net/http

transport

Transport

属性成员

idleConn空闲链接字典，最近用过的在队尾

key： proxy, schema, addr, onlyH1

value: []*persistConn

idelConnWait等待获取链接

idleLRU connLRU

方法

roundTrip =

循环重试，
1. 获取链接pconn = getConn，
2. 发送请求pconn.roundTrip(treq)
3. 收尾

getConn获取一个pc链接

生成一个wantConn，先去尝试空闲队列获取queueForIdleConn，如果不成功，则生成一个新的queurForDial

queueForIdleConn
从idleConn中获取一个pconn，填装到wantConn中

queueForDial 生成一个新链接，填装到wantConn中

tryPutIdelConn 放回空闲链接表

dialConnFor 生成新链接

dialConn
1. 启动一个链接，tcp
2. go readLoop
3. go writeLoop

persistConn

alt(RoundTrip) for TLS

roundTrip

1. 构造writeRequest，通过writech触发writeLoop写事件

2. 构造requestAndChan，通过reqchan触发readLoop 读事件readResponse

3. 通过2中构造体中的一个管道resc来接受事件完成的提醒，从而接触本次调用

readLoop 循环读

获取请求结果通过roundTrip写入数据，这里读数据
1. readResponse
2. resp.bodyIsWriteble || !hasBody 代表响应结束 转3，否则转4
3. 通过管道传递响应 rc.ch <- responseAndError{res: resp}，放回空闲链接，continue
4. 通过管道传递响应，等待读取完毕，判活，等待50ms然后回收空闲队列（防止重用链接互相干扰）

响应中的body需要调用方主动去关闭

ReadResponse

writeLoop

通过roundTrip写入数据，这里读取数据并发送
主要调用Request.write方法发送数据。
请求的body代码中会关闭，无需适用方关心。

wantConn

want for a conn or error

被wantConnQueue管理维护

cancel释放当前的pc，结束等待

tryDeliver 尝试转给pc给w

只是一个信息传递状态同步的中间态工具

transfer

net

conn

真正的链接

底层fd（netFD）

poll.FD