Julia Chancey

6592-4559

julia.chancey@ufl.edu

CNT 4731

Programming Assignment #2

How to compile and run code under Linux environment:

1. javac server.java

2. javac client.java
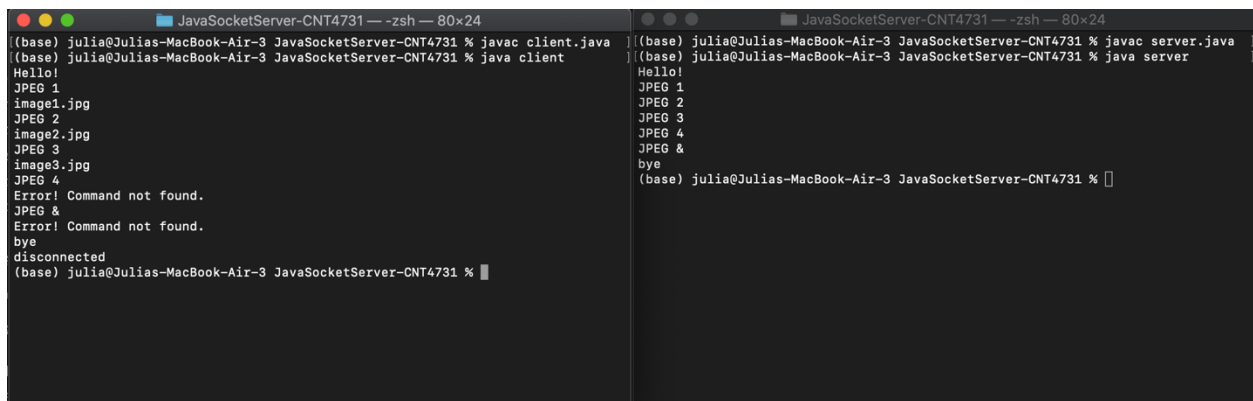
3. java server

4. java client

Server code structure:

In the server file, I create a new Socket Server on port 4559 (last 4 digits of my ufid) and a socket that establishes the client connections. As soon as the connection is established, I use an output stream to send and print "Hello!" to indicate the start of the program. I have a while loop where it accepts input streams from the client connection. With each input stream, I check what command has been passed in from the client, using a switch statement. If the command is JPEG 1, JPEG 2, or JPEG 3, I return the jpeg image file corresponding to the number provided. If the command is "bye", I write "disconnected" back, break out of the while loop, and close all resources. If the input stream does not equal any of the commands, I return "Error! Command not found." With each while loop iteration, I close the socket, input stream, and output stream at the end and create new instances of them at the beginning to ensure I'm able to get the new commands from the client each time. Once the exit command is passed in, I close the server and the program is terminated.

Client code structure:

In the client file, I create the resources needed for the server connection, such as the socket (using the localhost IP address and the port number), input stream, output stream, and scanner. When the program begins, my client reads the server's intro message using an input stream and prints it out. My program then enters a while loop that continues if the exit Boolean is not set to true. Inside the loop, I initialize the socket and output stream objects in addition, I use the scanner to retrieve user input and place the read object into a var variable, given that I don't

know the data type – it could be a file or a string. Then I place the type cast of the var variable to a File inside a try statement to see whether I have received a file or not and prints out the file name (I wasn't sure what exactly to do with the image file, the file is received and stores in the client program, it just wasn't clear to me what to do with it, so I printed out the name, please don't deduct points because I really didn't understand what to do with the file). If the type cast is unsuccessful, the program falls into the catch exception statement which type casts the var object into a string and prints it out. At the end of the loop, close the input and output streams, set the thread to sleep for 100 milliseconds and set the exit variable to true if user input was equal to "bye". Once the loop is broken out of, I close the socket and the scanner and terminate the program.

Execution results:

Above is the client and server running, having entered all the possible commands and some error commands too.

Lesson learnt:

I learned so much about how to build, send, and get objects from the client to the server using connections that I had never worked with before. I had always heard of using a client or using a server and having them connect etc but never actually seen how it's done or developed that connection. This assignment makes me wonder how websites who have client and server applications connect them and whether they use java socket servers or not. I would love to play around with the assignment and see if I could potentially send get and fetch requests to/from a database and get a better understanding for how the backend of websites work.

Additional comments:

I read the assignment file like 5 times and I don't know if I missed the instruction multiple times or not but I simply had no idea what to do with the image file once the client had received it, therefore I just printed the name of the file out but I know how to do anything with the file. I could store it or print out its absolute path etc.