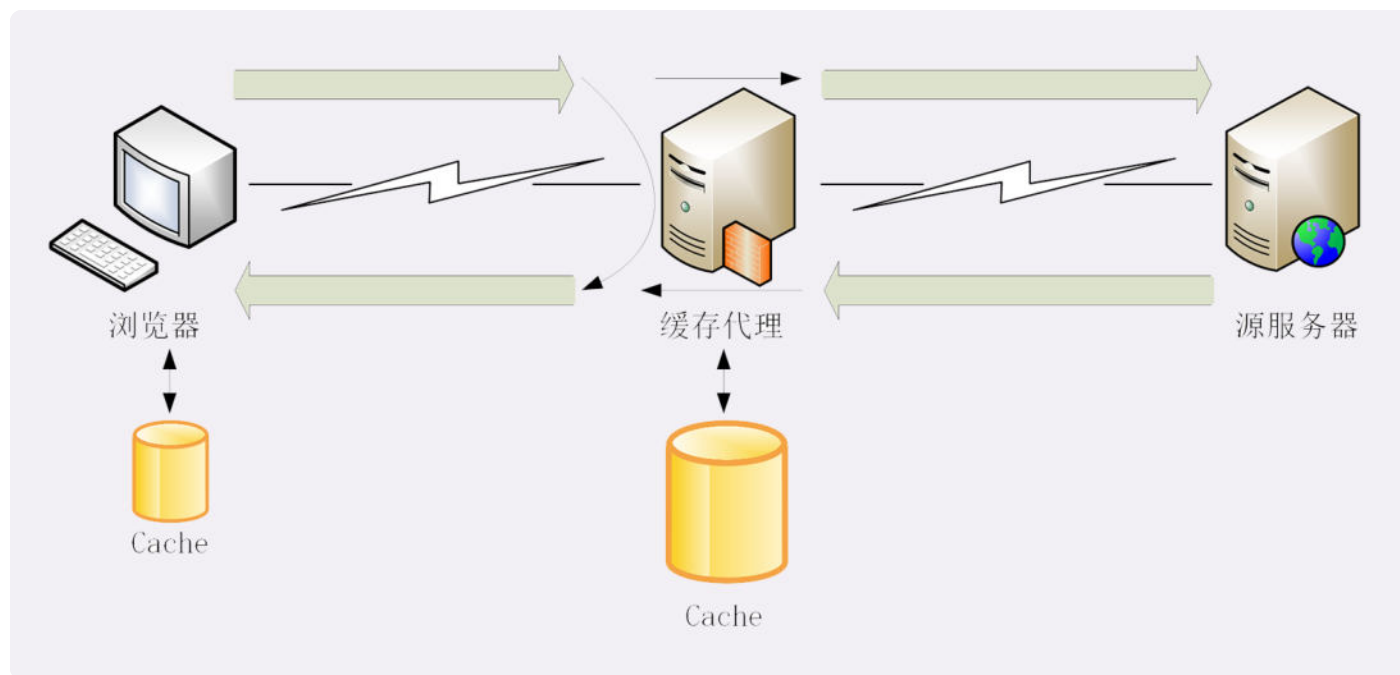
HTTP 的服务器缓存功能主要由代理服务器来实现（即缓存代理），而源服务器系统内部虽然也经常有各种缓存（如 Memcache、Redis、Varnish 等），但与 HTTP 没有太多关系，所以这里暂且不说。

## 缓存代理服务

我还是沿用“生鲜速递 + 便利店”的比喻，看看缓存代理是怎么回事。

便利店作为超市的代理，生意非常红火，顾客和超市双方都对现状非常满意。但时间一长，超市发现还有进一步提升的空间，因为每次便利店接到顾客的请求后都要专车跑一趟超市，还是挺麻烦的。

干脆这样吧，给便利店配发一个大冰柜。水果海鲜什么的都可以放在冰柜里，只要产品在保鲜期内，就允许顾客直接从冰柜提货。这样便利店就可以一次进货多次出货，省去了超市之间的运输成本。



通过这个比喻，你可以看到：在没有缓存的时候，代理服务器每次都是直接转发客户端和服务器的报文，中间不会存储任何数据，只有最简单的中转功能。

加入了缓存后就不一样了。

代理服务收到源服务器发来的响应数据后需要做两件事。第一个当然是把报文转发给客户端，而第二个就是把报文存入自己的 Cache 里。



下一次再有相同的请求，代理服务器就可以直接发送 304 或者缓存数据，不必再从源服务器那里获取。这样就降低了客户端的等待时间，同时节约了源服务器的网络带宽。

在 HTTP 的缓存体系中，缓存代理的身份十分特殊，它“既是客户端，又是服务器”，同时也“既不是客户端，又不是服务器”。

说它“即是客户端又是服务器”，是因为它面向源服务器时是客户端，在面向客户端时又是服务器，所以它即可以用客户端的缓存控制策略也可以用服务器端的缓存控制策略，也就是说它可以同时使用第 20 讲的各种“Cache-Control”属性。

但缓存代理也“即不是客户端又不是服务器”，因为它只是一个数据的“中转站”，并不是真正的数据消费者和生产者，所以还需要有一些新的“Cache-Control”属性来对它做特别的约束。

