Web缓存

1. 浏览器缓存
2. 页面缓存（动态页面转成静态页面）
3. 数据库缓存
4. nginx反向代理缓存

**预备知识：**

http缓存：

私有缓存：浏览器缓存

共享缓存：代理缓存

常见的 HTTP 缓存只能存储 [GET](https://developer.mozilla.org/zh-CN/docs/Web/HTTP/Methods/GET) 响应，

http请求头

cache-control 在请求头或响应头中实现缓存机制

指令：public 可以被任何对象缓存，共享缓存（如客户端、代理服务器）

Private 只能被单个用户缓存即代理服务器无法缓存，不使用与共享缓存 （默认）

No-cache 强制缓存用户，使用缓存数据前发送验证到原始服务器

Only if cache缓存存在的话，就使用缓存

Max-age缓存存储周期，相对于请求时间与expires相反，单位秒

s-max-age

max-age" 或者 "s-max-age"使用后expires请求头会被忽略。

缓存静态资源：Cache-Control:public, max-age=31536000

If-modified-since：是一个条件式请求首部，为资源上一次修改的时间，服务器会根据这个携带时间返回200或者304，响应中会在**Last-Modified**携带上次修改的时间。只在get的请求中。**If-None-Match**存在时该请求头失效。

If-None-Match：在get请求中，服务器中资源的etag与该首部列出的值相匹配时返回200，验证失败时返回304，引发服务器状态改变时返回412，需要注意的是，服务器端在生成状态码为 304 的响应的时候，必须同时生成以下会存在于对应的 200 响应中的首部：Cache-Control、Content-Location、Date、**ETag**（资源标识：验证资源是否改变）、Expires 和 Vary 。

**浏览器缓存**

**浏览器是如何判断缓存是否过期？**

应该是根据Response Header里面的Cache-Control和Expires这两个属性，当两个都存在时，Cache-Control优先级较高。

**Cache-Control**

该字段用于控制浏览器在什么情况下直接使用本地缓存而不向服务器发送请求。一般具有以下值：

Public：指示响应可被任何缓存区缓存。

Private：指示对于单个用户的整个或部分响应消息，不能被共享缓存处理。这允许服务器仅仅描述当用户的部分响应消息，此响应消息对于其他用户的请求无效。

no-cache：指示请求或响应消息不能缓存。

no-store：用于防止重要的信息被无意的发布。在请求消息中发送将使得请求和响应消息都不使用缓存。

max-age：指示浏览器可以接收生存期不大于指定时间（以秒为单位）的响应。

min-fresh：指示浏览器可以接收响应时间小于当前时间加上指定时间的响应。

max-stale：指示浏览器可以接收超出超时期间的响应消息。如果指定max-stale消息的值，那么浏览器可以接收超出超时期指定值之内的响应消息。

PS：其实主要关注max-age这个值就行了。

**Expires**

Expires 头部字段提供一个日期和时间，在该日期前的所有对该资源的请求都会直接使用浏览器缓存而不用向服务器请求。

例如：

Expires: Sun, 08 Nov 2009 03:37:26 GMT

注意：

1. cache-control max-age 和 max-stale将覆盖Expires header。

2. 使用Expires存在服务器端时间和浏览器时间不一致的问题。

3. 另外有人说Expires 是HTTP 1.0的东西，现在默认浏览器均默认使用HTTP 1.1，所以它的作用基本忽略（暂不明了）。

**服务端如何判断缓存已失效？**

服务端通过If-Modified-Since（Last-Modified）和If-None-Match（Etag）这两个属性的值来判断缓存是否失效的。

**Last-Modified/If-Modified-Since**

Last-Modified/If-Modified-Since要配合Cache-Control使用。

Last-Modified：响应资源的最后修改时间。

If-Modified-Since：当缓存过期时，发现资源具有Last-Modified声明，则在请求头带上If-Modified-Since（值就是Last-Modified）。服务器收到请求后发现有头If-Modified-Since则与被请求资源的最后修改时间进行比对。若最后修改时间较新，说明资源又被改动过，则响应HTTP 200整片资源内容（写在响应消息包体内）;若最后修改时间较旧，说明资源无新修改，则响应HTTP 304，告知浏览器继续使用所保存的cache。

**Etag/If-None-Match** If-None-Match的优先级高于**If-Modified-Since**

Etag/If-None-Match也要配合Cache-Control使用。

Etag：资源在服务器的唯一标识（生成规则由服务器决定）。Apache中，ETag的值，默认是对文件的索引节（INode），大小（Size）和最后修改时间（MTime）进行Hash后得到的。

If-None-Match：当缓存过期时，发现资源具有Etage声明，则在请求头带上If-None-Match（值就是Etag）。服务器收到请求后发现有头If-None-Match 则与被请求资源的相应校验串进行比对，决定返回200或304。

**为什么有了Last-Modified还要Etag？**

Etag的出现主要是为了解决几个Last-Modified比较难解决的问题：

1. Last-Modified标注的最后修改只能精确到秒级，如果某些文件在1秒钟以内，被修改多次的话，它将不能准确标注文件的修改时间。

2. 如果某些文件会被定期生成，当有时内容并没有任何变化，但Last-Modified却改变了，导致文件没法使用缓存。

3. 有可能存在服务器没有准确获取文件修改时间，或者与代理服务器时间不一致等情形。

