ITCS 6166/8166 Computer Communication Networks

Project – 1 Report

HTTP Client and Server

Team Members:

1. Arunkumar Bagavathi (800888454)
2. Chetan Borse (800936059)

Programming Language: Python Version 3

Objective:

The objective of this project is to implement HTTP server and HTTP client that run-on simple HTTP/1.1. In this project, our goal is to implement GET and PUT requests from client to the server and the server handles the client requests.

HTTP GET and PUT requests:

GET and PUT requests are most commonly used methods between server and client in HTTP/1.1. In GET request, client requests the server to return a file (more like downloading a file from the server). In PUT request, client asks the server to store a file in server's location (more like uploading a file to the server).

We used socket programming and TCP connection to handle GET and PUT requests and returning server response.

HTTP Server:

In HTTP, the server runs at a particular port in a particular IP address (localhost or 127.0.0.1 in case of single machine running both server and client). We implemented a multithreaded server, which can serve requests from multiple clients at a same time. Thus, the server is ready to serve the client request all the time. Server can do the following:

- Setup connection with the client
- Receive GET request – Returns a message: "200 OK" followed by the requested file if the file exists or returns a message: "404 Not Found" if the requested file does not exist
- Receive PUT request - Accept and save the file in the server directory and responds to the client with "200 OK File Created" message
- After finishing the client request, closes the TCP connection and starts listening for other client requests

HTTP Client:

In HTTP, client can request a file from the server or request to save client's file in the server after successfully connecting with the server. Only after client sending some request to the server, any GET or PUT task can be achieved. Client can do the following:

- Connecting with the server which is currently running at a given IP address and port number
- Giving a GET request to the server – requesting a file from the server
- Giving a PUT request to the server – requesting the server to save a file in server directory after successfully reading the given file
- Closing the connecting with server once the server responds to client with a success message

Instructions to run the python code:

1. Unzip the zip file that is attached with the submission
2. Go to the unzipped folder using terminal or load the folder in Anaconda software
3. Start the server using the command:

   python ServerApp.py
4. Optionally, you can specify server IP address, server port number and server directory for storing files using -t, -p and -w parameters respectively like

   python ServerApp.py -t <IP_ADDRESS> -p <PORT_NUMBER> -w <SERVER_FILE_DIRECTORY>

   These parameters were made optional. So, the server program will take default values assigned to it even if you do not give these values. Default server IP_ADDRESS of is set to **127.0.0.1** and default server PORT_NUMBER is set to **8080**We created a default server and client directory for saving files in server and client respectively. The directories are inside the *data* folder. Each client and server directory contains one file. "index.html" in the server folder and "client_index.html"

5. Start the client with the command:

   Python ClientApp.py -s <CLIENT_IP> -c <CLIENT_PORT> -t <SERVER_IP> -p <SERVER_PORT> -m <REQUEST_METHOD> -f <FILE_NAME> -d <CLIENT_FILE_DIRECTORY>

   All parameters for running the client contains default values. Default values are:

   - CLIENT_IP – 127.0.0.1
   - CLIENT_PORT – 8081
   - SERVER_IP – 127.0.0.1
   - SERVER_PORT – 8080
   - REQUEST_METHOD – GET
   - FILE_NAME – index.html
   - CLIENT_FILE_DIRECTORY – points to project_directory/data/client
6. Thus, the command **python ClientApp.py** sends the server, running in IP address 127.0.0.1 at port number 8080, a GET request to return a file "index.html"

7. To make a proper GET request, use the command:
   python ClientApp.py -t <SERVER_IP> -p <SERVER_PORT> -m GET -f <FILE_NAME>
8. To make a proper PUT request, use the command:
   python ClientApp.py -t <SERVER_IP> -p <SERVER_PORT> -m PUT -f <FILE_NAME> -d <FILE_DIRECTORY>

Results:

