官网参考：<http://hc.apache.org/httpclient-3.x/tutorial.html>

<http://hc.apache.org/httpcomponents-core-ga/httpcore/apidocs/org/apache/http/protocol/HttpRequestExecutor.html>

编译时指定编码格式(防止中文乱码)：E:\deleting\java\httpclient>javac -encoding utf-8 TestHttpClient.java

使用方法

使用HttpClient发送请求、接收响应很简单，一般需要如下几步即可。

1. 创建HttpClient对象。

2. 创建请求方法的实例，并指定请求URL。如果需要发送GET请求，创建HttpGet对象；如果需要发送POST请求，创建HttpPost对象。

3. 如果需要发送请求参数，可调用HttpGet、HttpPost共同的setParams(HetpParams params)方法来添加请求参数；对于HttpPost对象而言，也可调用setEntity(HttpEntity entity)方法来设置请求参数。

4. 调用HttpClient对象的execute(HttpUriRequest request)发送请求，该方法返回一个HttpResponse。

5. 调用HttpResponse的getAllHeaders()、getHeaders(String name)等方法可获取服务器的响应头；调用HttpResponse的getEntity()方法可获取HttpEntity对象，该对象包装了服务器的响应内容。程序可通过该对象获取服务器的响应内容。

6. 释放连接。无论执行方法是否成功，都必须释放连接

创建对象🡪get或者post方法🡪设置参数🡪获取结果信息🡪关闭连接资源

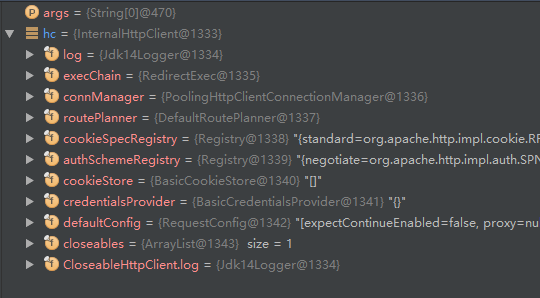
创建对象：CloseableHttpClient hc=null;

HttpResponse response;

创建对象过程会抛出异常，因此要放在try-catch中

使用httpclients的静态方法createdefault()来创建默认对象

默认设置如下：

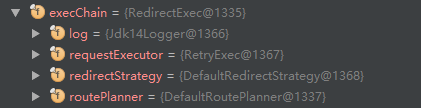


参数解析：

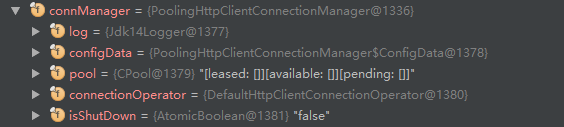
Log——是apache-jar包自带的日志记录器



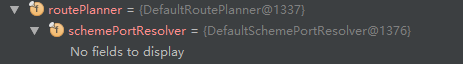
ExecChain——执行链



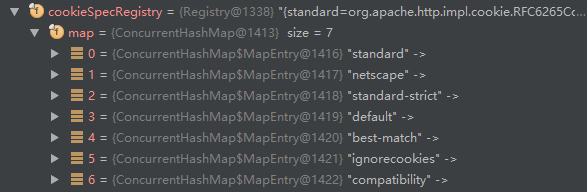
ConnManager——连接管理器



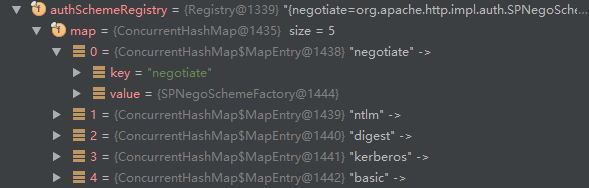
Routplanner——路线计划



CookieSpecRegistry——特定cookies的记录



AuthschemeRegistry——认证注册



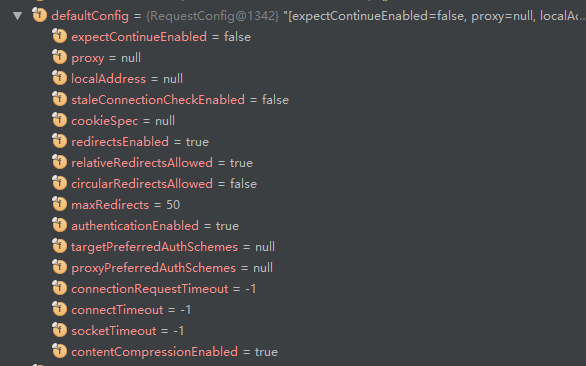
Cookiestore——cookie的存储



Credentialsprovider——提供的凭据



Defaultconfig——默认的配置



Closeables——可关闭的



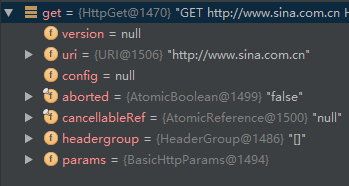
CloseableHttpclient.log——可关闭的http客户端日志



使用GET方法获得网页数据：

HttpGet get=new HttpGet("http://www.sina.com.cn");

