CSUS, College of Engineering and Computer Science
Department of Computer Science
CSC 138 – Computer Network and Internet

Socket Programming Assignment 2 – Web Server

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

Attention: Although the examples discussed in class are in Python/C, your submission can be in Python/C/Java. If you choose do so, the caveat is that there is more help (see below) if you do it in Python. Code plagiarism is absolutely **NOT** allowed! Please prepare for a **demonstration** of running your program in front of the instructor/grader and answer their questions.

Instructions: (Textbook Page 179) This is the first of a series of programming assignments in the text book that will be assigned in the course of the semester. Students can find full details of these assignments, as well as important snippets of the Python code, at the Web site for the text book.
http://wps.pearsoned.com/ecs_kurose_compnetw_6/.

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.

In the **textbook**'s **companion Web site**, we provide the skeleton code for your server. Your job is to complete the code, run your server, and then test your server by sending requests from browsers running on different hosts. If you run your server on a host that already has a Web server running on it, then you should use a different port than port 80 for your Web server.

Detailed Instructions on Running the Server:
You may create a HTML file (e.g. HelloWorld.html) and put it in the same directory that the server is in. Run the server program. You should determine the IP address of the host that is running the server (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 and client on the same machine. It could also be the IP address of athena.ecs.csus.edu if you are running the web server on Athena. From another host (the client side), open a browser and provide the corresponding URL.

For example:
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 code. 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 if you are using Athena to do the lab). It is recommended that you choose a unique port number. Then try to get a file that is not present at the server. You should get a “404 Not Found” message.

Deliverable: An **electronic submission** of lab report should be submitted to my Canvas before the deadline. You should include both your **source code** (could be a source file or a screenshot) and at least **three screenshots** that can help you demonstrate your work: *one screenshot for the execution of source code in terminal, one screenshot for displaying the file in the web browser, and one screenshot for displaying “404 Not Found” message in web browser*.

Requirement: Code plagiarism is absolutely **NOT** allowed! Please also prepare for a **demonstration** of running your program in front of the instructor/grader and answer their **questions** (which are about your code). You grade will be based on both the report and your performance during demonstration.

The report will all be evaluated based on the following grading criteria.

Report Correctness, Completeness, Clarity	20%+15%+15%
Demonstration Correctness, Completeness, Question	20%+15%+15%