Web Client

Mir-lab

What to do

I. Make your Web Client and Report

- Connect Webserver with your Webserver
- Capture your Webserver and WebClient
- Make a Report with your code explanation, program pictures and Feelings.

II. Mark with an automatic scoring program (Reference Auto Marking Program Guideline_WEB Client)

- Mission1 : Handle USER-AGENT in HTTP Request header
- Mission2 : GET/POST Method Request
- Mission3: POST Method
- (Optional) Mission4: GUI

Auto Marking Program

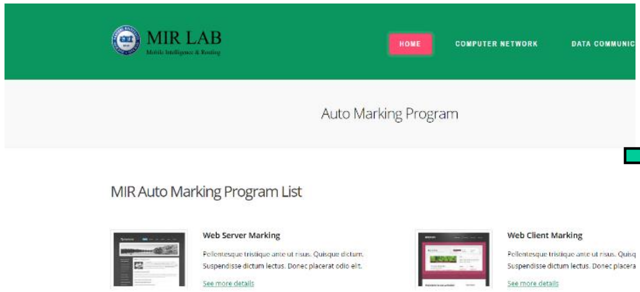
List of grading items

Mission1 : Handle USER-AGENT in HTTP Request header

Mission2 : GET/POST Method Request

Mission3: POST Method

(Optional) Mission4: GUI



Auto Marking Program

MIR Auto Marking Program List

Web Server Marking

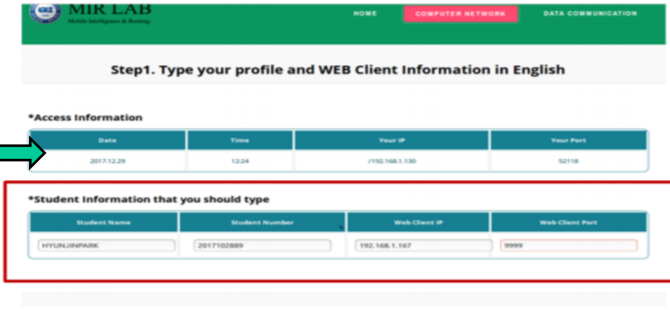
Pellentesque tristique ante ut risus. Quisque dictum. Suspendisse dictum lectus. Donec placerat odio elit.

See more details

Web Client Marking

Pellentesque tristique ante ut risus. Quisq Suspendisse dictum lectus. Donec placera

See more details



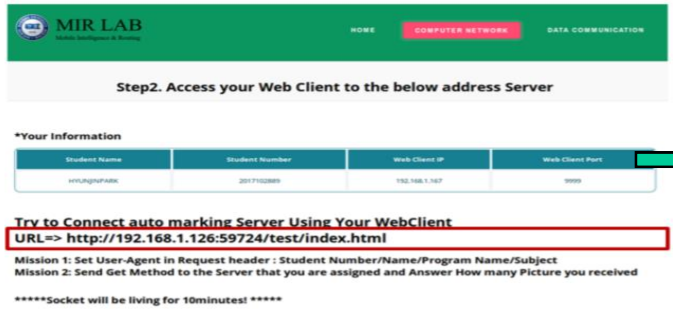
Step1. Type your profile and WEB Client Information in English

*Access Information

Date	Time	Your IP	Your Port
2017.12.29	12:24	/192.168.1.130	52118

*Student Information that you should type

Student Name	Student Number	Web Client IP	Web Client Port
HYUNJINPARK	2017102889	192.168.1.167	9999



Step2. Access your Web Client to the below address Server

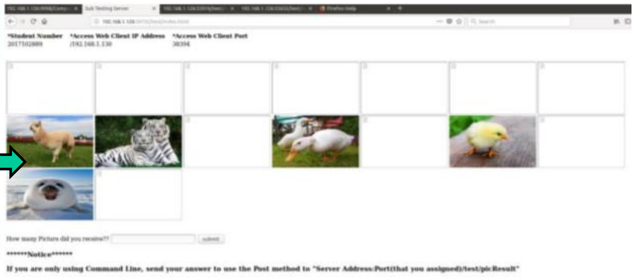
*Your Information

Student Name	Student Number	Web Client IP	Web Client Port
HYUNJINPARK	2017102889	192.168.1.167	9999

Try to Connect auto marking Server Using Your WebClient
URL=> http://192.168.1.126:59724/test/index.html

Mission 1: Set User-Agent in Request header : Student Number/Name/Program Name/Subject
Mission 2: Send Get Method to the Server that you are assigned and Answer How many Picture you received

*****Socket will be living for 10minutes! *****

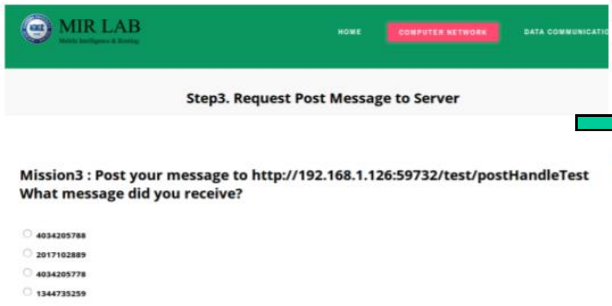


*Student Number	*Access Web Client IP Address	*Access Web Client Port
2017102889	/192.168.1.130	38394

How many Picture did you receive??

