数据通信作业-3

姓名：费扬 学号：519021910917 日期：2022.04.20

一、 实验名称及内容

Assignment3：*WebClient&WebServer*

使用 winsock 编程，主要目标任务为：

Task1: Write a client program to execute a single HTTP GET to a Web server.

即编写一个客户端程序来执行一个到Web服务器的HTTP GET。

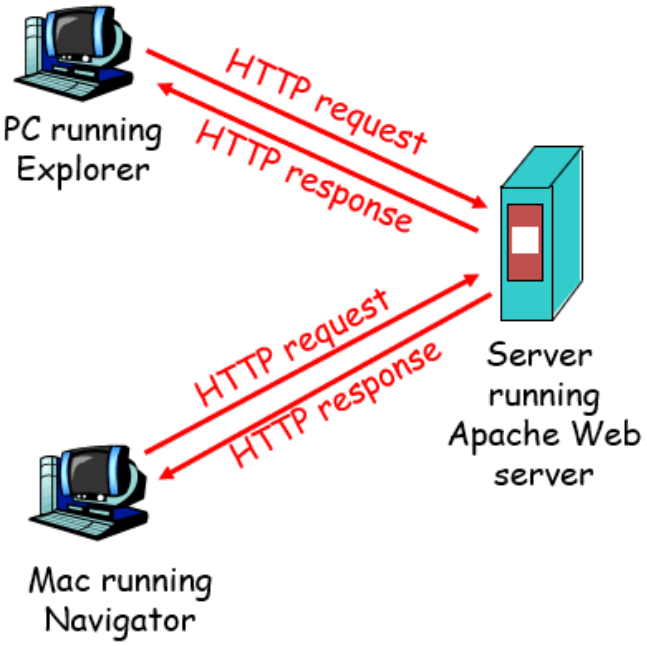
Task2: Write a A TCP server. Accept a single connection from a web browser. Port 80. Responds with an HTML message.

即编写一个TCP服务器接受来自web浏览器的单个连接。80号端口，以HTML消息进行响应。

Task3: Write an extended Web server for Windows, Handles multiple clients. Serves HTML, text, and GIF images.

即为Windows编写扩展Web服务器处理多个客户端，提供HTML、文本和GIF图像

二、 实验过程和结果



Http 协议原理

Task1：

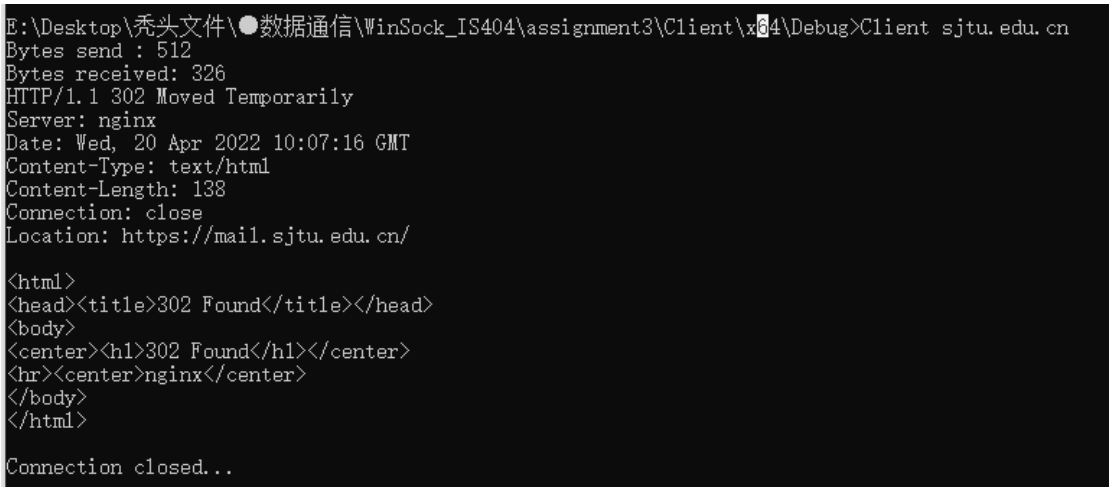
基本操作步骤：基本框架类似 TCPClient，在其基础上修改一点点用于 http 协议：

WebClient：

1. Initializes Winsock.
2. Creates a socket.
3. Connects to the server.
4. Sends and receives data.
5. Disconnects.

代码见打包文件中 Client.cpp，不予展示：

之后在命令行操作输出：



在 cmd 中访问百度和 sjtu 官网（这里还没改 host）

Task2：

WebServer：

1. Initialize Winsock
2. Create a socket
3. Bind the socket
4. Listen on the socket for a client
5. Accept a connection from a client
6. Receive and send data (minor change)
7. Disconnect.

