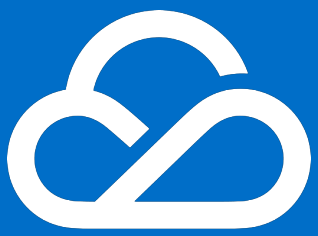
腾讯云内容分发网络

运维指南

产品文档



腾讯云

【版权声明】

【商标声明】

【服务声明】

本文档意在向客户介绍腾讯云全部或部分产品、服务的当时的整体概况，部分产品、服务的内容可能有所调整。您所购买的腾讯云产品、服务的种类、服务标准等应由您与腾讯云之间的商业合同约定，除非双方另有约定，否则，腾讯云对本文档内容不做任何明示或模式的承诺或保证。

文档目录

CNAME配置

接入cdn审核通过之后，系统会为您自动分配一下cname域名（*.fast.cdntip.com)，请至域名服务提供商处完成CNAME配置。(CNAME域名是不能直接访问的）

cdn1.png

域名CNAME解析设置方法如下（严格按照接入cdn的域名做cname解析）

1、进入您的域名管理中心，选中您要设置解析的域名并点击”解析 “：

解析.png

2、点击“添加记录” ：

添加记录.png

3、主机记录请填写www或cdn等主机名，记录类型请选择CNAME，记录值请填写CNAME域名，点击“保存”：

注：

DNSPod设置方法

CNAME记录的添加方式

A.主机记录处填子域名（比如需要添加www.123.com的解析，只需要在主机记录处填写www即可；如果只是想添加123.com的解析，主机记录直接留空，系统会自动填一个“@”到输入框内，@的CNAME会影响到MX记录的正常解析，添加时慎重考虑）。

B.记录类型为CNAME。

C.线路类型（默认为必填项，否则会导致部分用户无法解析；在上图中，默认的作用为：除了联通用户之外的所有用户，都会指向1.com）。

D.记录值为CNAME指向的域名，只可以填写域名，记录生成后会自动在域名后面补一个“.”，这是正常现象。

E.MX优先级不需要填写。

F.TTL不需要填写，添加时系统会自动生成，默认为600秒（TTL为缓存时间，数值越小，修改记录各地生效时间越快）。

万网设置方法

域名CNAME解析设置方法

别名（CNAME）解析允许您将多个域名映射到同一台计算机。例如，有一台计算机名为“host.mydomain.com”（A记录），它同时提供WWW和MAIL服务；为了便于用户访问服务，可以为该计算机设置两个别名（CNAME）：WWW和MAIL。这两个别名的全称就是“www.mydomain.com”和“mail.mydomain.com”。实际上他们都指向“host.mydomain.com”。

注意：CNAME解析时记录值只能填写另一个域名，即domain，而非网址；且CNAME记录与其它各类型的同名记录会存在冲突，不能有重复前缀的解析记录存在，例如您的域名为abc.com您设置了一条www.abc.com就不能再设置www.abc.com的其它类型的解析记录，否则会导致所有解析记录冲突不生效。

域名CNAME解析设置方法

1、登录万网会员中心；

2、点击会员中心左侧导航栏中的【产品管理】--“我的云解析”进入万网云解析列表页。

3、点击要解析的域名，进入解析记录页。

4、进入解析记录页后，点击新增解析按钮，开始设置解析记录。

5、若要设置CNAME解析记录，将记录类型选择为CNAME；主机记录即域名前缀，可任意填写（如：www）；记录值填写为当前域名指向的另一个域名；解析线路，TTL默认即可。

6、填写完成后，点击保存按钮，完成解析设置。

友情提示：

1）CNAME解析记录RR中不能为空，且CNAME记录与其它各类型的同名记录都存在冲突，不能有重复前缀的解析记录存在，例如您的域名为abc.com您设置了一条www.abc.com就不能再设置www.abc.com的其它类型的解析记录，否则会导致所有解析记录冲突不生效。

2）新增解析实时生效，而修改解析需要72小时的全球生效时间。

新网设置方法

设置别名（CNAME记录）

即：别名记录。这种记录允许您将多个名字映射到同一台计算机。 通常用于同时提供WWW和MAIL服务的计算机。例如，有一台计算机名为“host.mydomain.com”（A记录）。 它同时提供WWW和MAIL服务，为了便于用户访问服务。可以为该计算机设置两个别名（CNAME）：WWW和MAIL。（如：图六）

（图六：CNAME记录设置）

验证CNAME是否生效

不同的DNS服务商，CNAME生效的时间略有不同，一般会在半个小时之内生效。您也可以通过PING的方式来查询是CNAME是否生效，如果PING到tcdn.qq.com表示域名CNAME生效。

CDN缓存那些事

1 CDN是什么？

谈到CDN的作用，可以用8年买火车票的经历来形象比喻：

8年前，还没有火车票代售点一说，12306.cn更是无从说起。那时候火车票还只能在火车站的售票大厅购买，而我所住的小县城并不通火车，火车票都要去市里的火车站购买，而从县城到市里，来回就是4个小时车程，简直就是浪费生命。后来就好了，小县城里出现了火车票代售点，可以直接在代售点购买火车，方便了不少，全市人民再也不用在一个点苦逼的排队买票了。