******Notice******

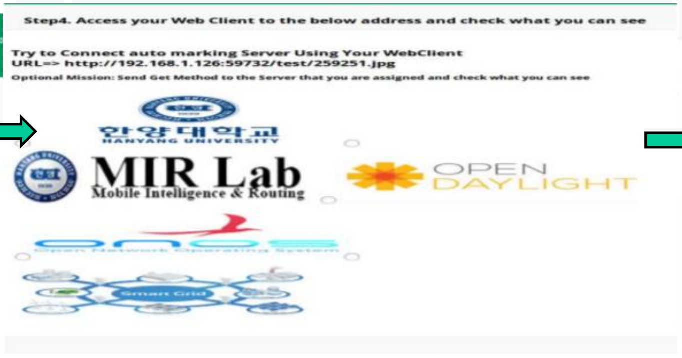
If you are only using Command Line, send your answer to use the Post method to "Server Address:Port(that you assigned)/test/picResult"



Step3. Request Post Message to Server

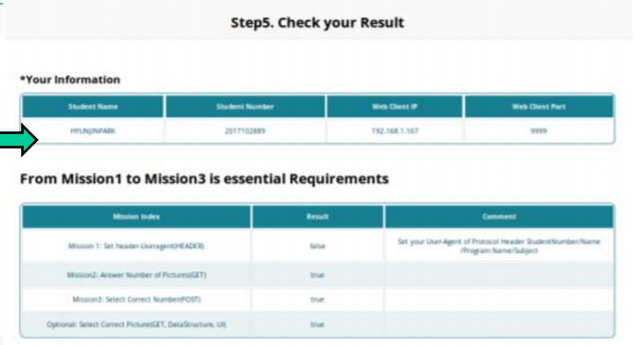
Mission3 : Post your message to http://192.168.1.126:59732/test/postHandleTest
What message did you receive?

- ○ 4034205788
- ○ 2017102889
- ○ 4034205778
- ○ 1344735259



Step4. Access your Web Client to the below address and check what you can see

Try to Connect auto marking Server Using Your WebClient
URL=> http://192.168.1.126:59732/test/259251.jpg
Optional Mission: Send Get Method to the Server that you are assigned and check what you can see



Step5. Check your Result

*Your Information

Student Name	Student Number	Web Client IP	Web Client Port
HYUNJINPARK	2017102889	192.168.1.167	9999

From Mission1 to Mission3 is essential Requirements

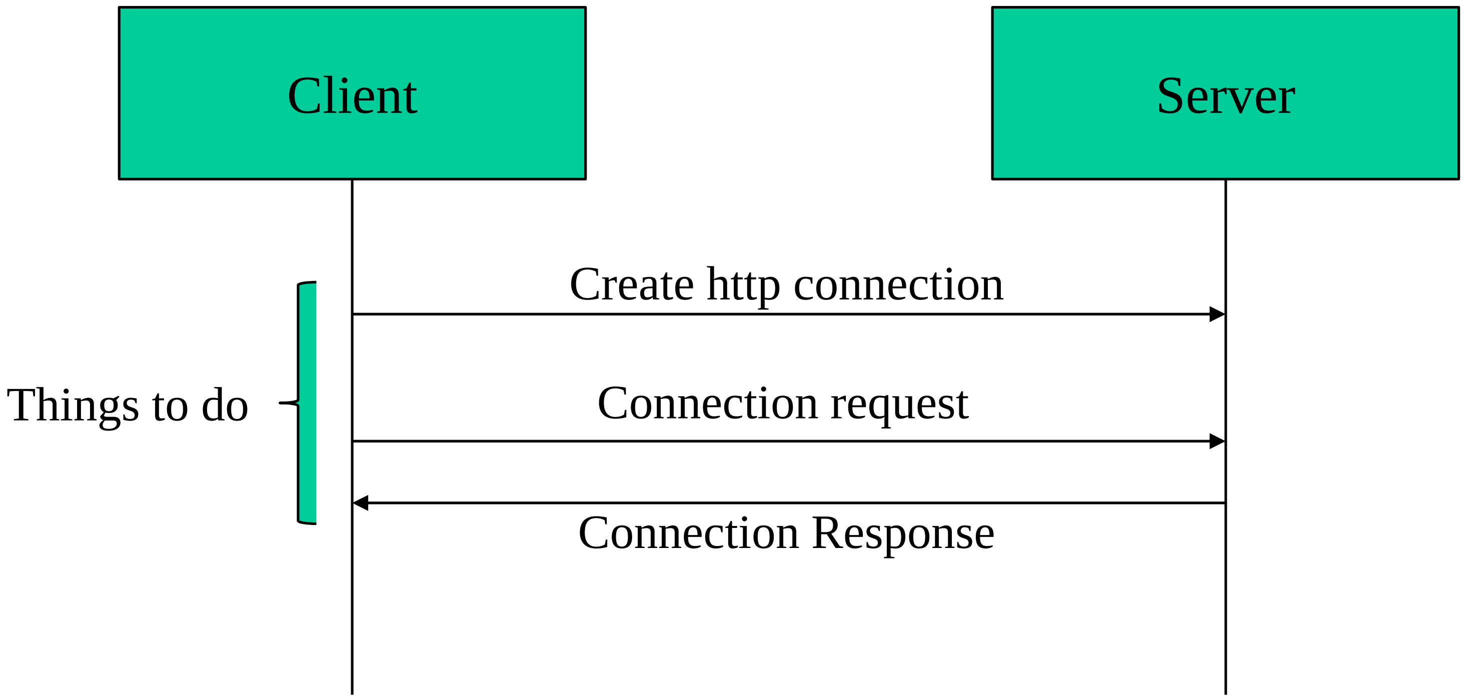
Mission Index	Result	Comment
Mission 1: Set header-Useragent(HEADER)	false	Set your User-Agent of Protocol Header StudentNumber/Name /Program Name/Subject
Mission2: Answer Number of Pictures(GET)	true	
Mission3: Select Correct Number(POST)	true	
Optional: Select Correct Picture(GET, DataStructure, UI)	true	