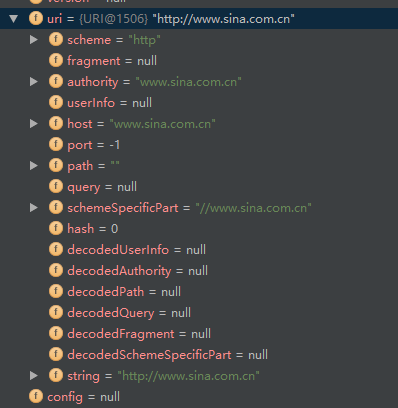
Get如下所示：



参数说明：

Version——版本

Uri——唯一资源标识符



Config——构建此get的配置信息

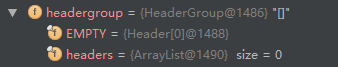
Aborted——中止



CancellableRef——



Headergroup——请求头信息



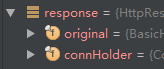
Parameters——参数



执行get获取网页内容：

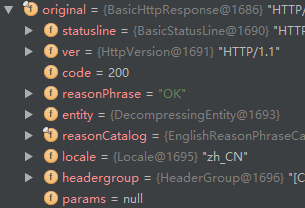
response=hc.execute(get);

response如下

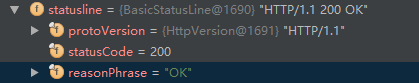


参数说明如下：

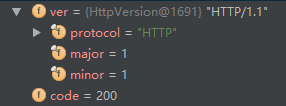
Original——原始的



statusLine——状态信息

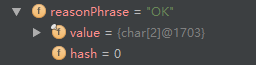


Ver——http的版本

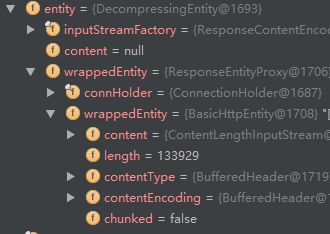


Code——状态码

ReasonParse——状态解析

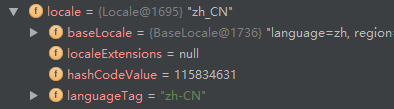


Entity——获取的网页实体

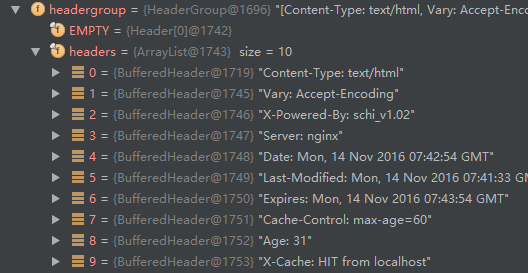


wrappedRnity表示是压缩的实体信息，它下面的类似头信息

locale——本地化信息

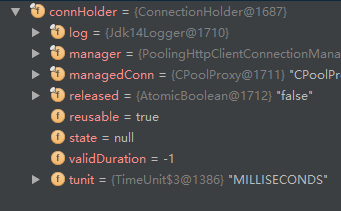


Headergroup——响应的头信息



Params——参数

ConnHolder——连接保持信息



读取网内entity的不同方式：

//一次读取一个字符(两个字节）  
InputStreamReader isr=new InputStreamReader(entity.getContent());  
int ch;  
ch=isr.read();  
while(-1!=ch){  
 System.out.print((char)ch);  
 ch=isr.read();  
}  
isr.close();  
  
//一次读取多个字符  
InputStreamReader isr=new InputStreamReader(entity.getContent());  
char[] chars=new char[1024];  
int len=isr.read(chars);  
while(-1!=len){  
 System.out.print(new String(chars,0,len));  
  
  
 len=isr.read(chars);  
}  
isr.close();  
  
 //缓冲流是包裹流，要将实际的流在初始化时转换为包裹流  
//超快  
BufferedReader bfd=new BufferedReader(new InputStreamReader(entity.getContent()));  
String str=bfd.readLine();  
while(null!=str){  
 System.out.print(str);  
 System.out.println("");  
 str=bfd.readLine();  
}  
bfd.close();  
  
//这也比一个字节一个字节输块，缓冲区减少了读一个写一个  
BufferedReader bfd=new BufferedReader(new InputStreamReader(entity.getContent()));  
int ch;  
ch=bfd.read();  
while(-1!=ch){  
 System.out.print((char)ch);  
 ch=bfd.read();  
}  
bfd.close();  
\*/  
  
  
/\*  
//在中文情况下会乱码  
//一次读取一个字节  
InputStream is=entity.getContent();  
int ch;  
ch=is.read();  
while(-1!=ch){  
 System.out.print((char)ch);  
 ch=is.read();  
}  
is.close();  
  
//一次读取多个字节  
InputStream isr=entity.getContent();  
byte[] bytes=new byte[1024];  
int len=isr.read(bytes);  
while(-1!=len){  
 System.out.write(bytes,0,len);  
 len=isr.read(bytes);  
}  
  
//缓冲流是包裹流，要将实际的流在初始化时转换为包裹  
//更快  
BufferedInputStream bfs=new BufferedInputStream(entity.getContent());  
byte[] bytes=new byte[1024];  
int len=bfs.read(bytes);  
while(-1!=len){  
 System.out.print(new String(bytes,0,len));  
 len=bfs.read(bytes);  
}  
  
 bfs.close();

Get方法自动处理重定向，因此最终获得的是重定向后的网页内容，而post方法可以得到重定向信息。

System.*out*.println(response.getFirstHeader("Location").getValue());

得到状态码：

response.getStatusLine().getStatusCode()

请求准备阶段操作httpget或者httppost或者httpclient,其中get和post是用来设置参数的，httpclient主要是用来执行get或者post的方法的

响应阶段操作response