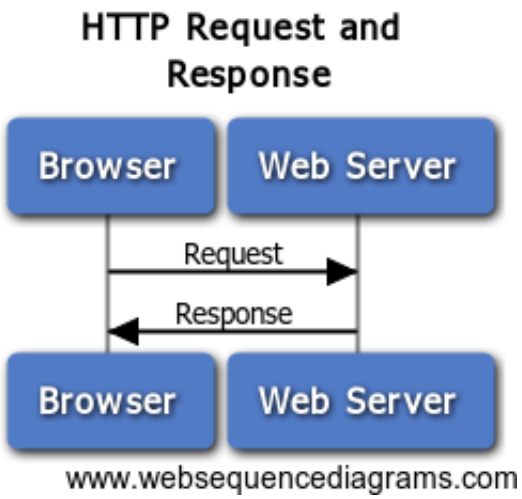
CDN就可以理解为分布在每个县城的火车票代售点，用户在浏览网站的时候，CDN会选择一个离用户最近的CDN边缘节点来响应用户的请求，这样海南移动用户的请求就不会千里迢迢跑到北京电信机房的服务器（假设源站部署在北京电信机房）上了。

CDN的优势很明显：（1）CDN节点解决了跨运营商和跨地域访问的问题，访问延时大大降低；（2）大部分请求在CDN边缘节点完成，CDN起到了分流作用，减轻了源站的负载。

2 缓存是什么？

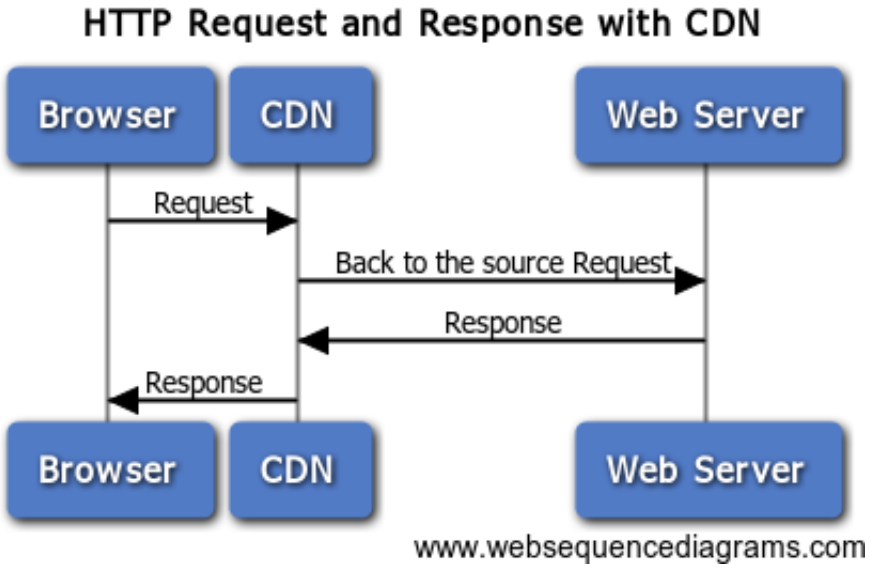
这里不深究CDN背后高大上的架构，也不讨论CDN如何做到全局流量调度策略，本文着重讨论在有了CDN后，数据是如何被缓存的。缓存是一个到处都存在的用空间换时间的例子。通过使用多余的空间，我们能够获取更快的速度。