Overview

- Develop a simple Web-client for HTTP
- This Web-client must be connected with Web-server
- At least Get & Post Method should be ready
- If you make extra methods such as post or head, you might get extra scores

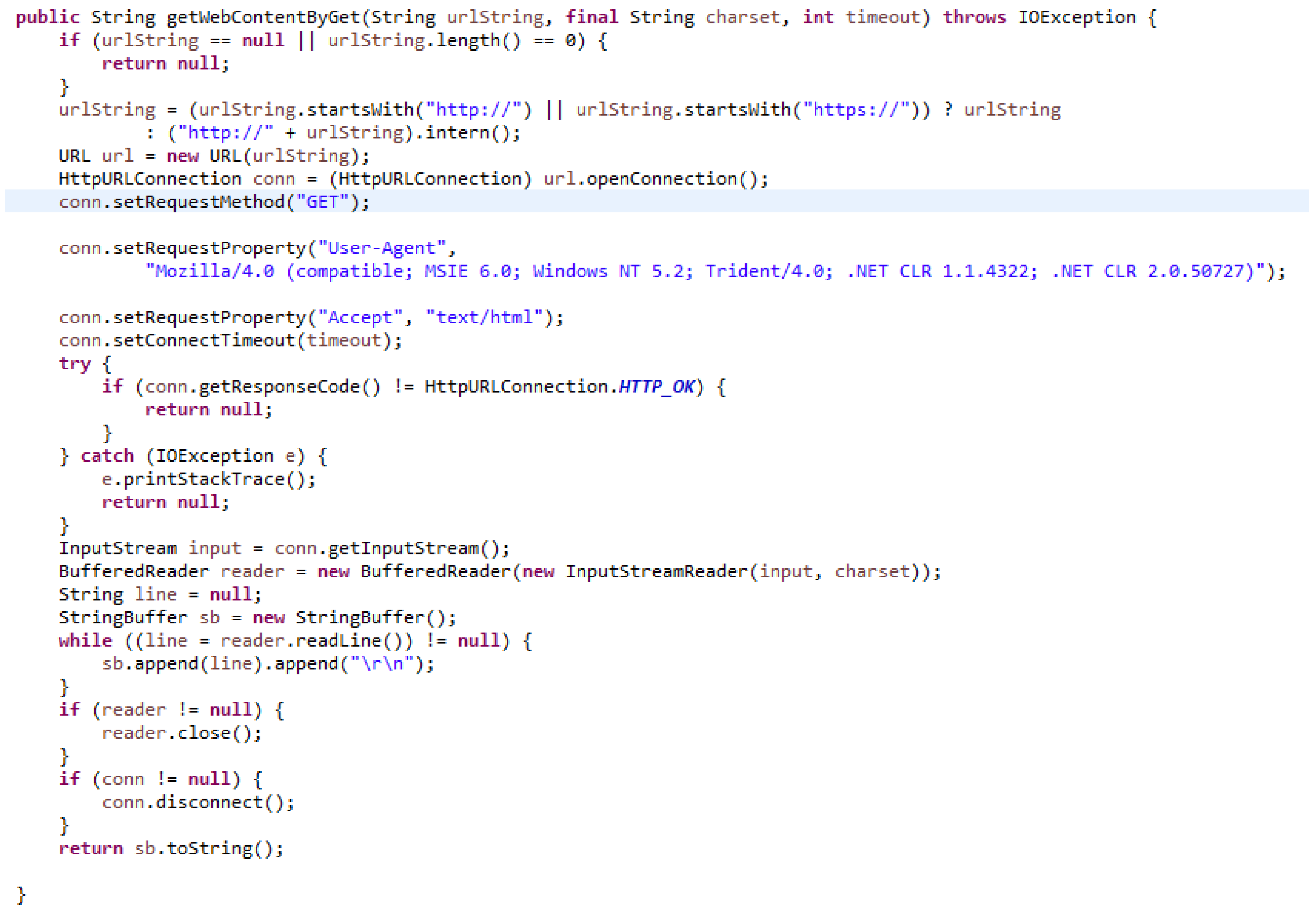
Web-Client

- Web-Client connects with Web-Server in HTTP



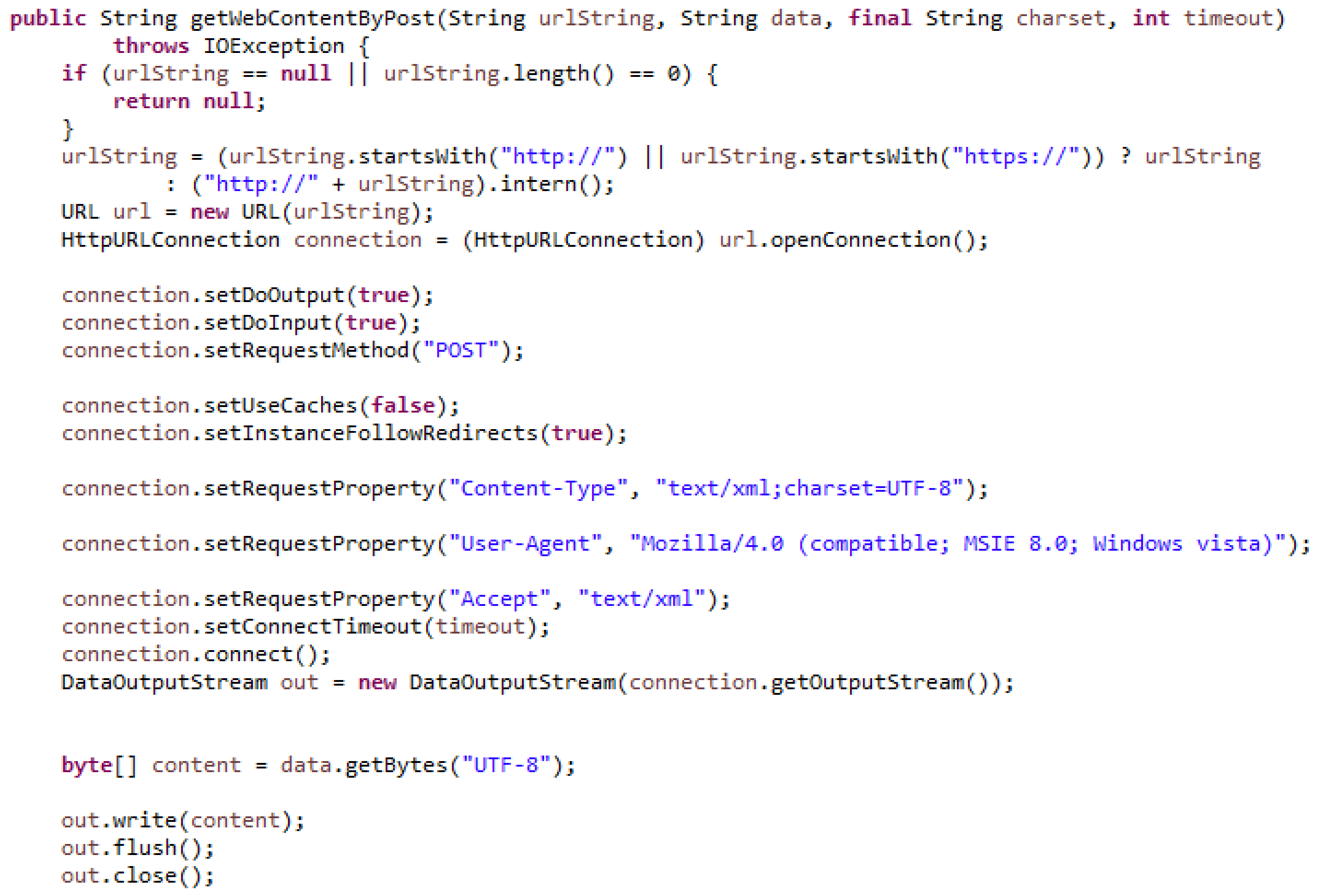
JAVA

get



```java
public String getWebContentByGet(String urlString, final String charset, int timeout) throws IOException {
    if (urlString == null || urlString.length() == 0) {
        return null;
    }
    urlString = (urlString.startsWith("http://") || urlString.startsWith("https://")) ? urlString
            : ("http://" + urlString).intern();
    URL url = new URL(urlString);
    HttpURLConnection conn = (HttpURLConnection) url.openConnection();
    conn.setRequestMethod("GET");

    conn.setRequestProperty("User-Agent",
            "Mozilla/4.0 (compatible; MSIE 6.0; Windows NT 5.2; Trident/4.0; .NET CLR 1.1.4322; .NET CLR 2.0.50727)");

    conn.setRequestProperty("Accept", "text/html");
    conn.setConnectTimeout(timeout);
    try {
