一、组员及分工

* 吴秦恺 项目架构
* 刘星驰 项目主体
* 谢沛东 项目功能点
* 陈凯悦 测试、文档

二、功能描述

项目实现了HTTP的请求和响应，完成了要求的全部功能点，包括：

1. HTTP客户端（命令行）可以发送请求报文、呈现响应报文

2. HTTP客户端对301、302、304的状态码做相应的处理

3. HTTP服务器端支持200、301、302、304、404、405、500的状态码

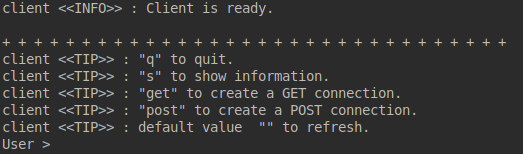
4. HTTP服务器端实现长连接

5. MIME支持文本、图片、音频等多种类型

下面给出一个示例：

首先，启动HTTP客户端和服务器端，均出现提示：

客户端

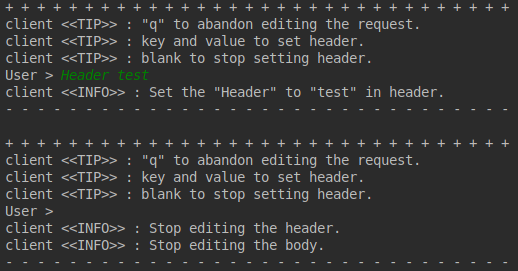
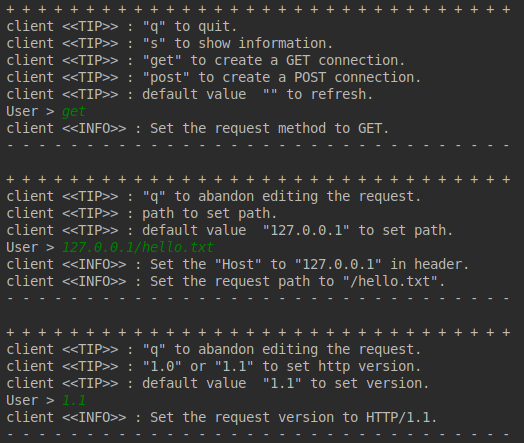


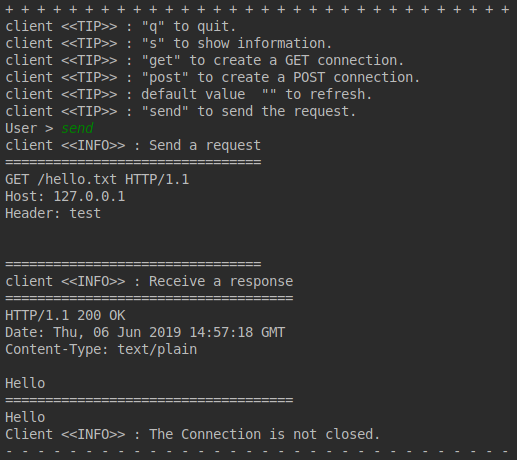
服务器端

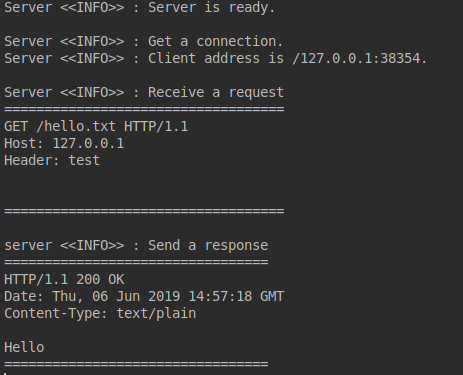


依次输入get发起GET请求，输入127.0.0.1/hello.txt确定URL，输入1.1确定HTTP版本，输入“Header test”添加头部键值对，回车结束编辑头部（由于get请求一般没有主体，结束编辑头部同时结束编辑主体），输入send发送请求。此时我们可以看到：

