



链滴

# Http 缓存 4：论某一资源被缓存和使用缓存的条件

作者：[Lord-X](#)

原文链接：<https://ld246.com/article/1587990330589>

来源网站：[链滴](#)

许可协议：[署名-相同方式共享 4.0 国际 \(CC BY-SA 4.0\)](#)

<p> </p>

<h2 id="Http缓存-论某一资源被缓存和使用缓存的条件">Http 缓存：论某一资源被缓存和使用缓存的条件</h2>

<p>某一请求使用缓存的基本步骤有两个，一个是先将响应的资源缓存下来，另一个是在发起请求时足使用缓存的条件。下面先来看看什么样的响应会被缓存下来。</p>

<h3 id="什么样的响应会被缓存">什么样的响应会被缓存</h3>

<p>这里引用 RFC7234 原文做简要说明（原文：page6，第三章节）</p>

<p>A cache MUST NOT store a response to any request, unless:</p>

<ul>

<li>The request method is understood by the cache and defined as being<br>cacheable, and</li>

<li>the response status code is understood by the cache, and</li>

<li>the "no-store" cache directive (see Section 5.2) does not appear<br>in request or response header fields, and</li>

<li>the "private" response directive (see Section 5.2.2.6) does not<br>appear in the response, if the cache is shared, and</li>

<li>the Authorization header field (see Section 4.2 of [RFC7235]) does<br>not appear in the request, if the cache is shared, unless the<br>response explicitly allows it (see Section 3.2), and</li>

<li>the response either:

<ul>

<li>contains an Expires header field (see Section 5.3), or</li>

<li>contains a max-age response directive (see Section 5.2.2.8), or</li>

<li>contains a s-maxage response directive (see Section 5.2.2.9)<br>and the cache is shared, or</li>

<li>contains a Cache Control Extension (see Section 5.2.3) that<br>allows it to be cached, or</li>

<li>has a status code that is defined as cacheable by default (see<br>Section 4.2.2), or</li>

<li>contains a public response directive (see Section 5.2.2.5).</li>

</ul>

</li>

</ul>

<p><strong>大概解释一下：</strong></p>

<p>一个响应满足以下条件，即可被缓存：</p>

<ul>

<li>请求的方法必须能被缓存理解，而且必须是能被缓存的方法。例如 GET、HEAD 方法可以被缓存，POST 和 PATCH 如果设置了合适的 Header(Content-Location)也可以被缓存，但 PUT 和 DELETE 方法不可被缓存。</li>

<li>响应码可以被缓存理解，以下这些响应码是可以被缓存的：200、203、204、206、300、301、04、405、410、414、501。</li>

<li>请求头和响应头中都没有指定 "no-store" 头部</li>

<li>对于共享缓存来说（代理服务器），在响应头中没有指定为 "private"</li>

<li>对于共享缓存来说（代理服务器），请求中没有 "Authorization" 头部</li>

<li>响应头中含有 "Expires"、"max-age"、"s-maxage"、"public"，或通过 "Cache Control Extension" 明确指明要缓存，或返回的响应码指明要缓存时</li>

</ul>

<h3 id="如何命中缓存">如何命中缓存</h3>

<p>同样引用 RFC7234 原文做简要说明（原文：page8，第四章节）</p>

<p>When presented with a request, a cache MUST NOT reuse a stored<br>response, unless:</p>

<ul>

<li>The presented effective request URI (Section 5.5 of [RFC7230]) and<br>

- that of the stored response match, and
- the request method associated with the stored response allows it to be used for the presented request, and
- selecting header fields nominated by the stored response (if any) match those presented (see Section 4.1), and
- the presented request does not contain the no-cache pragma (Section 5.4), nor the no-cache cache directive (Section 5.2.1), unless the stored response is successfully validated (Section 4.3), and
- the stored response does not contain the no-cache cache directive (Section 5.2.2), unless it is successfully validated (Section 4.3), and
- the stored response is either:
  - fresh (see Section 4.2), or
  - allowed to be served stale (see Section 4.2.4), or
  - successfully validated (see Section 4.3).