        if (conn.getResponseCode() != HttpURLConnection.HTTP_OK) {
            return null;
        }
    } catch (IOException e) {
        e.printStackTrace();
        return null;
    }
    InputStream input = conn.getInputStream();
    BufferedReader reader = new BufferedReader(new InputStreamReader(input, charset));
    String line = null;
    StringBuffer sb = new StringBuffer();
    while ((line = reader.readLine()) != null) {
        sb.append(line).append("\r\n");
    }
    if (reader != null) {
        reader.close();
    }
    if (conn != null) {
        conn.disconnect();
    }
    return sb.toString();

}
```

post



```java
public String getWebContentByPost(String urlString, String data, final String charset, int timeout)
        throws IOException {
    if (urlString == null || urlString.length() == 0) {
        return null;
    }
    urlString = (urlString.startsWith("http://") || urlString.startsWith("https://")) ? urlString
            : ("http://" + urlString).intern();
    URL url = new URL(urlString);
    HttpURLConnection connection = (HttpURLConnection) url.openConnection();

    connection.setDoOutput(true);
    connection.setDoInput(true);
    connection.setRequestMethod("POST");

    connection.setUseCaches(false);
    connection.setInstanceFollowRedirects(true);

    connection.setRequestProperty("Content-Type", "text/xml;charset=UTF-8");

    connection.setRequestProperty("User-Agent", "Mozilla/4.0 (compatible; MSIE 8.0; Windows vista)");

    connection.setRequestProperty("Accept", "text/xml");
    connection.setConnectTimeout(timeout);
    connection.connect();
    DataOutputStream out = new DataOutputStream(connection.getOutputStream());


    byte[] content = data.getBytes("UTF-8");

    out.write(content);
    out.flush();
    out.close();
```

C

Web Client in C

Usually don’t use library in C language (different JAVA and Python)

It gives you an example of parsing Header
and how to send GET method which means how to build Web Client

Web Client in C

```
import  httplib
```

Httplib is needed to import to make web client
This library supports user to make a connection with web server

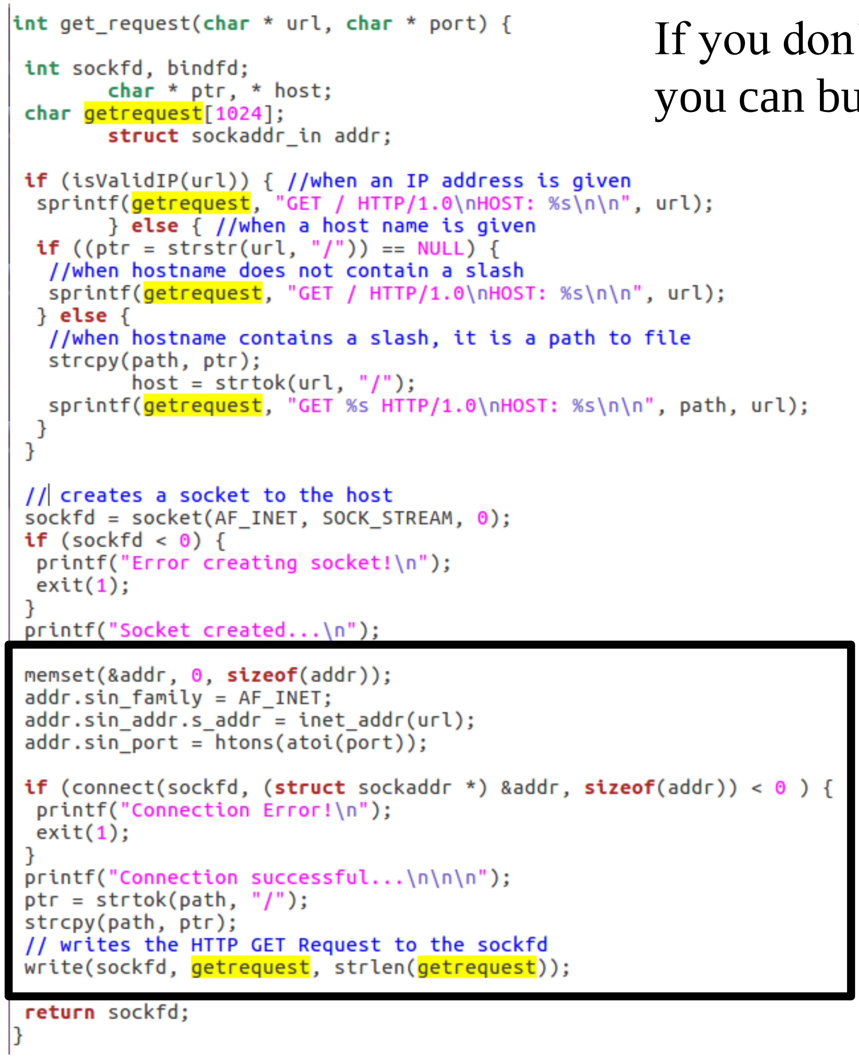
```
host = '127.0.0.1'
port = 8888
conn = httplib.HTTPConnection(host,port)
```