客户端





服务器端

HTTP请求响应成功。

三、关键数据逻辑

1. 项目界面

HTTP客户端在每个请求响应流程中，会显示三部分数据，首尾用”+“和”-“作为分割线

第一部分是输入提示，开头以”client <<TIP>> :“标识，会提示下一步输入指令。

第二部分是用户本身的输入。

第三部分是输出信息，一般会显示请求报文、响应报文、连接信息等，展示日志。

HTTP服务器端与客户端类似，而且更为简单直观，不予赘述。

2.请求报文

请求报文分为三部分：起始行、头部和主体。其中头部和主体间有”\r\n“的空行。

起始行包括三部分：请求方式、请求路径、HTTP版本。它们之间用空格隔开。

头部包含多行键值对，键和值之间用冒号隔开，键值对间以”\r\n“换行。

主体部分包含一个由任意数据组成的数据块。

3.响应报文

请求报文分也为三部分：起始行、头部和主体。其中头部和主体间也有”\r\n“的空行。

起始行包括三部分：HTTP版本、状态码、原因。它们之间用空格隔开。

头部包含多行键值对，键和值之间用冒号隔开，键值对间以”\r\n“换行。

主体部分包含一个由任意数据组成的数据块。

**响应报文的状态码**

200 OK    请求成功  
404 Not Found    服务器无法找到被请求的页面。  
405 Method Not Allowed    请求中指定的方法不被允许。  
500 Internal Server Error    请求未完成。服务器遇到不可预知的情况。

要做应对的状态码：

301 Moved Permanently    所请求的页面已经转移至新的url。  
302 Found    所请求的页面已经临时转移至新的url。  
304 Not Modified    未按预期修改文档。客户端有缓冲的文档并发出了一个条件性的请求（一般是提供If-Modified-Since头表示客户只想比指定日期更新的文档）。服务器告诉客户，原来缓冲的文档还可以继续使用。

应对做法：

**对于301与302：**

解释：

301与302都表示资源被移动，响应报文的头部(header)中增加Location属性，返回新的资源地址，响应的实体中应当包含指向新的URI的超链接及简短说明。。对于301，如果是get请求客户端应当自动向新地址发送以后的请求。对于302，如果是get请求客户端应当自动向原有地址发送以后的请求。

实现：

客户端应对两种状态码没有区别，总的来说，如果是get的话,对新地址再发一遍请求,post默认不做。实现的时候服务器端将有一个301map与一个302map来保存旧地址与新地址。

**对于304：**

解释：

在浏览器第一次请求某一个URL时，服务器端的返回状态会是200，内容是你请求的资源，同时有一个Last-Modified的属性标记此文件在服务期端最后被修改的时间，格式类似这样：

Last-Modified: Fri, 12 May 2006 18:53:33 GMT

客户端第二次请求此URL时，根据 HTTP 协议的规定，浏览器会向服务器传送 If-Modified-Since 报头，询问该时间之后文件是否有被修改过：

If-Modified-Since: Fri, 12 May 2006 18:53:33 GMT

如果服务器端的资源没有变化，则自动返回 HTTP 304 （Not Modified.）状态码，内容为空，这样就节省了传输数据量。当服务器端代码发生改变或者重启服务器时，则重新发出资源，返回和第一次请求时类似。从而保证不向客户端重复发出资源，也保证当服务器有变化时，客户端能够得到最新的资源。

HTTP/1.1引入了Etag(Entity Tags).Etag仅仅是一个和文件相关的标记，可以是一个版本标记,比如说v1.0.0或者说"2e681a-6-5d044840"这么一串看起来很神秘的编码。但是HTTP/1.1 标准并没有规定Etag的内容是什么或者说要怎么实现，唯一规定的是Etag需要放在""内。

ETag: "50b1c1d4f775c61:df3"

客户端的查询更新格式是这样的：

If-None-Match: "50b1c1d4f775c61:df3"

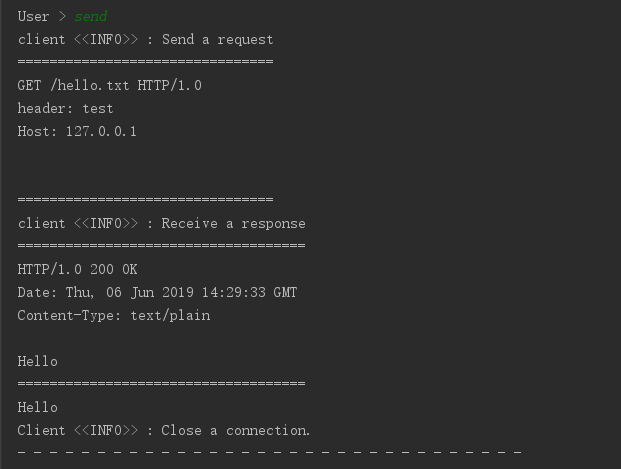
实现：

实现时客户端发送请求，接受了200的状态后，看头部中是否有Last-Modified，Etag，有的话将资源关联在map里，下次发送的时候在头部中加If-Modified-Since与If-None-Match。