## 源服务器的缓存控制

🔗 第 20 讲介绍了 4 种服务器端的“Cache-Control”属性：max-age、no-store、no-cache 和 must-revalidate，你应该还有印象吧？

这 4 种缓存属性可以约束客户端，也可以约束代理。

但客户端和代理是不一样的，客户端的缓存只是用户自己使用，而代理的缓存可能会为非常多的客户端提供服务。所以，需要对它的缓存再多一些限制条件。

首先，我们要区分客户端上的缓存和代理上的缓存，可以使用两个新属性“private”和“public”。

“private”表示缓存只能在客户端保存，是用户“私有”的，不能放在代理上与别人共享。而“public”的意思就是缓存完全开放，谁都可以存，谁都可以用。

比如你登录论坛，返回的响应报文里用“Set-Cookie”添加了论坛 ID，这就属于私人数据，不能存在代理上。不然，别人访问代理获取了被缓存的响应就麻烦了。

其次，缓存失效后的重新验证也要区分开（即使用条件请求“Last-modified”和“ETag”），“must-revalidate”是只要过期就必须回源服务器验证，而新



的“**proxy-revalidate**”只要求代理的缓存过期后必须验证，客户端不必回源，只验证到代理这个环节就行了。

再次，缓存的生存时间可以使用新的“**s-maxage**”（s 是 share 的意思，注意 maxage 中间没有“-”），只限定在代理上能够存多久，而客户端仍然使用“max-age”。

还有一个代理专用的属性“**no-transform**”。代理有时候会对缓存下来的数据做一些优化，比如把图片生成 png、webp 等几种格式，方便今后的请求处理，而“no-transform”就会禁止这样做，不许“偷偷摸摸搞小动作”。