Making a connection with webserver.
Host means IP, port means Port number

```
conn.request("GET",  "/")
response = conn.getresponse()
print r1.status,  r1.reason

data1 = response.read()
print data1
```

With conn.request("GET") you can send get message to web server through HTTP Connection
When you send it, response would return.
With this response you can know the information of the response.
Such as the web data or response types
And also you can send HTTP messages such as "PUT" and "POST" though conn.request

Web Client in C(example)



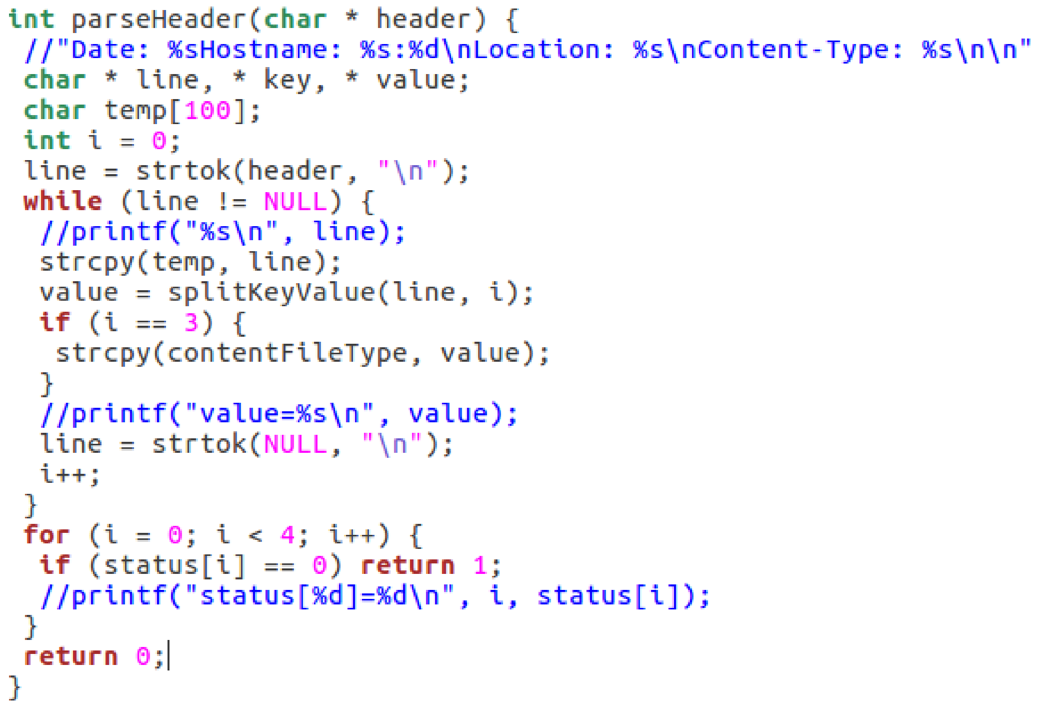
```
int get_request(char * url, char * port) {

 int sockfd, bindfd;
        char * ptr, * host;
 char getrequest[1024];
        struct sockaddr_in addr;

 if (isValidIP(url)) { //when an IP address is given
  sprintf(getrequest, "GET / HTTP/1.0\nHOST: %s\n\n", url);
        } else { //when a host name is given
  if ((ptr = strstr(url, "/")) == NULL) {
   //when hostname does not contain a slash
   sprintf(getrequest, "GET / HTTP/1.0\nHOST: %s\n\n", url);
  } else {
   //when hostname contains a slash, it is a path to file
   strcpy(path, ptr);
          host = strtok(url, "/");
   sprintf(getrequest, "GET %s HTTP/1.0\nHOST: %s\n\n", path, url);
  }
 }

 // creates a socket to the host
 sockfd = socket(AF_INET, SOCK_STREAM, 0);
 if (sockfd < 0) {
  printf("Error creating socket!\n");
  exit(1);
 }
 printf("Socket created...\n");

 memset(&addr, 0, sizeof(addr));
 addr.sin_family = AF_INET;
 addr.sin_addr.s_addr = inet_addr(url);
 addr.sin_port = htons(atoi(port));

 if (connect(sockfd, (struct sockaddr *) &addr, sizeof(addr)) < 0 ) {
  printf("Connection Error!\n");
  exit(1);
 }
 printf("Connection successful...\n\n\n");
 ptr = strtok(path, "/");
 strcpy(path, ptr);
 // writes the HTTP GET Request to the sockfd
 write(sockfd, getrequest, strlen(getrequest));

 return sockfd;
}
```