1. **Running GET command and the file exists in the server**

```
2017-02-09 22:48:34,811 HTTPServer [INFO] Established client connection with 127.0.0.1:55857
2017-02-09 22:48:34,811 HTTPServer [DEBUG] New HTTP server socket thread started for the client 127.0.0.1:55857
2017-02-09 22:48:34,813 HTTPServer [INFO] [8080->127.0.0.1:55857] Receiving client request
2017-02-09 22:48:34,813 HTTPServer [DEBUG] [8080->127.0.0.1:55857] Request Method: GET
2017-02-09 22:48:34,814 HTTPServer [DEBUG] [8080->127.0.0.1:55857] Requested File: index.html
2017-02-09 22:48:34,814 HTTPServer [DEBUG] [8080->127.0.0.1:55857] Requested Arguments: None
2017-02-09 22:48:34,814 HTTPServer [INFO] [8080->127.0.0.1:55857] Serving web page: E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1\data\server\index.html
2017-02-09 22:48:34,814 HTTPServer [DEBUG] [8080->127.0.0.1:55857] Sending HTTP response!
2017-02-09 22:48:34,815 HTTPServer [DEBUG] [8080->127.0.0.1:55857] Creating HTTP server response
2017-02-09 22:48:34,816 HTTPServer [DEBUG] [8080->127.0.0.1:55857] 1024 bytes were sent.....
2017-02-09 22:48:34,816 HTTPServer [DEBUG] [8080->127.0.0.1:55857] 1024 bytes were sent.....
2017-02-09 22:48:36,817 HTTPServer [INFO] [8080->127.0.0.1:55857] Closing client connection
```

Figure 1: Server responding to client's GET request

```
E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1>python ClientApp.py
2017-02-09 22:48:34,809 HTTPClient [INFO] Creating client socket.....
2017-02-09 22:48:34,810 HTTPClient [INFO] Connecting with HTTP server 127.0.0.1:8080
2017-02-09 22:48:34,811 HTTPClient [INFO] [GET] Sending http request to HTTP server.....
2017-02-09 22:48:34,811 HTTPClient [DEBUG] [GET] HTTP Request: GET /index.html HTTP/1.1
Host: 127.0.0.1

2017-02-09 22:48:34,815 HTTPClient [INFO] [GET] HTTP Response: HTTP/1.1 200 OK
Date: 'Thu, 09 Feb 2017 22:48:34'
Server: HTTPServer [127.0.0.1:8080]
Connection: keep-alive


2017-02-09 22:48:34,816 HTTPClient [DEBUG] [GET] 1024 bytes received.....
2017-02-09 22:48:34,816 HTTPClient [DEBUG] [GET] 1024 bytes received.....
2017-02-09 22:48:36,817 HTTPClient [INFO] Closing client socket.....
```

Figure 2: Client giving GET request and receiving response from the server

Client receives “200 OK” message and the requested file “index.html” is now present in the default client directory

2. Running a GET command and the requested file does not exist in the server

```
2017-02-10 19:59:04,570 HTTPServer [INFO] Launching HTTP server on 127.0.0.1:8080
2017-02-10 19:59:04,570 HTTPServer [INFO] HTTP server is successfully launched at 127.0.0.1:8080
2017-02-10 19:59:04,570 HTTPServer [INFO] Press Ctrl+C to shut down the running HTTP server and exit.
2017-02-10 19:59:04,570 HTTPServer [INFO] HTTP server is listening at port 8080
2017-02-10 19:59:07,355 HTTPServer [INFO] Established client connection with 127.0.0.1:58814
2017-02-10 19:59:07,355 HTTPServer [DEBUG] New HTTP server socket thread started for the client 127.0.0.1:58814
2017-02-10 19:59:07,355 HTTPServer [INFO] [8080->127.0.0.1:58814] Receiving client request
2017-02-10 19:59:07,355 HTTPServer [DEBUG] [8080->127.0.0.1:58814] Request Method: GET
2017-02-10 19:59:07,355 HTTPServer [DEBUG] [8080->127.0.0.1:58814] Requested File: client_inde.html
2017-02-10 19:59:07,355 HTTPServer [DEBUG] [8080->127.0.0.1:58814] Requested Arguments: None
2017-02-10 19:59:07,355 HTTPServer [INFO] [8080->127.0.0.1:58814] Serving web page: E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1\data\server\client_inde.html
2017-02-10 19:59:07,355 HTTPServer [WARNING] [8080->127.0.0.1:58814] E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1\data\server\client_inde.html: File not found!
2017-02-10 19:59:07,355 HTTPServer [DEBUG] [8080->127.0.0.1:58814] Sending HTTP response!
2017-02-10 19:59:07,355 HTTPServer [DEBUG] [8080->127.0.0.1:58814] Creating HTTP server response
2017-02-10 19:59:07,355 HTTPServer [INFO] [8080->127.0.0.1:58814] Closing client connection
```