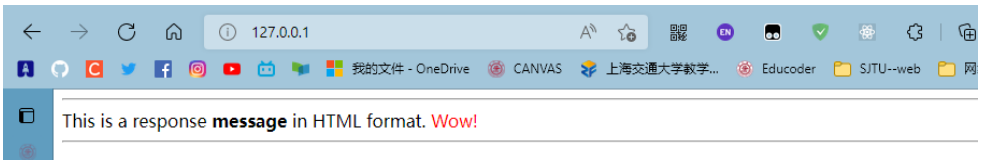
```
char outbuf[DEFAULT_BUFLEN];
int recvbuflen = DEFAULT_BUFLEN;
char recvbuf[DEFAULT_BUFLEN];

iResult = recv(ConnectSocket, recvbuf, recvbuflen, 0);
for (int i = 0; i < iResult; i++) {
    cout << recvbuf[i];
}
cout << endl;
strcpy_s(outbuf, "<html><body><hr>This is a response <b>message</b> in HTML format. <font color=red>Wow!</

cout << outbuf << endl;
iResult = send(ConnectSocket, outbuf, strlen(outbuf) + 1, 0);//+1
cout << iResult << endl;
if (iResult == SOCKET_ERROR) {
    cout << "send failed with error : " << WSAGetLastError() << endl;
}
else {
    cout << "send finished." << endl;
}
iResult = shutdown(ConnectSocket, SD_SEND);
printf("%d", iResult);
closesocket(ConnectSocket);
WSACleanup();
```

有所变动的地方代码展示

代码见打包文件中 WebServer.cpp，不予展示：
之后在命令行操作输出，并可以在浏览器中输入 127.0.0.1 查看：




```
E:\Desktop\秃头文件\●数据通信\WinSock_IS404\assignment3\WebServer\x64\Debug>WebServer
socket conencted.
bind finished.
listen finished
192
accept finished .
GET / HTTP/1.1
Host: 127.0.0.1
Connection: keep-alive
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="100", "Microsoft Edge";v="100"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safar
i/537.36 Edg/100.0.1185.44
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchang
e;v=b3;q=0.9
Purpose: p
<html><body><hr>This is a response <b>message</b> in HTML format. <font color=red>Wow!</font><hr></body></html>
112
send finished.
0
E:\Desktop\秃头文件\●数据通信\WinSock_IS404\assignment3\WebServer\x64\Debug>
```

这里可以看到首先打开服务器，建立连接，传输信息，在浏览器中可以看到预先写好的 html 文件信息，在命令行中有相应反馈。

Task3：

WebServer2：

基本操作步骤：

1. 在 WebServer 的基础上修改为 WebServer2
2. Adding handle_get()
3. Initialize Winsock
4. Create a socket
5. Bind the socket
6. Main loop to listen, accept, and then spin-off a thread to handle the GET
7. Disconnect

```
100  void handle_get(void* in_arg)
101  {
102      unsigned int   client_s;             // Client socket descriptor
103      char           in_buf[BUF_SIZE];     // Input buffer for GET request
104      char           out_buf[BUF_SIZE];    // Output buffer for HTML response
105      int            fh;                   // File handle
106      int            buf_len;              // Buffer length for file reads
107      char           command[BUF_SIZE];    // Command buffer
108      char           file_name[BUF_SIZE];  // File name buffer
109      int            retcode;              // Return code
110      int          j;
111
112      // Set client_s to in_arg
113      client_s = (unsigned int)in_arg;
114
115      printf("thread %d... \n", client_s);
116      // Receive the (presumed) GET request from the Web browser
117      retcode = recv(client_s, in_buf, BUF_SIZE, 0);
118      printf("thread %d...received web request: \n", client_s);
119      for (j = 0; j < retcode; j++)
120          printf("%c", in_buf[j]);
121
122      // If the recv() return code is bad then bail-out (see note #3)
123      if (retcode <= 0)
124      {
125          printf("ERROR - Receive failed --- probably due to dropped connection \n");
126          closesocket(client_s);
127          _endthread();
128      }
129
130      // Parse out the command from the (presumed) GET request and filename
131      sscanf_s(in_buf, "%s %s \n", &command, BUF_SIZE, &file_name, BUF_SIZE);
132
133      // Check if command really is a GET, if not then bail-out
134      if (strcmp(command, "GET") != 0)
135      {
136          printf("ERROR - Not a GET --- received command = '%s' \n", command);
137          closesocket(client_s);
138          _endthread();
139      }
140
141      // It must be a GET... open the requested file
142      //  - Start at 2nd char to get rid of leading "\"
143      _sopen_s(&fh, &file_name[1], _O_RDONLY | _O_BINARY,
144          _SH_DENYNO, _S_IREAD | _S_IWRITE);
145
146      // If file does not exist, then return a 404 and bail-out
147      if (fh == -1)
```