If you don't use http library
you can build Get/Post Request method as you see left side

HTTP Protocol is based on TCP(Network Layer)

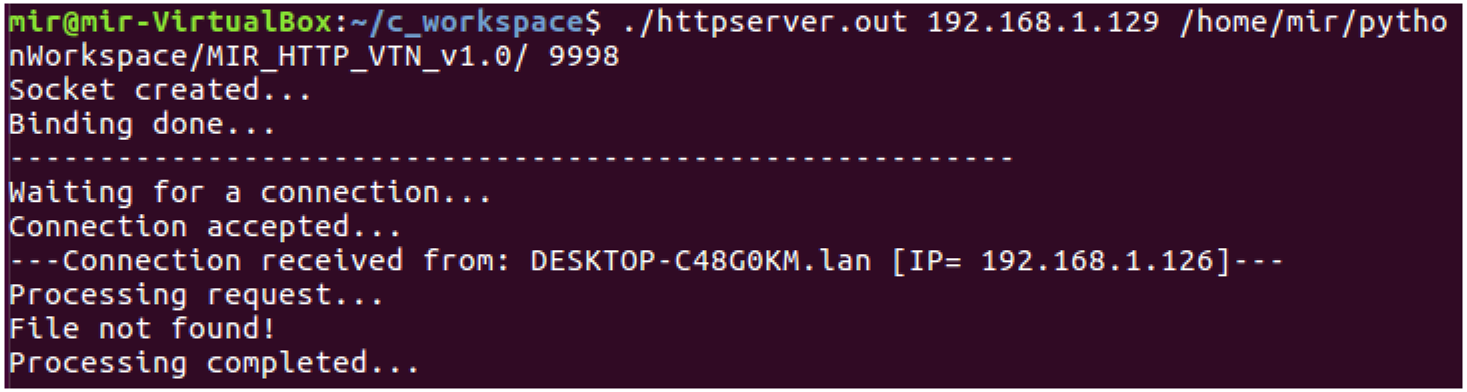
Apply TCP Client to HTTP Client

Web Client in C(Parse Header)



```c
int parseHeader(char * header) {
 //"Date: %sHostname: %s:%d\nLocation: %s\nContent-Type: %s\n\n"
 char * line, * key, * value;
 char temp[100];
 int i = 0;
 line = strtok(header, "\n");
 while (line != NULL) {
  //printf("%s\n", line);
  strcpy(temp, line);
  value = splitKeyValue(line, i);
  if (i == 3) {
   strcpy(contentFileType, value);
  }
  //printf("value=%s\n", value);
  line = strtok(NULL, "\n");
  i++;
 }
 for (i = 0; i < 4; i++) {
  if (status[i] == 0) return 1;
  //printf("status[%d]=%d\n", i, status[i]);
 }
 return 0;
}
```

