的对象副本在缓存上已经有了：

```
If-None-Match: "v2.6"
If-None-Match: "v2.4","v2.5","v2.6"
If-None-Match: "foobar","A34FAC0095","Profiles in Courage"
```

7.8.7 强弱验证器

缓存可以用实体标签来判断，与服务器相比，已缓存版本是不是最新的（与使用最近修改日期的方式很像）。从这个角度来看，实体标签和最近修改日期都是*缓存验证器*（cache validator）。

有时，服务器希望在对文档进行一些非实质性或不重要的修改时，不要使所有的已缓存副本都失效。HTTP/1.1 支持“弱验证器”，如果只对内容进行了少量修改，就允许服务器声明那是“足够好”的等价体。

只要内容发生了变化，强验证器就会变化。弱验证器允许对一些内容进行修改，但内容的主要含义发生变化时，通常它还是会变化的。有些操作不能用弱验证器来实现（比如有条件地获取部分内容），所以，服务器会用前缀“W/”来标识弱验证器。

```
ETag: W/"v2.6"
If-None-Match: W/"v2.6"
```

不管相关的实体值以何种方式发生了变化，强实体标签都要发生变化。而相关实体在语义上发生了比较重要的变化时，弱实体标签也应该发生变化。

注意，原始服务器一定不能为两个不同的实体重用一个特定的强实体标签值，或者为两个语义不同的实体重用一个特定的弱实体标签值。不管过期时间是多少，缓存条目都可能会留存任意长的时间，因此，假设缓存不会再次通过它在过去某个时刻获得的验证器，对一个条目进行验证是不合适的。

7.8.8 什么时候应该使用实体标签和最近修改日期

如果服务器回送了一个实体标签，HTTP/1.1 客户端就必须使用实体标签验证器。如果服务器只回送了一个`Last-Modified`值，客户端就可以使用`If-Modified-Since`验证。如果实体标签和最后修改日期都提供了，客户端就应该使用这两种再验证方
181 案，这样 HTTP/1.0 和 HTTP/1.1 缓存就都可以正确响应了。

除非 HTTP/1.1 原始服务器无法生成实体标签验证器，否则就应该发送一个出去，如果使用弱实体标签有优势的话，发送的可能就是个弱实体标签，而不是强实体标签。而且，最好同时发送一个最近修改值。