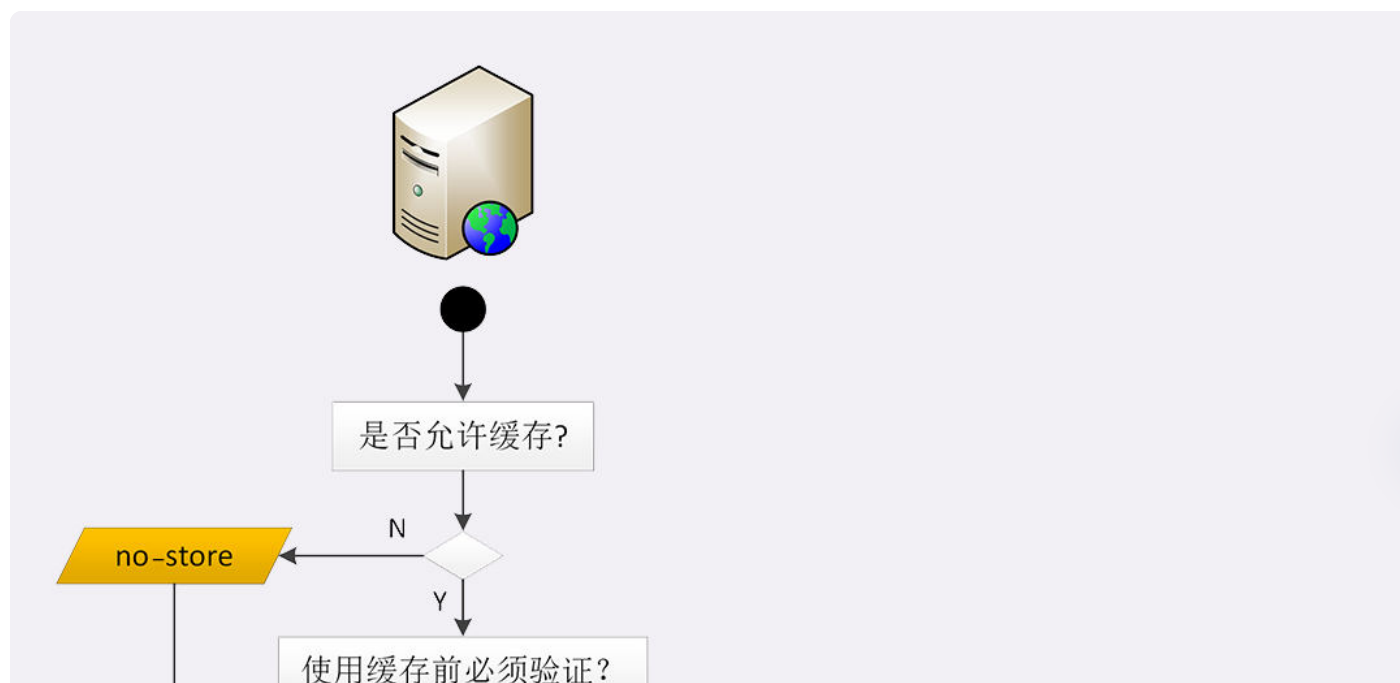
这些新的缓存控制属性比较复杂，还是用“便利店冷柜”来举例好理解一些。

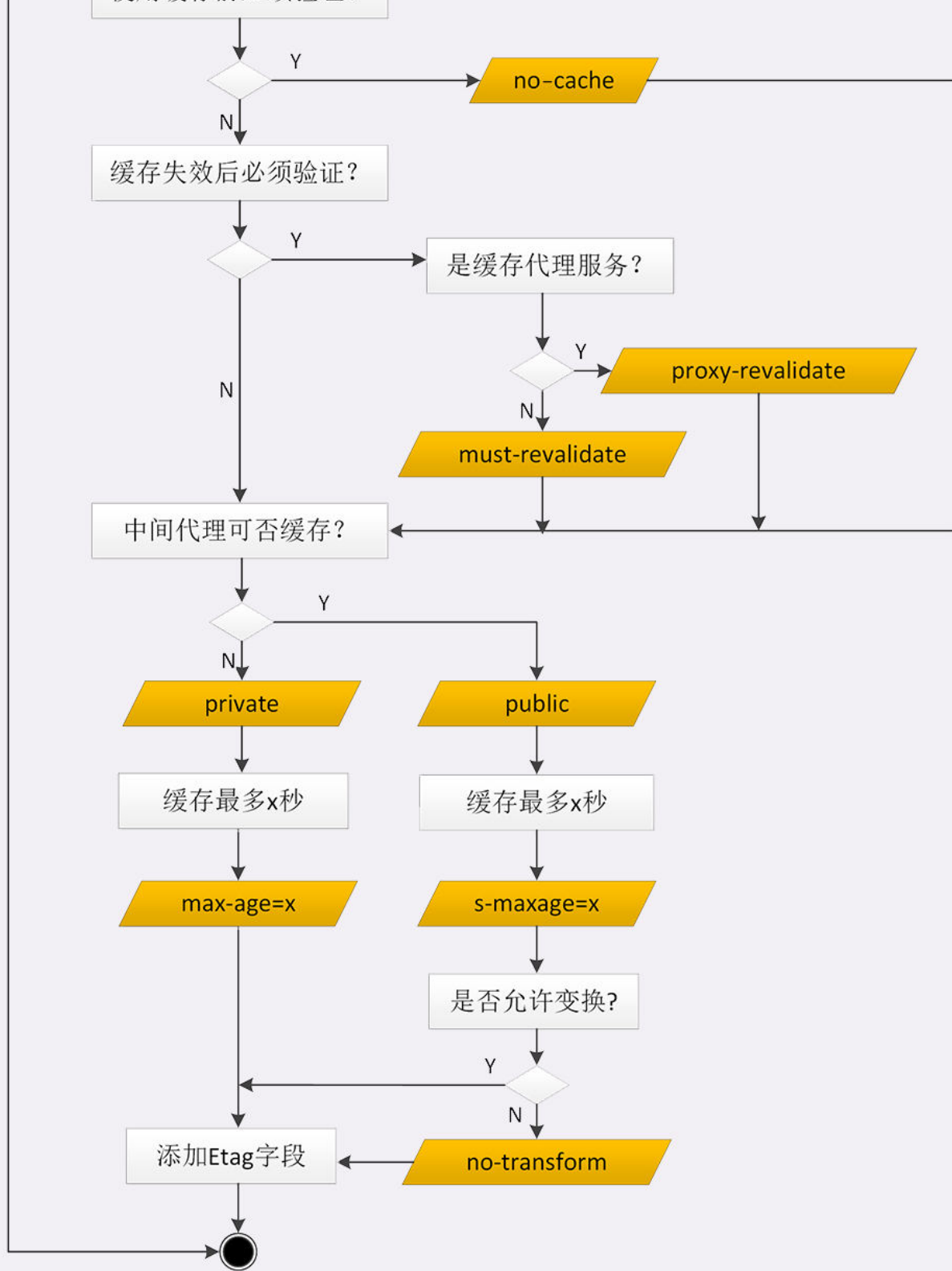
水果上贴着标签“private, max-age=5”。这就是说水果不能放进冷柜，必须直接给顾客，保鲜期 5 天，过期了还得去超市重新进货。

冻鱼上贴着标签“public, max-age=5, s-maxage=10”。这个的意思就是可以在冰柜里存 10 天，但顾客那里只能存 5 天，过期了可以来便利店取，只要在 10 天之内就不必再找超市。

排骨上贴着标签“max-age=30, proxy-revalidate, no-transform”。因为缓存默认是 public 的，那么它在便利店和顾客的冰箱里就都可以存 30 天，过期后便利店必须去超市进新货，而且不能擅自把“大排”改成“小排”。

