链滴

OkHttp 中文文档之 Calls(原创汉化搬运)

作者：washmore

原文链接：https://ld246.com/article/1506671986849

来源网站：链滴

The HTTP client’ s job is to accept your request and produce its response. This is simple in th ory but it gets tricky in practice.

Http client 的作用就是接收你的请求然后提供对此的响应,这从原理来说很简单,但是实践起来却比较手.

Requests(请求)

Each HTTP request contains a URL, a method (like GET or POST), and a list of headers. Reques s may also contain a body: a data stream of a specific content type.

每个Http request都包含一个URL,一个method(诸如GET 或者 POST之类),以及一系列的headers.有请求或许还包含reqeustBody(针对不同的content type有不同的数据流格式)

Responses(响应)

The response answers the request with a code (like 200 for success or 404 for not found), hea ers, and its own optional body.

response是对request的应答,包含一个code(比如200代表请求成功,或者404代表请求资源未找到),响头headers,以及可选的response body内容.

Rewriting Requests(重写请求)

When you provide OkHttp with an HTTP request, you’ re describing the request at a high-lev l: “fetch me this URL with these headers.” For correctness and efficiency, OkHttp rewrites your re uest before transmitting it.

当你使用OkHttp发起一个HTTP request时,代表你在描述一个高标准的request: “给我用这些headers定这个URL” 正确且高效,(为了达到这样的效果)OkHttp将会在真正发起这个request前重写它

(**注:王婆卖瓜orz**)

OkHttp may add headers that are absent from the original request, including Content-Length, ransfer-Encoding, User-Agent, Host, Connection, and Content-Type. It will add an Accept-Enc ding header for transparent response compression unless the header is already present. If yo ’ ve got cookies, OkHttp will add a Cookie header with them.

OkHttp会额外添加一些原始request中缺失的headers,包括Content-Length, Transfer-Encoding, Us r-Agent, Host, Connection, 以及 Content-Type.还会添加一个叫Accept-Encoding的header用于明压缩响应内容,除非这个requestr之前就已经指定了这个header.如果你获得了cookies,那么OkHttp会添加一个Cookie的header,内容就是你获得的这些cookies.(**注:这里应该是指从上下文环境中拿到的ookies,比如我刚刚用OKhttp发起一次请求并正确响应,那从这个响应里面是能知道cookies内容的,一次请求就能用上**)

Some requests will have a cached response. When this cached response isn’ t fresh, OkHttp an do a conditional GET to download an updated response if it’ s newer than what’ s cached. This requires headers like If-Modified-Since and If-None-Match to be added.

有些请求或许会得到一份缓存过的响应,当这些缓存过的响应不刷新,而且未缓存的响应内容比缓存过要新的话,OkHttp将会进行一次 有条件的 下载更新响应内容.当然,这个功能需要添加If-Modified-Since或者 If-None-Match这类的header才能生效(**注:这里是指遇到这种场景OKhttp会自动添加If-Modifid-Since 或者 If-None-Match来进行处理,还是说需要手动添加If-Modified-Since 或者 If-None-**

atch这样的header,OKHttp才会做这样的处理,我暂时还不敢确认,后续尝试过来再来落实,或许有已确认过的大佬在留言告知)

Rewriting Responses(重写响应)

If transparent compression was used, OkHttp will drop the corresponding response headers ontent-Encoding and Content-Length because they don’ t apply to the decompressed respo se body.

如果透明压缩规则被使用(**注:见前文**),OKHttp就会丢弃掉响应中的Content-Encoding and Content-L ngth这两个headers,因为他们无法正确反映解压过的响应内容(**注:偏小**)

If a conditional GET was successful, responses from the network and cache are merged as dir cted by the spec.

如果一个有条件的GET请求被执行,那么responses也会按照这个条件规则进行修改.

Follow-up Requests(302请求)

When your requested URL has moved, the webserver will return a response code like 302 to i dicate the document’ s new URL. OkHttp will follow the redirect to retrieve a final response.

如果你的requested URL被转移,那么web服务器就回返回一个响应code302来给你指定一个新的请求址,OkHttp将会重新请求这个重定向过的URL来获得最终的response.

If the response issues an authorization challenge, OkHttp will ask the Authenticator (if one is onfigured) to satisfy the challenge. If the authenticator supplies a credential, the request is ret ied with that credential included.

如果response需要权限验证,OkHttp就会向Authenticator(如果配置过的话) 申请身份凭证来尝试通验证,如果认证器能够提供凭证的话,OkHttp就会带着这个凭证尝试重新请求.

Retrying Requests(重试请求)

Sometimes connections fail: either a pooled connection was stale and disconnected, or the w bserver itself couldn’ t be reached. OkHttp will retry the request with a different route if one i available.

有些时候connections会失败:要么是connections死了断开,要么是这个服务器节点本身就不可访问了, kHttp就会重新尝试请求另外可用的route节点.

Calls(调用)

With rewrites, redirects, follow-ups and retries, your simple request may yield many requests nd responses. OkHttp uses Call to model the task of satisfying your request through however many intermediate requests and responses are necessary. Typically this isn’ t many! But it’ s omforting to know that your code will continue to work if your URLs are redirected or if you f ilover to an alternate IP address.

因为以上这些情况,一个简单的request也许都会产生很多requests和responses,OkHtt用了Call这样种模型来出色的完成这样需要多个中间请求和响应的工作.典型的手段不多,但是能让你的代码在URL转移或者某个服务器节点发生故障的情况下也能继续正常工作.(**注:2333又开始自夸了/doge**)

Calls are executed in one of two ways:

- **Synchronous:** your thread blocks until the response is readable.
- **Asynchronous:** you enqueue the request on any thread, and get called back on another thr ad when the response is readable.

Calls 包含两种方式:

- **同步的:** 你的线程将被阻塞直到响应内容可读
- **异步的:** 你把请求推到其他线程的队列中,然后当响应内容可读之后在另外的线程上通过回调函数 call d back来获取

Calls can be canceled from any thread. This will fail the call if it hasn’ t yet completed! Code t at is writing the request body or reading the response body will suffer an IOException when it call is canceled.

Calls 能在任何线程被取消,如果请求没有完成,被取消后这个请求就将会失败! 如果代码正在执行reques body写操作或者response body读操作,那么此时取消Calls的行为将会抛出一个IOException异常.

Dispatch(分配)

For synchronous calls, you bring your own thread and are responsible for managing how man simultaneous requests you make. Too many simultaneous connections wastes resources; too ew harms latency.

对于同步调用来说,再多同时发起的requests你也能在该线程上妥当管理.(要注意的是)太多simultaneo s connections将会耗费大量的系统资源,而太少的simultaneous connections则将会让处理速度变慢 从而带来更高的延迟

For asynchronous calls, Dispatcher implements policy for maximum simultaneous requests. Y u can set maximums per-webserver (default is 5), and overall (default is 64).

对于异步调用来说,Dispatcher可以设置simultaneous requests的限制,你可以设置针对单个服务器的 发连接上限(默认是5),也可以设置整体最大上限(默认是64)

译:洗澡狂魔,原文wiki地址:https://github.com/square/okhttp/wiki/Calls