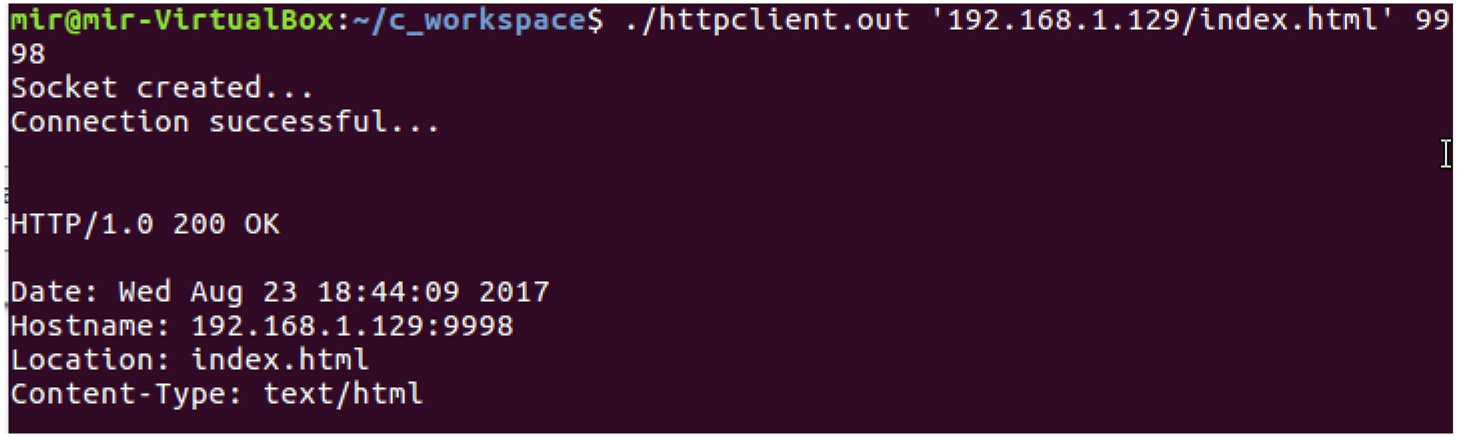
Display



```
mir@mir-VirtualBox:~/c_workspace$ ./httpserver.out 192.168.1.129 /home/mir/pytho
nWorkspace/MIR_HTTP_VTN_v1.0/ 9998
Socket created...
Binding done...
-------------------------------------------------------
Waiting for a connection...
Connection accepted...
---Connection received from: DESKTOP-C48G0KM.lan [IP= 192.168.1.126]---
Processing request...
File not found!
Processing completed...
```

HTTP Server

Handling of Get & Put message is required



```
mir@mir-VirtualBox:~/c_workspace$ ./httpclient.out '192.168.1.129/index.html' 99
98
Socket created...
Connection successful...


HTTP/1.0 200 OK

Date: Wed Aug 23 18:44:09 2017
Hostname: 192.168.1.129:9998
Location: index.html
Content-Type: text/html
```

HTTP Client

python

Web Client in Python

```
import  httplib
```

Httplib is needed to import to make web client
This library supports user to make a connection with web server

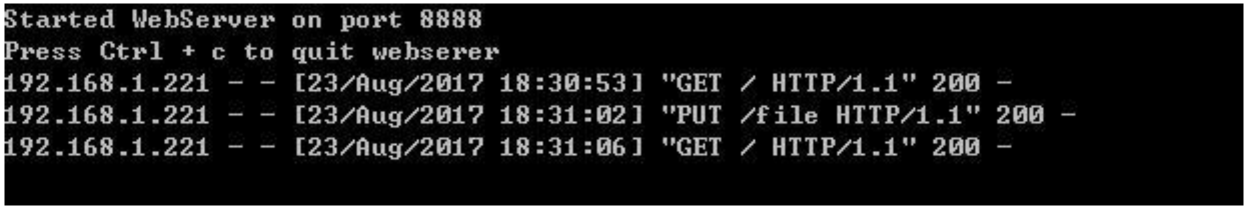
```
host = '127.0.0.1'
port = 8888
conn = httplib.HTTPConnection(host,port)
```

Making a connection with webserver.
Host means IP, port means Port number

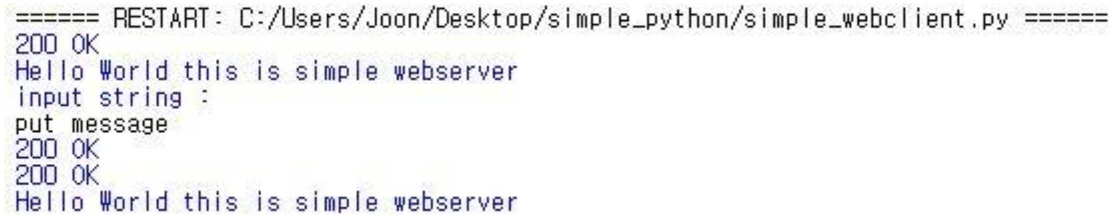
```
conn.request("GET",  "/")
response = conn.getresponse()
print r1.status,  r1.reason

data1 = response.read()
print data1
```

With conn.request("GET") you can send get message to web server through HTTP Connection
When you send it, response would return.
With this response you can know the information of the response.
Such as the web data or response types
And also you can send HTTP messages such as "PUT" and "POST" though conn.request

Display



```
Started WebServer on port 8888
Press Ctrl + c to quit webserer
192.168.1.221 - - [23/Aug/2017 18:30:53] "GET / HTTP/1.1" 200 -
192.168.1.221 - - [23/Aug/2017 18:31:02] "PUT /file HTTP/1.1" 200 -
192.168.1.221 - - [23/Aug/2017 18:31:06] "GET / HTTP/1.1" 200 -
```



```
====== RESTART: C:/Users/Joon/Desktop/simple_python/simple_webclient.py ======
200 OK
Hello World this is simple webserver
input string :
put message
200 OK
200 OK
Hello World this is simple webserver
```

Handling of Get & Put message is required