首先，看看网站没有接入CDN时，用户浏览器与服务器是如何交互的：

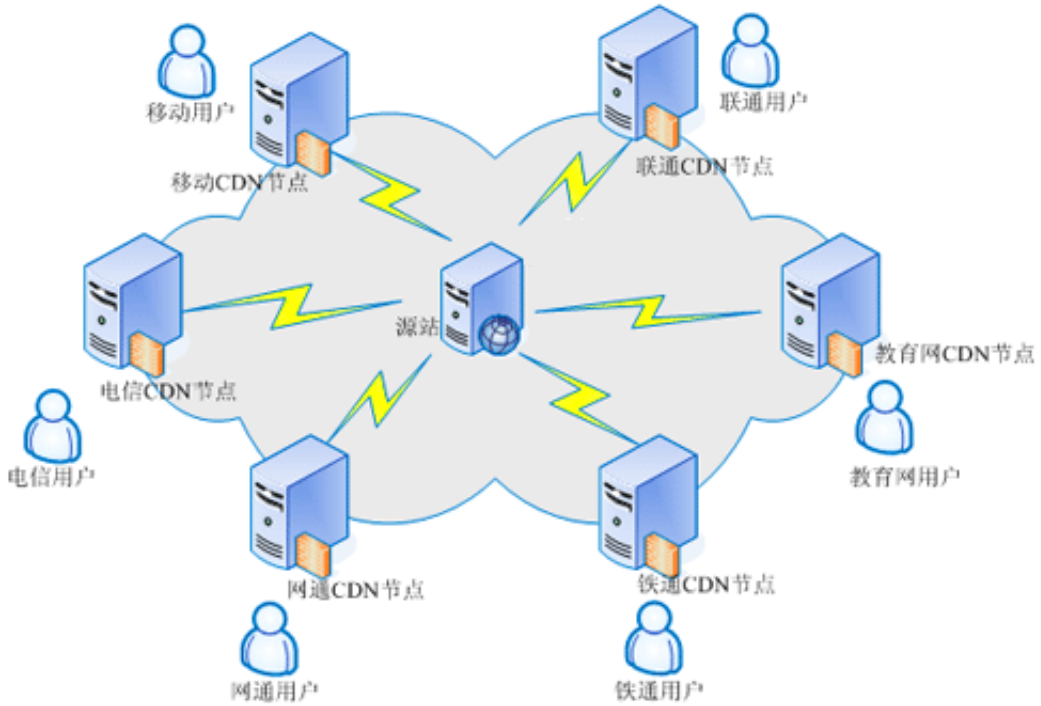


用户在浏览网站的时候，浏览器能够在本地保存网站中的图片或者其他文件的副本，这样用户再次访问该网站的时候，浏览器就不用再下载全部的文件，减少了下载量意味着提高了页面加载的速度。

如果中间加上一层CDN，那么用户浏览器与服务器的交互如下：



客户端浏览器先检查本地缓存是否过期，如果过期，则向CDN边缘节点发起请求，CDN边缘节点会检测用户请求数据的缓存是否过期，如果没有过期，则直接响应用户请求，此时一个完成http请求结束；如果数据已经过期，那么CDN还需要向源站发出回源请求（back to the source request），来拉取最新的数据。CDN的典型拓扑图如下：



图片来源：http://grefr.iteye.com/blog/2004248