进行了修改的地方：handle_get ()

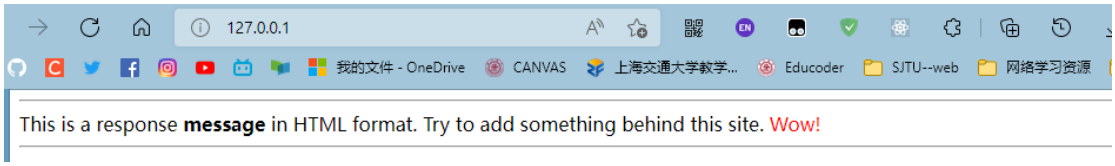
代码见打包文件中 WebServer2.cpp，不予展示：

之后在命令行操作启动 WebServer2：

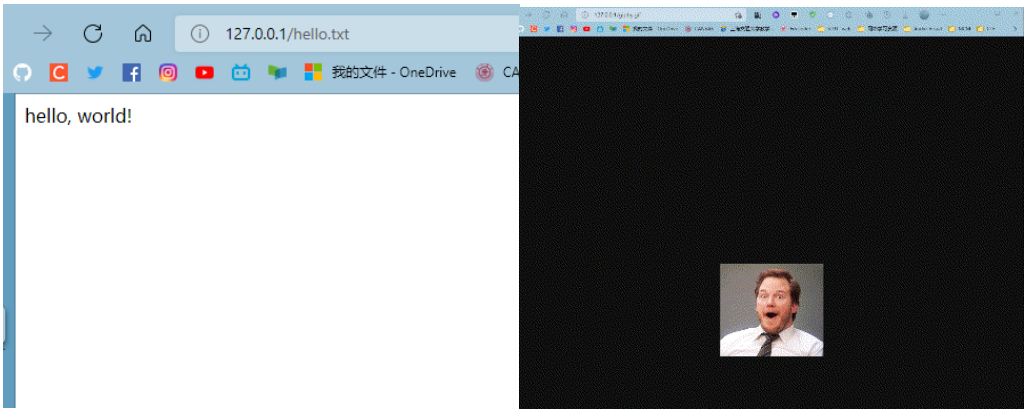
```
E:\Desktop\秃头文件\●数据通信\WinSock_IS404\assignment3\WebServer2\x64\Debug>WebServer2
main loop: linstening ...
client socket accepted, 200...
main loop: linstening ...
client socket accepted, 148...
main loop: linstening ...
thread 200...
thread 148...
thread 200...received web request:
GET / HTTP/1.1
Host: 127.0.0.1
Connection: keep-alive
Cache-Control: max-age=0
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="100", "Microsoft Edge";v="100"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safar
i/537.36 Edg/100.0.1185.44
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchang
e;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6

File '' not found --- sending an HTTP 404
```

这里可以看到在启动了服务器 WebServer2 后，再通过浏览器访问 127.0.0.1，有如上图的输出：



可以看到有固定的 html 输出（这里修改了本来的 FILE NOT FOUND）。



再在相应目录放入 hello.txt 和 giphy.gif 两个文件，访问 127.0.0.1/hello.txt 和 127.0.0.1/giphy.gif 效果如上：

