Worcester Polytechnic Institute
Department of Computer Science
CS3516– Computer Networks

Programming Assignment 1B – Web Server

Points: 80% of the entire Programming Assignment 1.

Goal: Socket programming assignments are to help you review and apply your conceptual knowledge from this class.

Instructions:
In this assignment, you will develop a simple Web server in Python that is capable of processing only one request. Specifically, your Web server will (i) create a connection socket when contacted by a client (browser); (ii) receive the HTTP request from this connection; (iii) parse the request to determine the specific file being requested; (iv) get the requested file from the server's file system; (v) create an HTTP response message consisting of the requested file preceded by header lines; and (vi) send the response over the TCP connection to the requesting browser. If a browser requests a file that is not present in your server, your server should return a "404 Not Found" error message.

The skeleton code is provided later in this assignment. Your job is to complete the code, run your server, and then test your server by sending requests from browsers running on different hosts.

Detailed Instructions on Running the Server:
In this assignment, you'll need a server machine (on which you run the server program you wrote), and a client machine (on which you run the web browser such as Chrome). The server machine and client machine can be the same physical machine, or different machines. You need to prepare two files: the server program, and a HTML file (e.g. HelloWorld.html, which is already provided to you).

You need to:
1) put the HTML file and the server program in the same directory on the server machine;
2) run the server program on the server machine;
3) on the client machine, open the web browser and type in the correct URL to access the HTML file. In this case, the HTML file will be displayed correctly in the web browser on the client machine.
4) Then try to request a file that is not present at the server. You should get a "404 Not Found" message in this case.

To find the correct URL in step 3), you should determine the correct IP address of the server machine (e.g., 128.238.251.26). The IP address could be the IP of local host (127.0.0.1) if you are running both server program and client (web browser) on the same machine. If you are running the web server on a campus machine (such as linux.wpi.edu), you need to check the correct IP address of that machine. In Linux, you can use the command ***ifconfig*** to find the IP.

For example:
The URL is written in the following way if the server machine's IP is 128.238.251.26, and the port number of the web service you specified in the server code is 6789:
http://128.238.251.26:6789/HelloWorld.html
or
http://127.0.0.1:6789/HelloWorld.html

'HelloWorld.html' is the name of the file you placed in the server directory. Note also the use of the port number after the colon. You need to replace this port number with whatever port you have used in the server program. In the above example, we have used the port number 6789. The browser should then display the contents of HelloWorld.html. If you omit ":6789", the browser will assume port 80 and you will get the web page from the server only if your server is listening at port 80 (This is not recommended. ***Please avoid using 80 or 8080 as the port number*** because they are usually reserved). It is recommended that you choose a unique port number.

Deliverables:
1. The server code (and the HTML file if you modified it);
2. A README (txt) file that spells out exactly how to run the server code.

Code plagiarism is absolutely **NOT** allowed. If needed, you may be asked for a **demonstration** of running your program in front of the instructor/SAs and answer their questions about your code. In this case, your grade will be based on both the report and your performance during demonstration.

Grading:
Grade breakup (%) assuming all documents are present:
1. Web server = 95 points
2. Readme file = 5 points

The grade will be a **ZERO** if the code does not run or gives a run-time error.

To get the FULL CREDIT for the Web server:
1. The web browser should be able to obtain the html from the server and displaying it.
2. When the browser requests a file that is not present at the server, it should get a "404 Not Found" message and display it on the browser.

Notes:
1. If you cannot install Python on your own machine, you can connect to ***linux.wpi.edu*** **server** on campus network to do the assignment. If you are off campus and ***you try to run the programs on the server***, you need to use VPN to connect back to campus network and use ssh to connect to the server. You don't need VPN or ssh if you run all your programs on your local machine.
   Please see more instruction here:
   https://hub.wpi.edu/article/786/connect-to-linux-cluster-using-a-terminal-program
2. When you connect to linux.wpi.edu, you may get hostnames other than`linux.wpi.edu` (as given the examples above). Please use the command `hostname` to find out where you are connected.
3. You must choose a server port number larger that 1023 (to be safe, choose a server port number larger than 5000).

4. If you need to kill a background process after you have started it, you can use the UNIX *kill* command. Use the UNIX *ps* command to find the process id of your server.
5. Make sure you close unused sockets that you use in your program.
6. If you abort your program, the socket may still hang around and the next time you try and bind a new socket to the port ID you previously used (but never closed), you may get an error.
7. On UNIX systems, you can run the command "netstat" to see which port numbers are currently assigned.