下面的流程图是完整的服务器端缓存控制策略，可以同时控制客户端和代理。





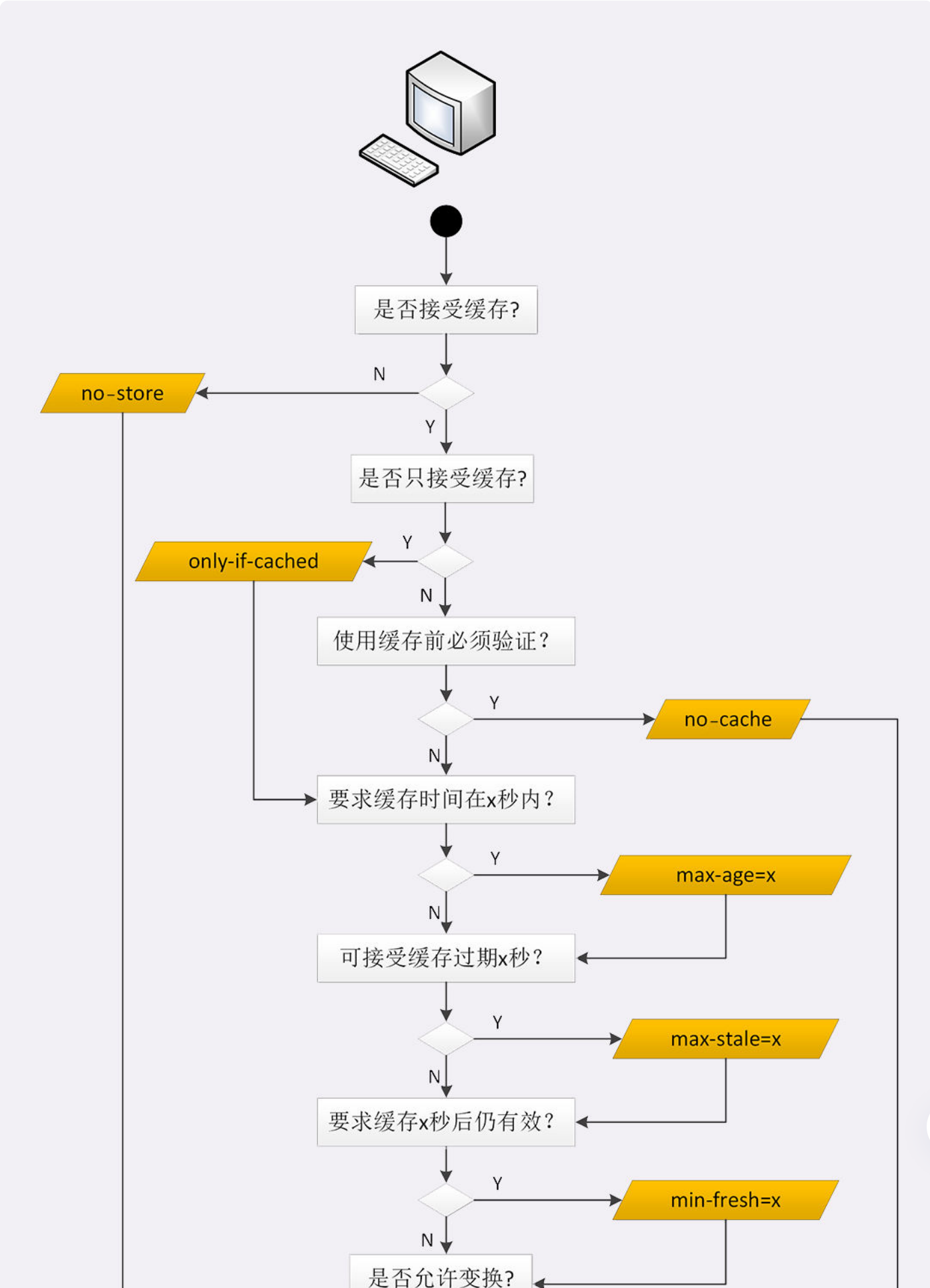
我还要提醒你一点，源服务器在设置完“Cache-Control”后必须要为报文加上“Last-modified”或“ETag”字段。否则，客户端和代理后面就无法使用条件请求来验证缓存是否有效，也就不会有 304 缓存重定向。

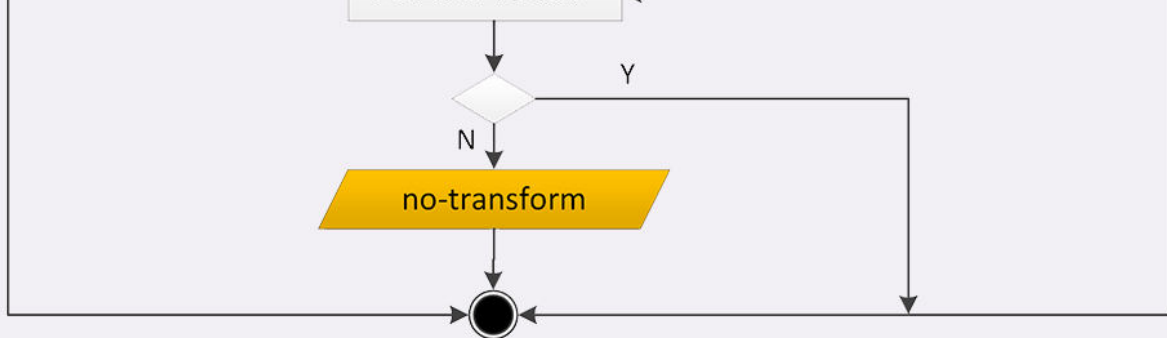
## 客户端的缓存控制

说完了服务器端的缓存控制策略，稍微歇一口气，我们再来看看客户端。



客户端在 HTTP 缓存体系里要面对的是代理和源服务器，也必须区别对待，这里我就直接上图了，来个“看图说话”。





max-age、no-store、no-cache 这三个属性在 [第 20 讲](#) 已经介绍过了，它们也是同样作用于代理和源服务器。

关于缓存的生存时间，多了两个新属性“max-stale”和“min-fresh”。

“max-stale”的意思是如果代理上的缓存过期了也可以接受，但不能过期太多，超过 x 秒也会不要。“min-fresh”的意思是缓存必须有效，而且必须在 x 秒后依然有效。