可以看到，在存在CDN的场景下，数据经历了客户端（浏览器）缓存和CDN边缘节点缓存两个阶段，下面分别对这两个阶段的缓存进行详细的剖析

3 客户端（浏览器）缓存

3.1 客户端缓存的缺点

客户端缓存减少了的服务器请求，避免了文件重复加载，显著地提升了用户地方。但是当网站发生了更新的时候（如替换了css、js以及图片文件），浏览器本地仍保存着旧版本的文件，从而导致无法预料后果。

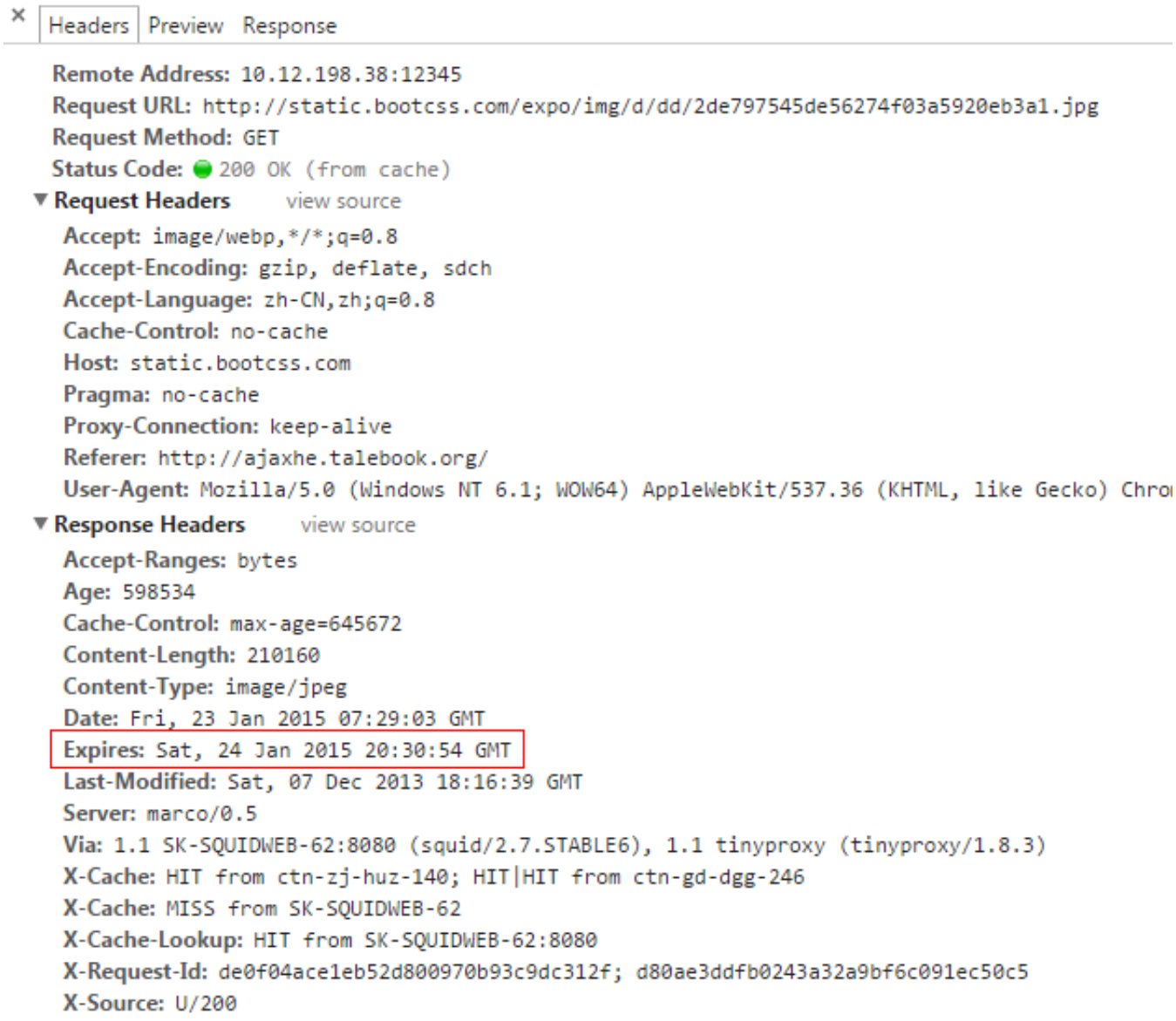
曾几何时，一个页面加载出来，页面各元素位置乱飘，按钮点击失效，前端GG都会习惯性地问一句：“缓存清了没？”，然后Ctrl+F5，Everything is OK。但有些时候，如果我们是简单地在浏览器地址栏中敲一个回车，或者是仅仅按F5刷新，问题依然没有解决，你可知道这三种不同的操作方式，决定浏览器不同的刷新缓存策略？