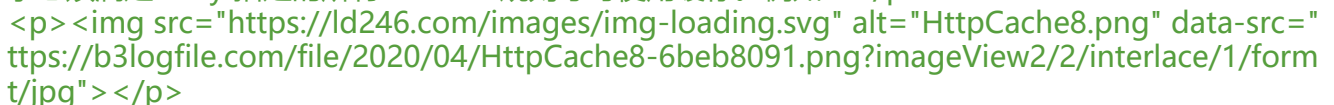
**大概解释一下:**

发起请求时, 满足一下条件即可使用缓存:

- URI 匹配, 如果一个 URI 有多个缓存, 则使用时间最近的
- 缓存的响应允许我们当前请求的 METHOD 使用缓存
- 头部匹配, 指缓存的响应中, Vary 指定的头部必须与本次请求的头部匹配
- 提交的请求不包含 "no-cache" 头部 (Pragma 和 Cache-Control 都不可包含)
- 缓存的响应不包含 "no-cache" 头部 (Pragma 和 Cache-Control 都不可包含)
- 缓存未过期, 或允许使用过期缓存(max-stale), 或针对过期缓存已经到源服务器成功验证(源服务器响应 304)

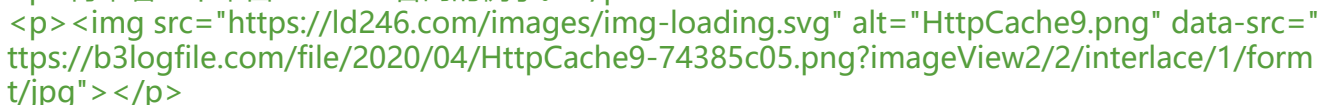
### Varying Response

这里对命中缓存时提到的头部匹配做一个介绍。Vary 是 Http 响应头的一个 Header, 他决定了一个请求在命中缓存时的 Header 匹配规则。当缓存的响应中指定了 Vary 时, 新的请求必须满足 Vary 指定的所有 Header 规则才可使用缓存。例如:



上图中的这个 js 资源 Response 中的 Vary 指明, 想要使用我这个缓存, 必须验证 Accept-Encoding 头的值, 再看 Request Header 中, 请求的 Accept-Encoding 头的值为 gzip, deflate。所以, 新的请求想要命中这个缓存, 也必须带这个头值能够匹配才行。

再来看一个来自 Mozilla 官网的例子。



Client1 向代理服务器 Cache 发起/doc 的请求, Cache 发现没有缓存, 向源服务器 Server 请求, Server 将结果响应给 Cache, 并带有一个 Vary 头部, 指定要校验 Content-Encoding 头部, 值为 gzip 的请求才能使用这份共享缓存。然后再响应给 Client1。

Client2 向代理服务器 Cache 发起/doc 请求, 并带有 Accept-Encoding 头值为 br。Cache 服务器跟本地缓存校对, 发现 Vary 校验不通过, 因此不能使用共享缓存, 再次向源服务器发起请求, 此时源服务器会返回响应, 并将请求的 br 加到限制条件中。再返回给 Client2。

Client3 向代理服务器 Cache 发起/doc 请求, 并带有 Accept-Encoding 头值为 br。Cache 服务器跟本地缓存校对, 发现 Vary 中指定的值包含 br, 校验通过, 直接返回共享

存, 不再请求源服务器。</p>

<h2 id="参考">参考</h2>

<ul>

<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Ftime.geekbang.org%2Fcourse%2Fintro%2F100026801%3Futm\_term%3Dpc\_interstitial\_259" target="\_blank" rel="nofollow ugc">极客时间 - Web 协议详解与抓包实战</a> </li>

<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fwww.rfc-editor.org%2Frfc%2Fpdf/rfc%2Frfc7234.txt.pdf" target="\_blank" rel="nofollow ugc">RFC7234</a> </li>

<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fdeveloper.mozilla.org%2Fen-US%2Fdocs%2FGlossary%2Fcacheable" target="\_blank" rel="nofollow ugc">Cacheable</a> </li>

<li> <a href="https://ld246.com/forward?goto=https%3A%2F%2Fdeveloper.mozilla.org%2Fen-US%2Fdocs%2FWeb%2FHTTP%2FCaching" target="\_blank" rel="nofollow ugc">Http Caching</a> </li>

</ul>