比如，草莓上贴着标签“max-age=5”，现在已经在冰柜里存了 7 天。如果有请求“max-stale=2”，意思是过期两天也能接受，所以刚好能卖出去。

但要是“min-fresh=1”，这是绝对不允许过期的，就不会买走。这时如果有另外一个菠萝是“max-age=10”，那么“7+1<10”，在一天之后还是新鲜的，所以就能卖出去。

有的时候客户端还会发出一个特别的“only-if-cached”属性，表示只接受代理缓存的数据，不接受源服务器的响应。如果代理上没有缓存或者缓存过期，就应该给客户端返回一个 504 (Gateway Timeout)。

## 实验环境

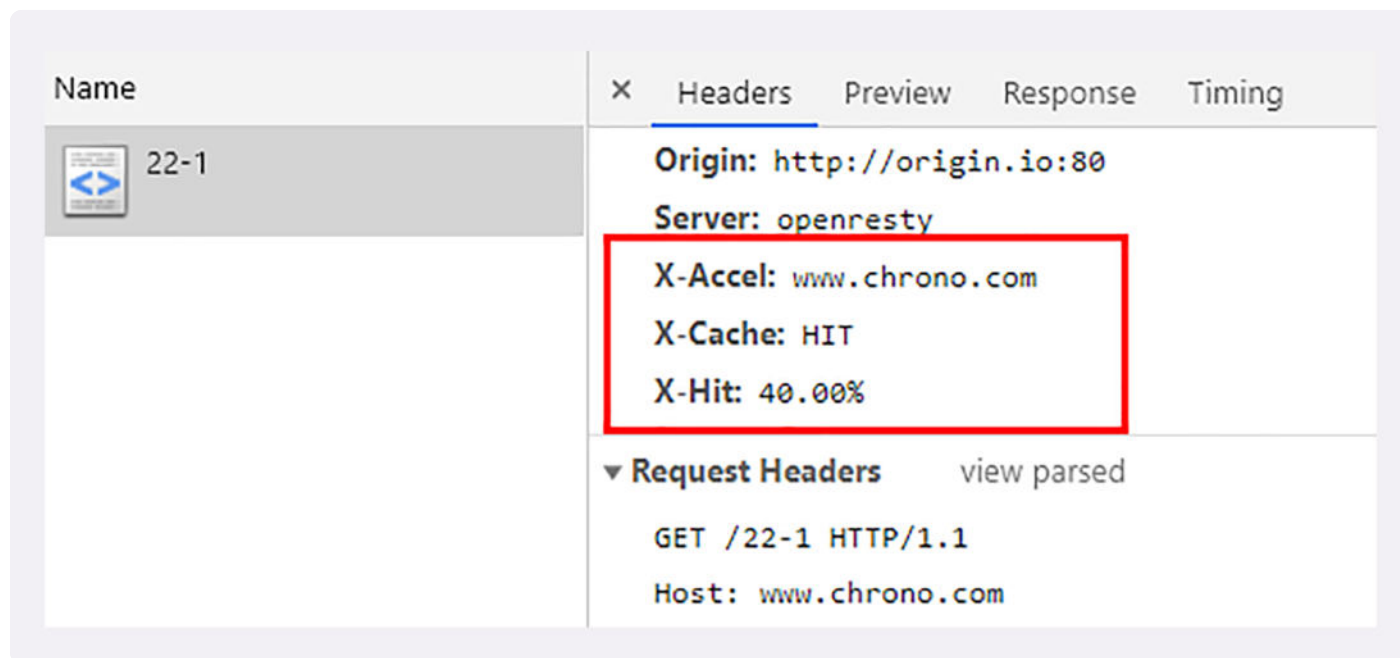
信息量有些大，到这里你是不是有点头疼了，好在我们还有实验环境，用 URI“/22-1”试一下吧。

它设置了“Cache-Control: public, max-age=10, s-maxage=30”，数据可以在浏览器里存 10 秒，在代理上存 30 秒，你可以反复刷新，看看代理和源服务器是怎么响应的，同样也可以配合 Wireshark 抓包。





代理在响应报文里还额外加了“X-Cache”“X-Hit”等自定义头字段，表示缓存是否命中和命中率，方便你观察缓存代理的工作情况。



## 其他问题

缓存代理的知识就快讲完了，下面再简单说两个相关的问题。

第一个是“**Vary**”字段，在 [第 15 讲](#) 曾经说过，它是内容协商的结果，相当于报文的一个版本标记。

同一个请求，经过内容协商后可能会有不同的字符集、编码、浏览器等版本。比如，“Vary: Accept-Encoding”“Vary: User-Agent”，缓存代理必须要存储这些不同的版本。

当再收到相同的请求时，代理就读取缓存里的“Vary”，对比请求头里相应的“Accept-Encoding”“User-Agent”等字段，如果和上一个请求的完全匹配，比如都是“gzip”“Chrome”，就表示版本一致，可以返回缓存的数据。

另一个问题是“**Purge**”，也就是“缓存清理”，它对于代理也是非常重要的功能，例如：

- 过期的数据应该及时淘汰，避免占用空间；
- 源站的资源有更新，需要删除旧版本，主动换成最新版（即刷新）；
- 有时候会缓存了一些本不该存储的信息，例如网络谣言或者危险链接，必须尽快把它们删除。