浏览器如何来确定使用本地文件还是使用服务器上的新文件？下面来介绍几种判断的方法。

3.2 浏览器缓存策略

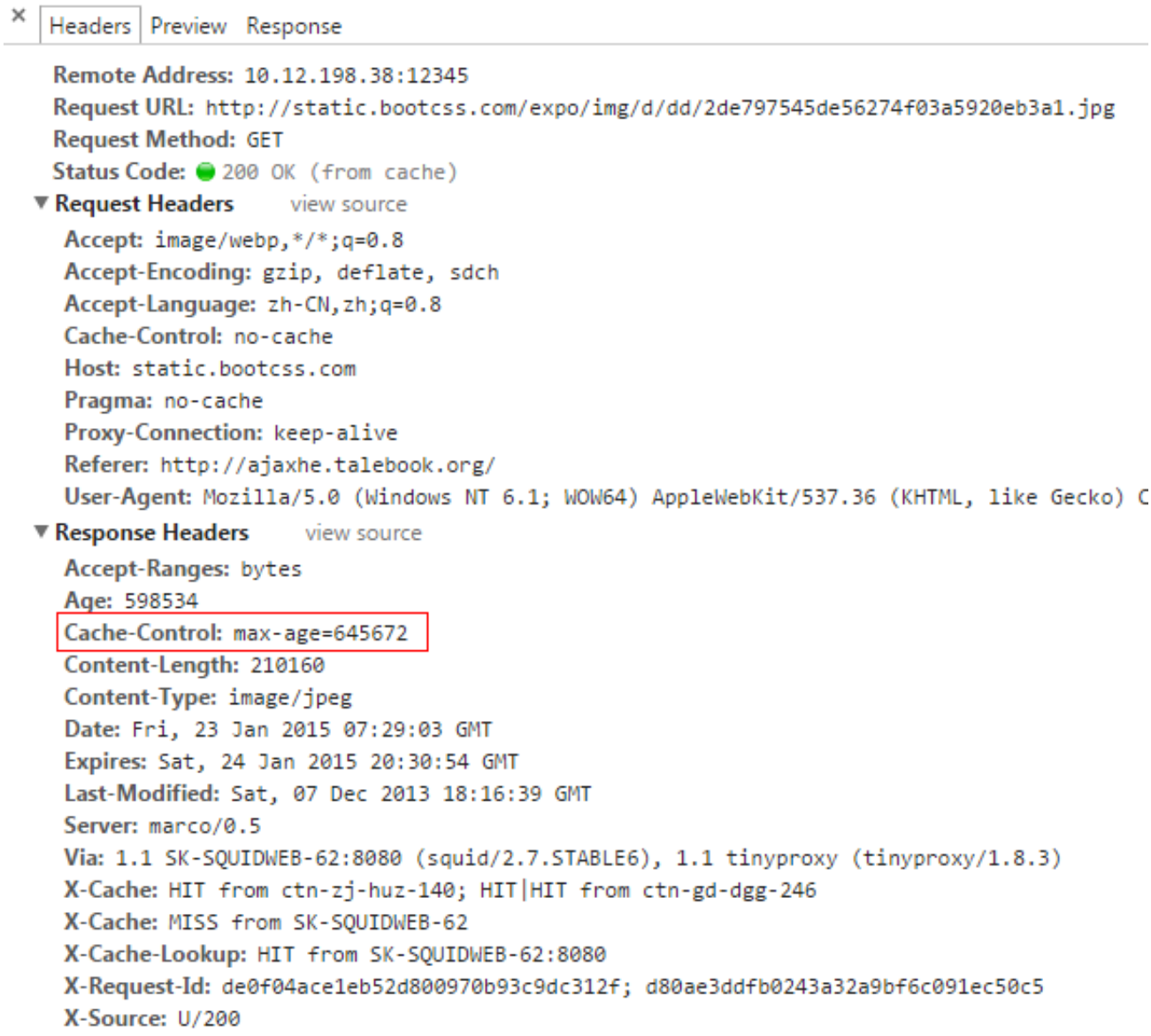
Expires

Expires:Sat, 24 Jan 2015 20:30:54 GMT



如果http响应报文中设置了Expires，在Expires过期之前，我们就避免了和服务器之间的连接。此时，浏览器无需想浏览器发出请求，只需要自己判断手中的材料是否过期就可以了，完全不需要增加服务器的负担。

Cache-control: max-age



Expires的方法很好，但是我们每次都得算一个精确的时间。max-age
标签可以让我们更加容易的处理过期时间。我们只需要说，这份资料你只能用一个星期就可以了。

Max-age 使用秒来计量，如：
Cache-Control:max-age=645672
指定页面645672秒（7.47天）后过期。

Last-Modified

服务器为了通知浏览器当前文件的版本，会发送一个上次修改时间的标签，例如：

```
Request URL: http://ajaxhe.talebook.org/index_files/jquery.min.js
Request Method: GET
Status Code: 200 OK (from cache)
▼ Request Headers      view source
  Accept: */*
  Accept-Encoding: gzip, deflate, sdch
  Accept-Language: zh-CN,zh;q=0.8
  Cache-Control: no-cache
  Cookie: Hm_lvt_15c141dd41d16b47f32d4c5476e1f6f4=1421996641; Hm_lpvt_15c141dd41d16b47f3
  Host: ajaxhe.talebook.org
  Pragma: no-cache
  Proxy-Connection: keep-alive
  Referer: http://ajaxhe.talebook.org/
  User-Agent: Mozilla/5.0 (Windows NT 6.1; WOW64) AppleWebKit/537.36 (KHTML, like Gecko)
▼ Response Headers      view source
  Accept-Ranges: bytes
  Content-Encoding: gzip
  Content-Length: 33225
  Content-Type: application/javascript
  Date: Fri, 23 Jan 2015 07:04:41 GMT
  ETag: "39001d-1762a-50bf790757e00"
  Last-Modified: Tue, 06 Jan 2015 08:26:32 GMT
  Server: Apache/2.2.22 (Ubuntu)
  Vary: Accept-Encoding
  Via: 1.1 SK-SQUIDWEB-62:8080 (squid/2.7.STABLE6), 1.1 tinyproxy (tinyproxy/1.8.3)
  X-Cache: MISS from SK-SQUIDWEB-62
  X-Cache-Lookup: HIT from SK-SQUIDWEB-62:8080
```