**四、DEMO:关键流程与结果显示**

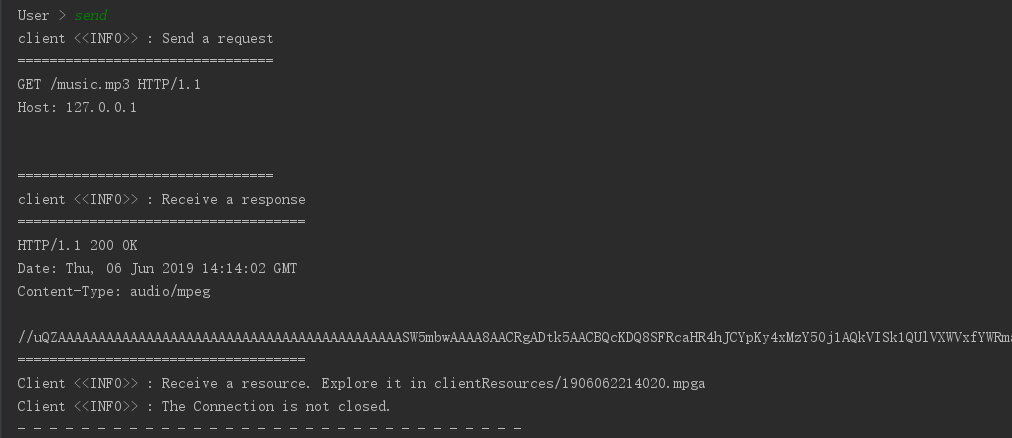
**1测试200，http1.0，短连接**

get  
127.0.0.1/hello.txt  
1.0  
Header test  
​  
send



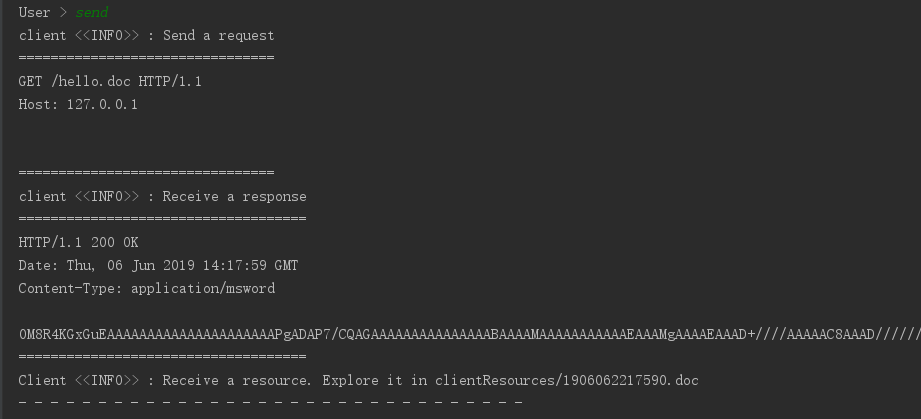
**2测试MIME（mp3）**

get  
127.0.0.1/music.mp3  
​  
​  
send



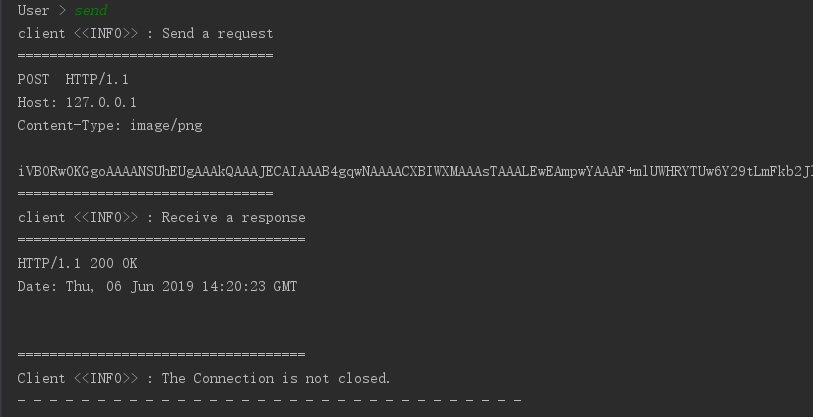