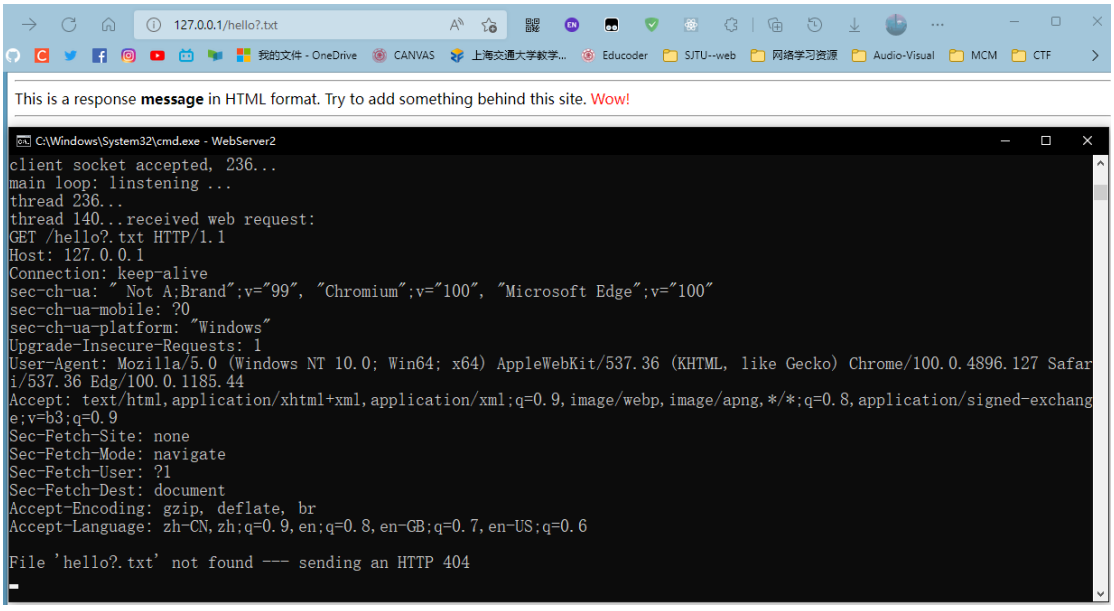
相应在命令行中查看输出：

```
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safar
i/537.36 Edg/100.0.1185.44
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchang
e;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6

Thread 144, ...Sending file 'giphy.gif'
Thread 144, ...comleted sending file 'giphy.gif'
socket 144 closed.
thread 144 ended.
thread 212...received web request:
ERROR - Receive failed --- probably due to dropped connection
client socket accepted, 76...
main loop: linstening ...
client socket accepted, 144...
thread 76...
main loop: linstening ...
thread 144...
thread 76...received web request:
GET /hello.txt HTTP/1.1
Host: 127.0.0.1
Connection: keep-alive
sec-ch-ua: " Not A;Brand";v="99", "Chromium";v="100", "Microsoft Edge";v="100"
sec-ch-ua-mobile: ?0
sec-ch-ua-platform: "Windows"
Upgrade-Insecure-Requests: 1
User-Agent: Mozilla/5.0 (Windows NT 10.0; Win64; x64) AppleWebKit/537.36 (KHTML, like Gecko) Chrome/100.0.4896.127 Safar
i/537.36 Edg/100.0.1185.44
Accept: text/html,application/xhtml+xml,application/xml;q=0.9,image/webp,image/apng,*/*;q=0.8,application/signed-exchang
e;v=b3;q=0.9
Sec-Fetch-Site: none
Sec-Fetch-Mode: navigate
Sec-Fetch-User: ?1
Sec-Fetch-Dest: document
Accept-Encoding: gzip, deflate, br
Accept-Language: zh-CN,zh;q=0.9,en;q=0.8,en-GB;q=0.7,en-US;q=0.6

Thread 76, ...Sending file 'hello.txt'
Thread 76, ...comleted sending file 'hello.txt'
socket 76 closed.
thread 76 ended.
```

可以发现在 Server2 中有查找到对应的文件并通过 web 显示无误。



如果查找不存在的文件则跟直接访问 127.0.0.1 效果一致，报错 404。

三、问题与思考

1. 直接 Client 访问 www.sjtu.edu.cn 的报错：

背景：DNS 解析协议，可以通过修改 C 盘下的 host 文件修复。

解释：这个问题上课已经解释了，直接访问 sjtu.edu.cn 是可以的，而访问 www.sjtu.edu.cn 时需要修改 host 文件即可。

2. 访问 www.baidu.com 时候有乱码：

猜测是编码问题，以 GBK 方式读取了 UTF-8 编码的中文。不影响 http 正常获取。

名称	示例	特点	产生原因
古文码	鐢辨湀瑕佸ソ濂藉涔犲ぉ澶╁悜涓?	大都为不认识的古文，并加杂日韩文	以GBK方式读取UTF-8编码的中文

3. 10013 报错：

经查阅，发现是端口被占用，改用 8080 即可。清空各占用端口或是重启也可以生效。(使用 8080 时访问 127.0.0.1:8080)