清理缓存的方法有很多，比较常用的一种做法是使用自定义请求方法“PURGE”，发给代理服务器，要求删除 URI 对应的缓存数据。

## 小结

1. 计算机领域里最常用的性能优化手段是“时空转换”，也就是“时间换空间”或者“空间换时间”，HTTP 缓存属于后者；
2. 缓存代理是增加了缓存功能的代理服务，缓存源服务器的数据，分发给下游的客户端；
3. “Cache-Control”字段也可以控制缓存代理，常用的有“private”“s-maxage”“no-transform”等，同样必须配合“Last-modified”“ETag”等字段才能使用；
4. 缓存代理有时候也会带来负面影响，缓存不良数据，需要及时刷新或删除。

## 课下作业

1. 加入了代理后 HTTP 的缓存复杂了很多，试着用自己的语言把这些知识再整理一下，画出有缓存代理时浏览器的工作流程图，加深理解。
2. 缓存的时间策略很重要，太大太小都不好，你觉得应该如何设置呢？

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 常用的缓存代理软件有 Squid、Varnish、ATS (Apache Traffic Server) 等，而 Nginx 不仅是 Web 服务器、代理服务器，也是一个出色的缓存代理服务器，堪称“全能”。



- 02 有的缓存代理在“Cache Hit”的时候会在响应报文里加一个 Age 头字段，表示报文的生存时间，即已经在缓存里存了多久，通常它会小于“Cache-Control”里的 max-age 值，如果大于就意味着数据是“陈旧的”（stale）。
- 03 判断缓存是否命中（Hit）类似于查询 hash 表，使用的 key 通常就是 URI，在 Nginx 里可以用指令“proxy\_cache\_key”自定义。
- 04 Nginx 对 Vary 的处理实际上是做了 MD5，把 Vary 头摘要后写入缓存，请求时不仅比较 URI，也比较摘要。

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

 赞 11  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。



## 学习推荐

# JVM + NIO + Spring

各大厂面试题及知识点详解

限时免费



## 精选留言 (33)

写留言



Teresa

2020-04-27

针对作业一的回答：

浏览器拿到一个网址的时候，先判断是否允许缓存，允许会先查看本地缓存:1.有缓存并在缓存可用期那直接拿来用。2.缓存不存在或者不可用 那需要请求。

浏览器拿到host，判断：1.ip+port 那直接请求对应的服务器 2.域名 那开展一系列的dns递归查询：先拿dns缓存，没有缓存->本地dns服务器->根dns服务器->顶级dns服务器->权威dns服务器->GSLB，查到ip返回最优ip组实现负载均衡，浏览器随机或者轮询取一个ip开始它的http请求之旅。

浏览器判断该网页是否允许缓存，然后添加Cache-Control的各种字段no-store是否允许缓存/no-cache缓存必须进行验证/no-cache只接受代理的缓存等,max-age最大生存时间 max-stale 短时间过期可用 min-fresh 最短有效时间等。If-Modified-Since/if-None-Match/Last-modified/ETag等字段用于判断服务端是否有更新。然后将请求发给代理服务器。请求代理服务器，如果是第一次，要经历浏览器和代理服务器的3次tcp握手进行连接，连接成功，发送http请求。

代理服务器拿到请求，首先查看是否允许缓存，允许那就查看自己本地缓存有没有，通过查看max-age/max-stale/min-fresh等信息判断是否过期，没有过期直接拿来用，将数据返回给客户端。如果过期了，代理服务器将用客户端的请求，再次像真实服务器进行请求。如果也是第一次连接，需要经历代理服务器和真实服务器的3次tcp握手，连接成功，发送请求。



真实服务器收到请求之后，通过if-Modified-Since/Last-Modified/if-None-Match/ETag等字段判断是否有更新，没有更新，直接返回304。如果有更新，则将数据打包http response 返回。返回头字段会添加Cache-Control字段，用来判断缓存的控制策略以及生存周期，no-store不允许缓存/no-cache使用缓存必须先验证/must-revalidate缓存不过期可用过期必须重新请求验证/proxy-revalidate缓存过期只要求代理进行请求验证 private不能在代理层保存只能在客户端保存/public缓存完全开放 s-maxage缓存在代理上可以缓存的时间 no-transform不允许代理对缓存做任何的改动。然后根据业务需求判断该地址是不是需要重定向，如果需要是短期的重定向还是永久的重定向，按需将状态码修改为301或者302。最后真实服务器将数据打包成http相应 回给代理服务器。

代理服务器收到真实服务器的回应数据，首先会查看Cache-Control里的字段，是否允许它进行缓存，如果是private，代理服务器不进行缓存，直接返回给客户端。public则根据s-maxage/no-transform进行缓存，如果可以优化并且代理服务器需要优化，那可能会先优化数据，否则同时将数据回发给客户端。

客户端收到数据，如果是304，则直接拿缓存数据进行渲染，并修改相关缓存变量，比如时间，以及缓存使用策略。如果收到了301或者302，那么客户端会再次发起新的url请求，进行跳转到最终的页面。

最后，底层tcp 经过4次挥手，完成关闭连接。

作者回复: 整理的非常详细完整，32个赞!