这样浏览器就知道他收到的这个文件创建时间，在后续的请求中，浏览器会按照下面的规则进行验证：

1. 浏览器：Hey，我需要jquery.min.js这个文件，如果是在 Tue, 06 Jan 2015 08:26:32 GMT 之后修改过的，请发给我。

2. 服务器：（检查文件的修改时间）

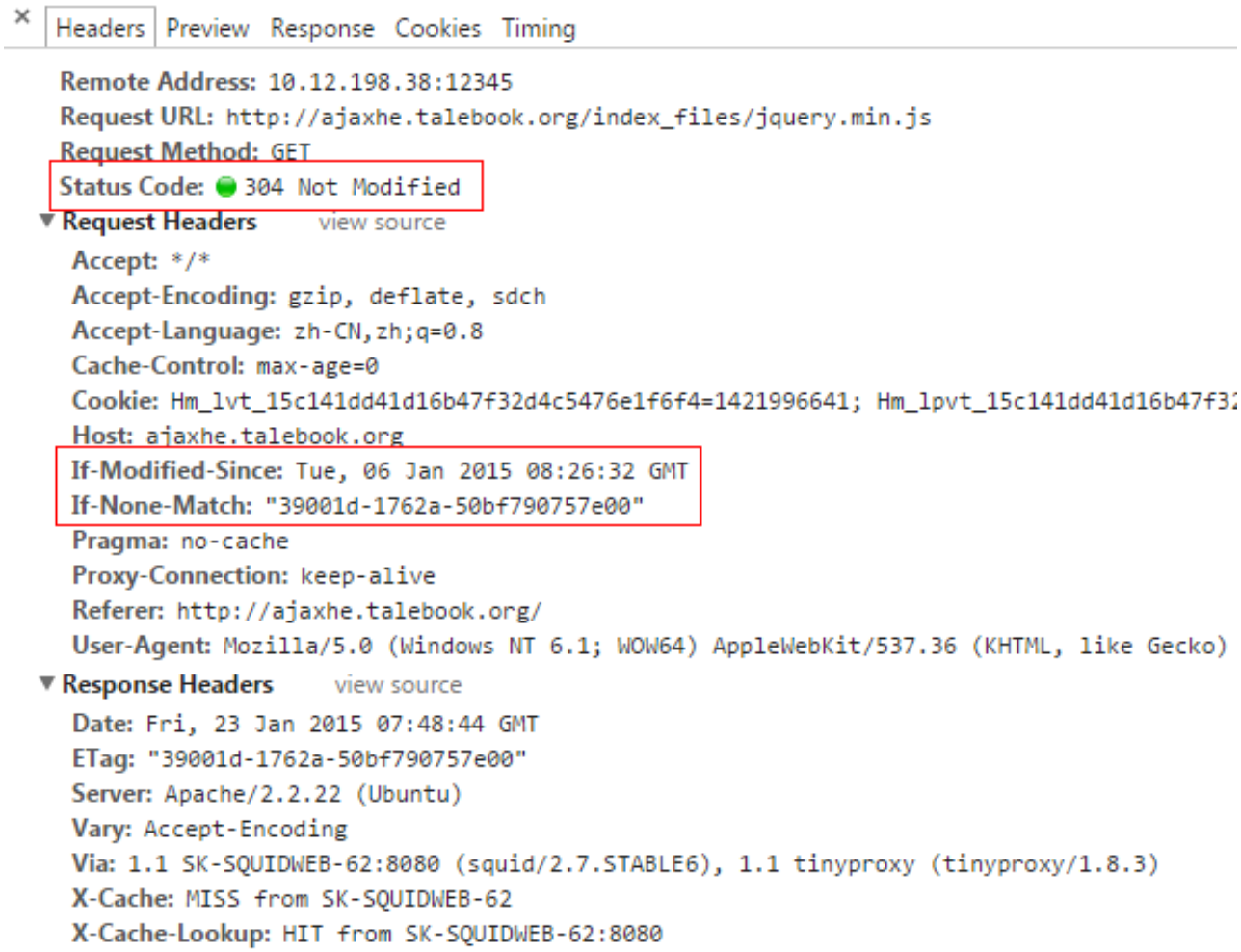
3. 服务器：Hey，这个文件在那个时间之后没有被修改过，你已经有最新的版本了。