Figure 3: Server responding to the client request that requested file does not exist

```
E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1>python ClientApp.py -f client_inde.html
2017-02-10 19:59:07,355 HTTPClient [INFO] Creating client socket.....
2017-02-10 19:59:07,355 HTTPClient [INFO] Connecting with HTTP server 127.0.0.1:8080
2017-02-10 19:59:07,355 HTTPClient [INFO] [GET] Sending http request to HTTP server.....
2017-02-10 19:59:07,355 HTTPClient [DEBUG] [GET] HTTP Request: GET /client_inde.html HTTP/1.1
Host: 127.0.0.1

2017-02-10 19:59:07,355 HTTPClient [INFO] [GET] HTTP Response: HTTP/1.1 404 Not Found
Date: 'Fri, 10 Feb 2017 19:59:07'
Server: HTTPServer [127.0.0.1:8080]
Connection: keep-alive

<html><body><p>Status Code 404: File Not Found!</p></body></html>
2017-02-10 19:59:07,355 HTTPClient [INFO] Closing client socket.....
```

Figure 4: Client requesting the unavailable file and getting "404" error message

3. Running a PUT command

```
E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1>python CLientApp.py -f client_index.html -m PUT
2017-02-09 23:10:39,490 HTTPClient [INFO] Creating client socket.....
2017-02-09 23:10:39,491 HTTPClient [INFO] Connecting with HTTP server 127.0.0.1:8080
2017-02-09 23:10:39,492 HTTPClient [INFO] [PUT] Sending http request to HTTP server.....
2017-02-09 23:10:39,493 HTTPClient [DEBUG] [PUT] HTTP Request: PUT /client_index.html HTTP/1.1
Host: 127.0.0.1


2017-02-09 23:10:41,493 HTTPClient [DEBUG] [PUT] 1024 bytes were sent.....
2017-02-09 23:10:41,494 HTTPClient [DEBUG] [PUT] 1024 bytes were sent.....
2017-02-09 23:10:45,494 HTTPClient [INFO] [PUT] HTTP Response: HTTP/1.1 200 OK File Created
Date: 'Thu, 09 Feb 2017 23:10:43'
Server: HTTPServer [127.0.0.1:8080]
Connection: keep-alive


2017-02-09 23:10:45,495 HTTPClient [INFO] Closing client socket.....
```

Figure 5: Client sending a PUT request

```
2017-02-09 23:10:39,492 HTTPServer [INFO] Established client connection with 127.0.0.1:55930
2017-02-09 23:10:39,493 HTTPServer [DEBUG] New HTTP server socket thread started for the client 127.0.0.1:55930
2017-02-09 23:10:39,494 HTTPServer [INFO] [8080->127.0.0.1:55930] Receiving client request
2017-02-09 23:10:39,494 HTTPServer [DEBUG] [8080->127.0.0.1:55930] Request Method: PUT
2017-02-09 23:10:39,495 HTTPServer [DEBUG] [8080->127.0.0.1:55930] Requested File: client_index.html
2017-02-09 23:10:39,495 HTTPServer [DEBUG] [8080->127.0.0.1:55930] Requested Arguments: None
2017-02-09 23:10:39,495 HTTPServer [INFO] [8080->127.0.0.1:55930] Writing to web server: E:\PyCharm Community Edition 2016.3.2\HTTP-v1.1\data\server\client_index.html
2017-02-09 23:10:41,494 HTTPServer [DEBUG] [8080->127.0.0.1:55930] 1024 bytes were written.....
2017-02-09 23:10:41,494 HTTPServer [DEBUG] [8080->127.0.0.1:55930] 1024 bytes were written.....
2017-02-09 23:10:43,495 HTTPServer [DEBUG] [8080->127.0.0.1:55930] Sending HTTP response!
2017-02-09 23:10:43,495 HTTPServer [DEBUG] [8080->127.0.0.1:55930] Creating HTTP server response
2017-02-09 23:10:43,496 HTTPServer [INFO] [8080->127.0.0.1:55930] Closing client connection
```

Figure 6: Server processing the PUT request

Now, the client_index.html is present in server directory

Exception in the code:

We can able to terminate the server using **Ctrl + C** only in the Unix or Mac machine. But the same command is not working in the Windows command prompt