共 3 条评论 >

👍 102



龙宝宝

2019-07-29

max-stale相当于延长了过期时间，min-fresh相当于缩短了过期时间，可以这样理解吗

作者回复: 可以这么理解，也是一个很好的角度。

共 3 条评论 >

👍 22



钱

2020-03-30

首先，老师的敬业精神令人钦佩，几乎每问必答，再者，没有给人一种你怎么这么笨这么简单的问题你还问的感受，而且看到有些地方不严谨也会干净利索的说自己可能弄错了。是个经验丰富技术精干为人真诚的大哥形象，工作或生活中有这样的朋友或大哥是一件很幸运的事情。

缓存的时间策略很重要，太大太小都不好，你觉得应该如何设置呢？

我觉得需要根据具体情况来定：

如果缓存的内容不变，那可以把缓存时间设置为永久。

如果缓存的内容会变化，但周期较长，可以根据她的变化周期来设置，比如：一天或一周

如果缓存的内容变化频繁，那缓存的过期时间就需要更短了，比如：一分钟



如果缓存的内容随时变化，且没啥规律，那还是不用用了

总之是根据场景来的核心是在提速的愿望能实现的前提下，数据也是最新的，否则不如不用缓存，从另一个角度来讲不用缓存几乎是不可能的，缓存在处处使用着，因为计算机本身就在各种各样的使用着缓存。如果完全不用直接从磁盘获取数据，也可以认为是使用缓存的一种特殊情况，缓存的过期时间为零即使即过期。

作者回复:

1.我的缺点就是太实在，所以一直只能当小兵，当不了boss，笑。

2.从http的缓存里也可以学到很多通用的知识，用在系统的其他地方也是可以的，所以说学习协议对程序员真的是一个基本功。

共 3 条评论 >

👍 19



院长。

2019-07-17

老师您好，我想请问一下，为什么有的地方说cache-control默认是private（比如cache-control的百度百科），有的地方说默认是public（比如您这篇文章），是百度百科的是错误的吗？还是根据场景不同所以默认不同吗？

作者回复: 我看了一下rfc，对于private和public没有明确的默认值说法，可能是我弄错了，需要再测试看看。

共 2 条评论 >

👍 13



闫飞

2019-07-18

min-fresh的含义是距离过期时间必须不短于约定的时间，保证取到的是短期内不会过时的内容。

但是这里有个风险是，万一服务器端因为某些原因重新刷新了资源(服务迁移等)，那么怎么反向通知缓存服务器去清理资源呢？尤其是已经返回给客户端的之前标记为fresh的资源？

作者回复: http协议里没有对此做出规定，一般的做法是由源站向代理发送pull请求，要求代理主动更新缓存。

这个pull请求不属于http协议，具体实现就看两者之间的约定了。



👍 4



何用

2019-07-17

老师，CDN 服务是不是就是缓存代理的一种应用？还有文中图片的 X-Accel 是什么意思呢？

作者回复: 1, 是的。

2. X-Accel是自定义字段, 和x-powered-by差不多, 意思是被谁加速。



4



ttsunami

2020-06-10

代理服务器真的那么听话吗？源给个private, 结果自己却偷偷做一些操作也可以吧？如何验证代理服务器的处理是否夹带私心呢？纯靠自觉吗？哈哈 颇有 将在外军令有所不受的味道

作者回复: 说的很对, http不是强制性的规范, 代理不遵守也是完全可以的, 如果有私下操作我们也很难判断出来, http本身没有这方面的要求。



4



Geek\_steven\_wang

2019-08-23

如果cache-control中没有 no-store no-cache must-revalidate这时浏览器会怎么处理？

Max-stale min-fresh 那个优先级高, 如果两个都有, 那个生效？

作者回复:

1.没有这些就按普通的缓存策略来处理, 在有效期内直接用, 过期发条件请求。

2.这两个属性本身就是冲突的, 如果同时给出, 那就由服务器自己决定策略了, rfc本身没有对此做出规定。



3



magict4

2021-06-06

老师您好,

我想问一下, 现在大多数网站都开启了 HTTPS, 缓存代理还有用武之地吗? 我对 HTTPS 的理解是它是端到端加密。介于客户端与服务器中间的缓存服务器是没法解密, 缓存数据的。





作者回复: 这个需要看缓存代理怎么配置了。

如果它在四层, 只是转发, 看不到https的内容, 就无法缓存。

如果它在七层, 本身实际上也是一个客户端, 能够与后端通信看到https的内容, 就可以缓存。



👍 2



鸟人

2019-07-29

请问时间换空间是什么呢?

作者回复: 比如数据压缩, 就是时间换空间, 增加了计算时间, 减少了数据量。



👍 3



徐海浪

2019-07-17

我们的做法是源服务器跟代理服务器联动, 源服务器有文件变化(版本发布), 触发更新代理服务器缓存。如果没有联动的机制, 简单粗暴根据应用升级周期设置过期时间也可以, 但这样会多做无用功。

作者回复: 欢迎经验分享!