4. 浏览器：太好了，那我就显示给用户了。

在这种情况下，服务器仅仅返回了一个304的响应头，减少了响应的数据量，提高了响应的速度。关于304响应，请参考：

http://www.cnblogs.com/ziyunfei/archive/2012/11/17/2772729.html

下图是按F5刷新页面后，页面返回304响应头。



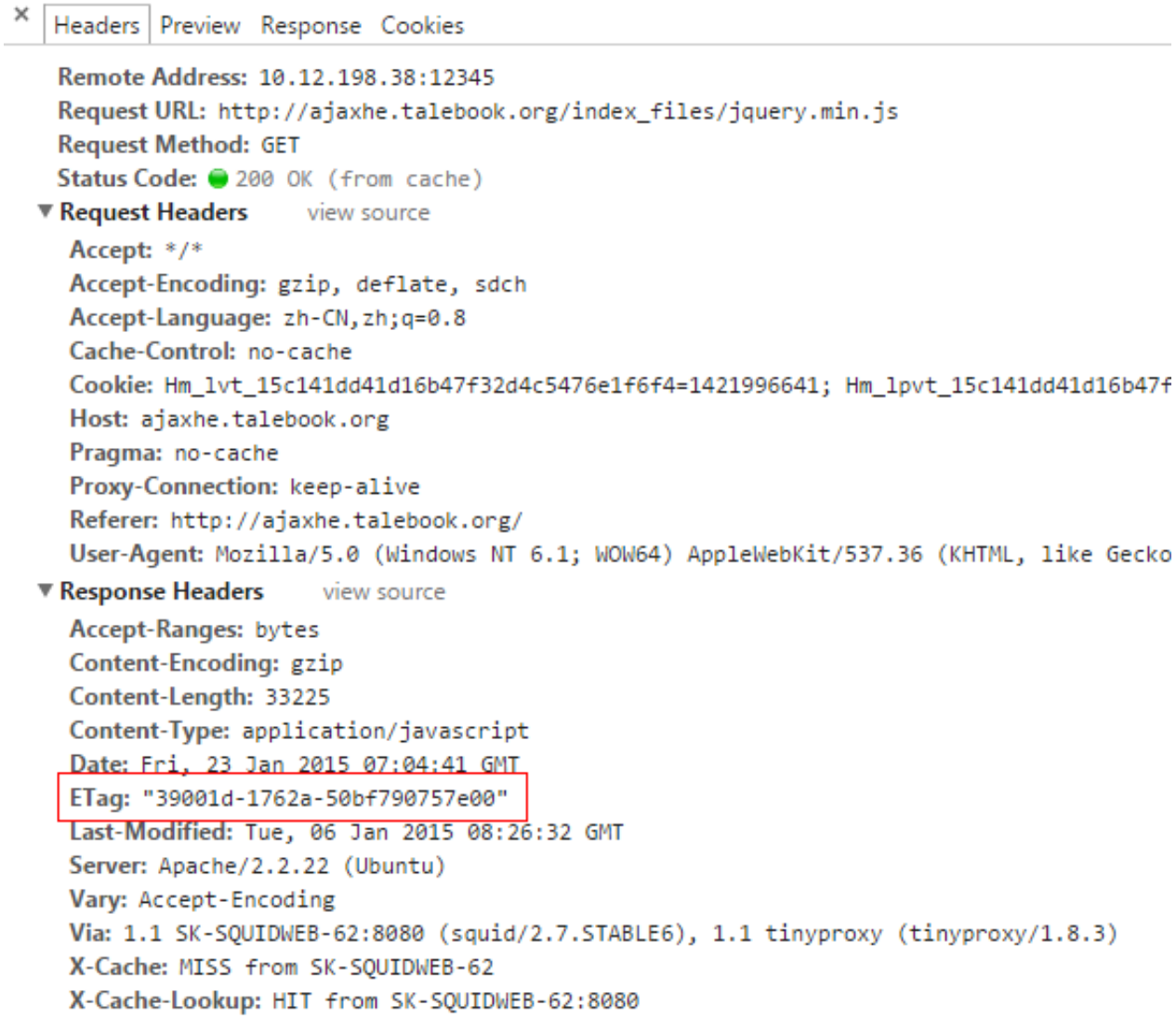
ETag

通常情况下，通过修改时间来比较文件是可行的。但是在一些特殊情况，例如服务器的时钟发生了错误，服务器时钟进行修改，夏时制DST到来后服务器时间没有及时更新，这些都会引起通过修改时间比较文件版本的问题。

ETag可以用来解决这种问题。ETag是一个文件的唯一标志符。就像一个哈希或者指纹，每个文件都有一个单独的标志，只要这个文件发生了改变，这个标志就会发生变化。

服务器返回ETag标签：

ETag:"39001d-1762a-50bf790757e00"