**3使用长连接、MIME（doc文档）**

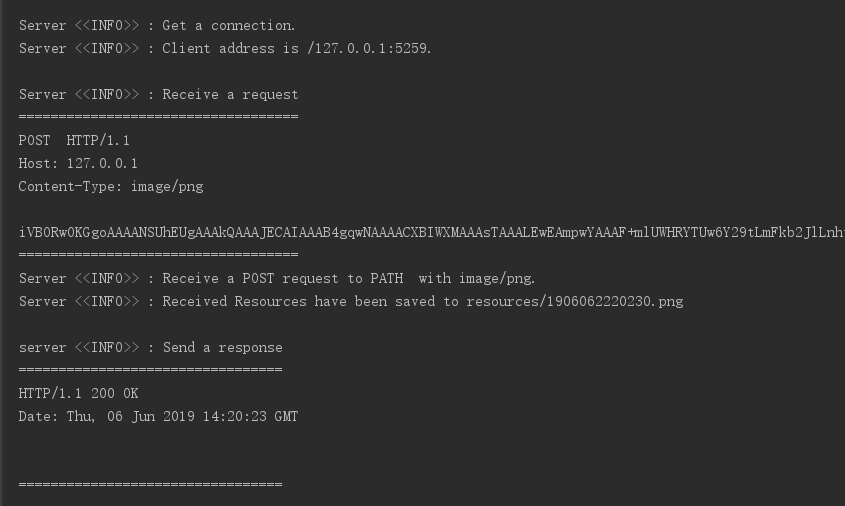
1  
/hello.doc  
​  
send



**4测试POST，MIME（png）**

post  
​  
​  
​  
b  
/DeepColor.png  
send





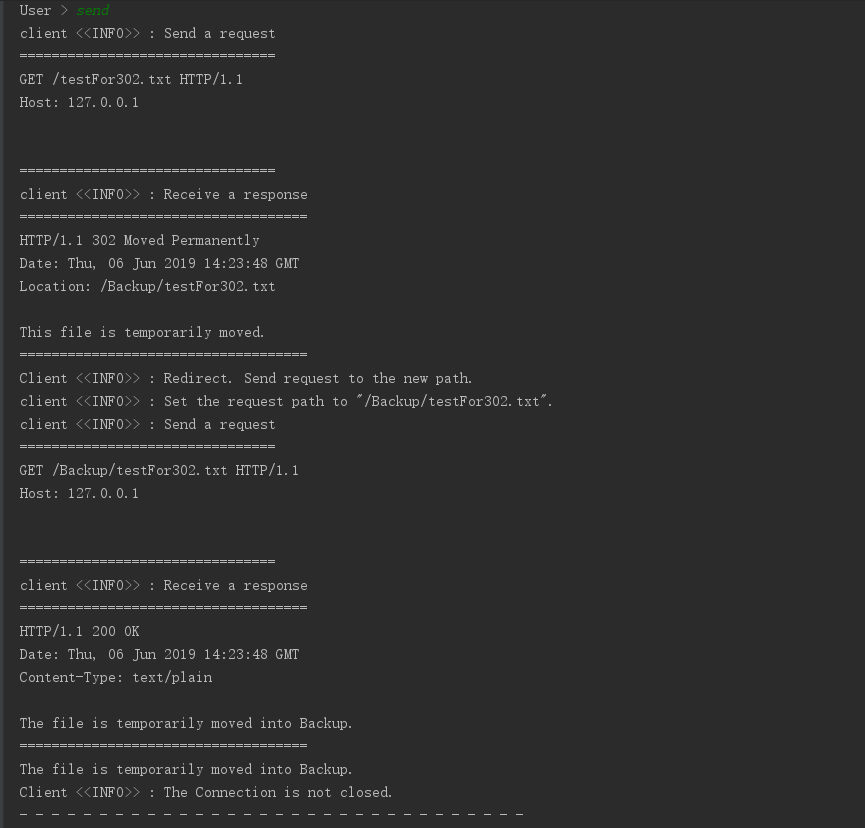
**5测试301**

get  
/testFor301.txt  
​  
​  
send



**6测试302**

get  
/testFor302.txt  
​  
​  
send



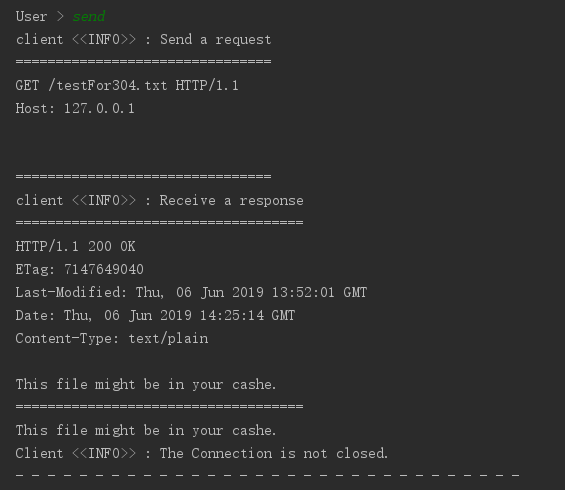
