Instructions

Due: 9/15/14 11:59PM

Complete the following assignment in pairs (with a partner). Submit your work into the Dropbox on D2L into the "Lab 1" folder. Both partners will submit the same solution.

Assignment

In this lab, you will learn the basics of socket programming for TCP connections in Python: how to create a socket, bind it to a specific address and port, as well as send and receive a HTTP packet. You will also learn some basics of HTTP header format.

You will develop a web server that handles one HTTP request at a time. Your web server should accept and parse the HTTP request, get the requested file from the servers file system, create an HTTP response message consisting of the requested file preceded by header lines, and then send the response directly to the client. If the requested file is not present in the server, the server should send an `HTTP 404 Not Found` message back to the client.

1. [5 points] Find a partner.

   Submit `partner.txt` with your partner's first and last name.

2. [10 points] Download the skeleton code `web_server.py` and fill in the missing pieces between `###Fill in start` and `###Fill in end`. Each place may require one or more lines of code.

   Submit your completed `web_server.py`.

3. [5 points] Put an HTML file (e.g., `HelloWorld.html`) in the same directory that the server is in. Run the server program. Determine the IP address of the host that is running the server (e.g., 128.238.251.26). From another host, open a browser and provide the corresponding URL. For example:

   `http://128.238.251.26:6789/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.

   Then try to get a file that is not present at the server. You should get a `404 Not Found` message.

   Submit two screenshots of your server displaying a `HelloWorld.html` and of the `404 Not Found` error.

4. [10 points] Instead of using a browser, write your own HTTP client to test your server. Your client will connect to the server using a TCP connection, send an HTTP request to the server, and display the server response as an output. You can assume that the HTTP request sent is a GET method. The client should take command line arguments specifying the server IP address or host name, the port at which the server is listening, and the path at which the requested object is stored at the server.

   The following is an input command format to run the client.

   ```
   python client.py server_host server_port filename
   ```

   Submit a working `client.py` program.

5. [2 points] BONUS QUESTION: I have created a service on `nl.cs.montana.edu` running on a random port in the range $[5000, 6000]$. What is the reply from the service, when you send it your student id?

   Submit `bonus.txt` containing the reply from the service.