接下来的访问顺序如下所示：

1. 浏览器：Hey，我需要jquery.min.js这个文件，有没有不匹配"39001d-1762a-50bf790757e00"这个串的

2. 服务器：（检查ETag...）

3. 服务器：Hey，我这里的版本也是"39001d-1762a-50bf790757e00"，你已经是最新的版本了

4. 浏览器：好，那就可以使用本地缓存了

如同 Last-modified 一样，ETag 解决了文件版本比较的问题。只不过 ETag 的级别比 Last-Modified

高一些。

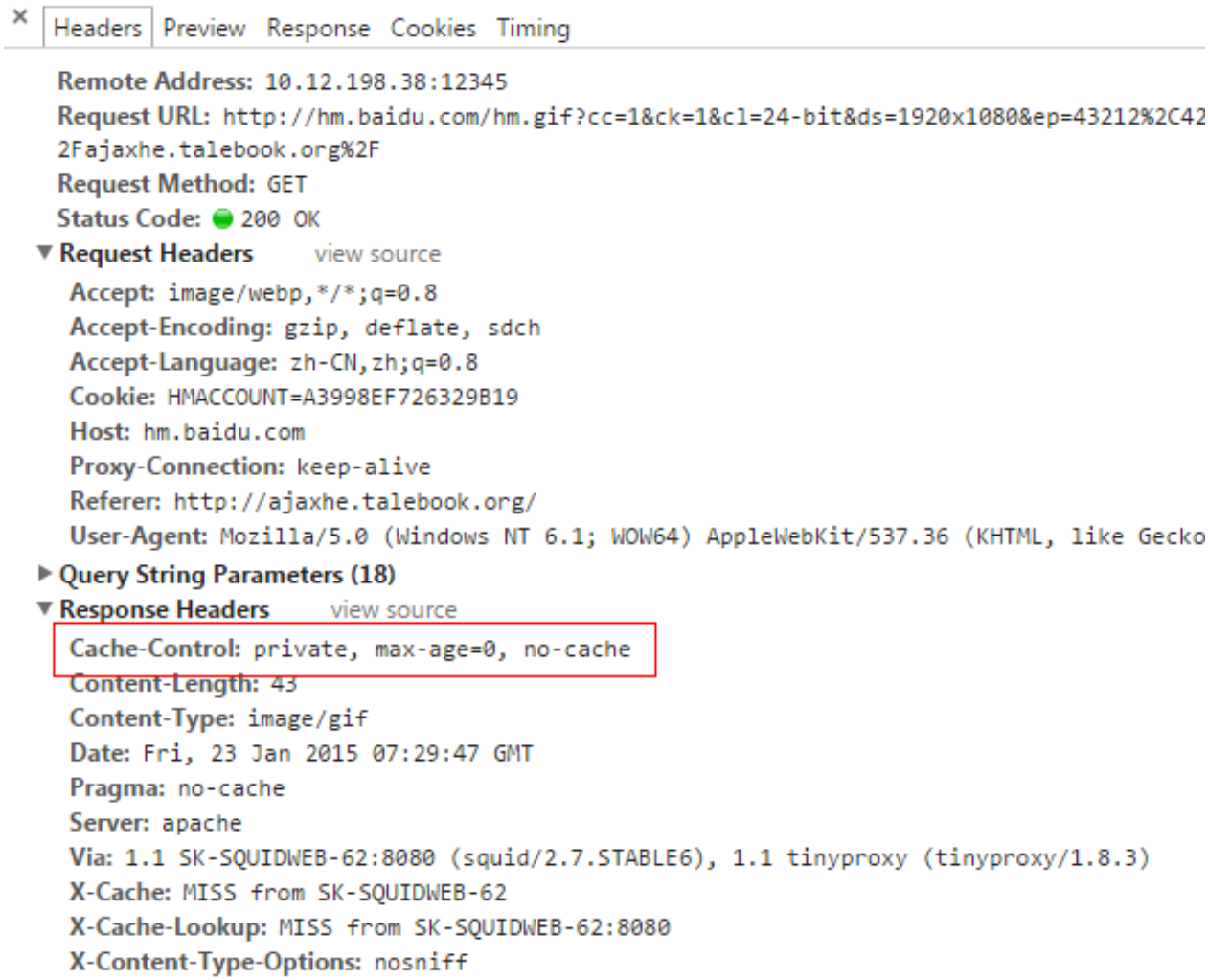
额外的标签

缓存标签永远不会停止工作，但是有时候我们需要对已经缓存的内容进行一些控制。

l Cache-control: public 表示缓存的版本可以被代理服务器或者其他中间服务器识别。

l Cache-control: private 意味着这个文件对不同的用户是不同的。只有用户自己的浏览器能够进行缓存，公共的代理服务器不允许缓存。

l Cache-control: no-cache 意味着文件的内容不应当被缓存。这在搜索或者翻页结果中非常有用，因为同样的URL，对应的内容会发生变化。



浏览器缓存刷新

1. 在地址栏中输入网址后按回车或点击转到按钮

浏览器以最少的请求来获取网页的数据，浏览器会对所有没有过期的内容直接使用本地缓存，从而减少了对浏览器的请求。所以，Expires，max-age标记只对这种方式有效。

2. 按F5或浏览器刷新按钮

浏览器会在请求中附加必要的缓存协商，但不允许浏览器直接使用本地缓存，它能够让 Last-Modified、ETag发挥效果，但是对Expires无效。

3. 按Ctrl+F5或按Ctrl并点击刷新按钮

这种方式就是强制刷新，总会发起一个全新的请求，不使用任何缓存。

4 CDN缓存

浏览器本地缓存失效后，浏览器会向CDN边缘节点发起请求。类似浏览器缓存，CDN边缘节点也存在着一套缓存机制。

4.1 CDN缓存的缺点

CDN的分流作用不仅减少了用户的访问延时，也减少的源站的负载。但其缺点也很明显：当网站更新时，如果CDN节点上数据没有及时更新，即便用户再浏览器使用Ctrl
+F5的方式使浏览器端的缓存失效，也会因为CDN边缘节点没有同步最新数据而导致用户访问异常。

4.2 CDN缓存策略

CDN边缘节点缓存策略因服务商不同而不同，但一般都会遵循http标准协议，通过http响应头中的Cache-control: max-age的字段来设置CDN边缘节点数据缓存时间。

当客户端向CDN节点请求数据时，CDN节点会判断缓存数据是否过期，若缓存数据并没有过期，则直接将缓存数据返回给客户端；否则，CDN节点就会向源站发出回源请求，从源站拉取最新数据，更新本地缓存，并将最新数据返回给客户端。

CDN服务商一般会提供基于文件后缀、目录多个维度来指定CDN缓存时间，为用户提供更精细化的缓存管理。

CDN缓存时间会对“回源率”产生直接的影响。若CDN缓存时间较短，CDN边缘节点上的数据会经常失效，导致频繁回源，增加了源站的负载，同时也增大的访问延时；若CDN缓存时间太长，会带来数据更新时间慢的问题。开发者需要增对特定的业务，来做特定的数据缓存时间管理。

4.3 CDN缓存刷新

CDN边缘节点对开发者是透明的，相比于浏览器Ctrl+F5的强制刷新来使浏览器本地缓存失效，开发者可以通过CDN服务商提供的“刷新缓存”接口来达到清理CDN边缘节点缓存的目的。这样开发者在更新数据后，可以使用“刷新缓存”功能来强制CDN节点上的数据缓存过期，保证客户端在访问时，拉取到最新的数据。