**7测试304**

第一次请求

get

/testFor304.txt

Send

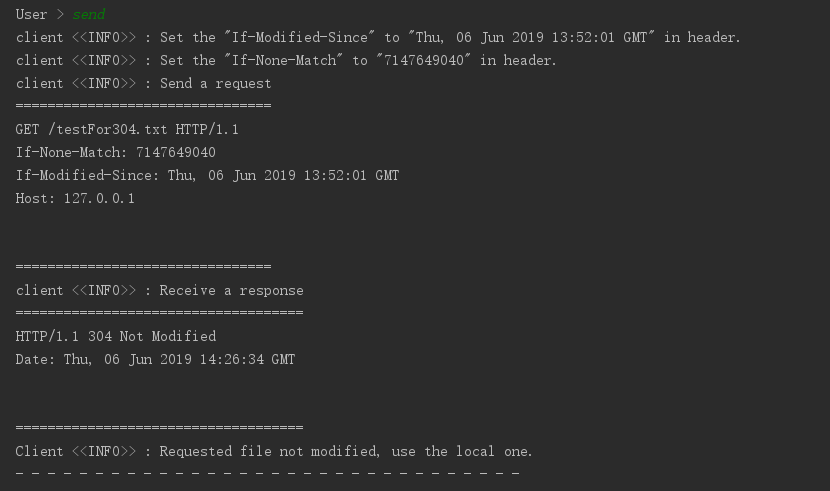


第二次请求

get

/testFor304.txt

Send

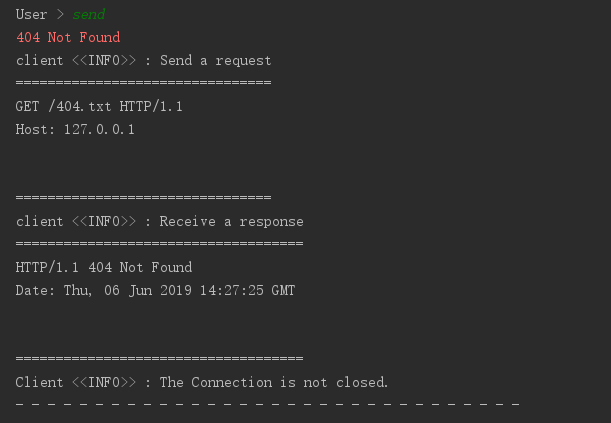


**8测试404**

get

/404.txt

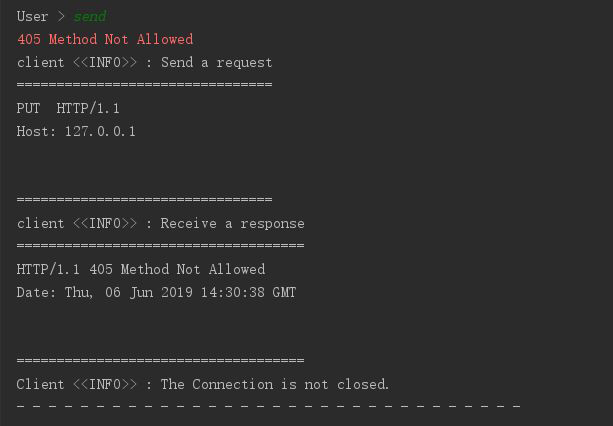
Send



**9测试405**

put

send



**10测试500**

get  
/error  
​  
​  
send