共 2 条评论 >

👍 2



居培波

2019-07-17

老师能结合nginx讲下缓存及代理吗? 还是后面探索篇有讲。

作者回复: 只要理解的http的缓存代理, nginx的是比较容易掌握的, 可以结合nginx的文档, 看proxy相关的指令, 逐条对照http的功能。

单纯讲nginx就有点太大了, 讲不过来。



👍 2



乘风破浪

2021-04-03

“must-revalidate”是只要过期就必须回源服务器验证, 而新的“proxy-revalidate”只要求代



理的缓存过期后必须验证，客户端不必回源，只验证到代理这个环节就行了。

看了一下MDN关于这两个头字段的解释

must-revalidate

Indicates that once a resource becomes stale, caches must not use their stale copy without successful validation on the origin server.

proxy-revalidate

Like must-revalidate, but only for shared caches (e.g., proxies). Ignored by private caches.

感觉MDN说的proxy-revalidate的意思似乎是对于存储在代理服务器上的共享缓存的验证策略，如果过期，必须回源验证，而不是说客户端的缓存【私有缓存】失效后回源到代理服务器验证。

而这也反过来推论，must-revalidate是向上游服务器验证cache，验证不一定回源到源服务器。

---请大师指正。。

作者回复: 我又仔细看了一下rfc7234，感觉是我没有太表述清楚。

proxy-revalidate意思是客户端可以到存储在代理服务器上的缓存做验证，私有缓存必须回源，其他不必回源。

同时代理服务器上的缓存失效了当然也必须回源验证。

表述上稍微有点绕，抱歉。



1



旅途

2020-08-12

我还要提醒你一点，源服务器在设置完“Cache-Control”后必须要为报文加上“Last-modified”或“ETag”字段。否则，客户端和代理后面就无法使用条件请求来验证缓存是否有效，也就不会有 304 缓存重定向

老师 这句话 没理解 能再说下吗



作者回复: “Cache-Control”字段最好和“Last-modified”或“ETag”字段搭配使用，这样客户端就可以更好地利用缓存，用条件请求来验证缓存是否有效。

如果只有“Cache-Control”，没有“Last-modified”或“ETag”，那么客户端在缓存失效后就无法发出条件请求，就得重新传一遍内容，浪费了带宽。



👍 2



**lesserror**

2019-12-10

1. 老师，我看那个完整的服务器端缓存控制策略图，在图上no-cache和must-revalidate两个一般只能存在一个吧？不能同时传递。
2. 还有：“比较常用的一种做法是使用自定义请求方法“PURGE””，意思是自己在代码里面写个方法处理缓存的数据，对吧？

作者回复：

1.是的。

2.没错，http允许自定义请求方法和处理逻辑。



👍 1



**Geek\_steven\_wang**

2019-08-22

1. 文中第一张图服务器缓存控制，其实是服务器根据缓存策略向response中插入各header的过程，缓存服务器，浏览器根据header处理缓存。
2. 文中客户端缓存过程中，是浏览器根据自己的缓存策略，向服务器发请求时，设置各header。但这些策略浏览器怎么知道呢？是用户通过浏览器设置界面设置吗？还是ajax请求时设置？如果不是前后端分类的应用，怎么设置这些header？
3. 既然服务器端有自己的缓存策略，那客户端请求上来时，服务器会根据客户端请求header调整策略吗？如果会，是应用服务器自己处理，还是程序员代码预先写好处理逻辑？
4. 什么情况下客户端会有only-if-cached

作者回复：

1.正确。

2.http客户端有很多种，浏览器只是其中之一，如果用Python、php自己写客户端，那就可以使用这些缓存策略。

浏览器通常有一些最基本的策略，而ajax等就可以自己灵活设置。

3.服务器控制资源和缓存，它会检查请求资源的有效期，与客户端的请求比对，返回304或者是新的



内容，应用服务器和应用服务器都可以设置。

4.only-if-cached这个我也没有见过具体的应用场景，但既然有这个属性，就应该是有用的。

共 2 条评论 >

👍 1



一步

2019-07-17

老师请教一个有关302的问题，就是再前端代码中有个post请求，去请求后端服务器，当后端服务器处理完业务逻辑后，需要重定向到另一个网站，返回的是一个302的状态码和响应头Location是另一个网站地址。这时候问题出现了，当返回到前端的时候，浏览器没有自动跳转重定向后的地址，而是当作接口去请求了需要跳转后的地址，然后就出现了跨域的问题这个是什么原因呢？

作者回复: 可以参考一下第18讲，改用303 see other。



👍 1



djfhchdh

2019-07-17

max-stale和min-fresh还是不太明白~~

作者回复: max-stale是可以接受的过期时间，min-fresh是可以接受的新鲜时间。

不好理解也没事，这两个属性用的不多，可以以后实际遇到了再体会。



👍 1



silence

2021-08-31

老师请问图中的小黑圆点是指的数据包吗？

作者回复: 不是。

黑圆点是uml语言里的要素，我在这里借用了一下，表示逻辑流程的开始和结束。



👍



Unknown element

2021-06-12

当再收到相同的请求时，代理就读取缓存里的“Vary”，对比请求头里相应的“Accept-Encodi

ng”“User-Agent”等字段，如果和上一个请求的完全匹配，比如都是“gzip”“Chrome”，就表示版本一致，可以返回缓存的数据。

老师问下在第15讲里说vary是响应头的字段，表示服务器根据哪些字段生成的响应报文，这里为什么成了请求头的字段？

作者回复: vary字段会存储请求头里相应的“Accept-Encoding”“User-Agent”等字段，形成一个对应请求的“vary”版本，并不是在客户端请求时发出的。

