Java Lab22: Sockets

This lab practices with sockets, clients, and servers. Test each problem by running the server code first, then the client code.

1. Create a new Java project called Lab22. Download the two .java files into the src directory.There are two main programs, one each in the two classes Client and Server. Run Server first, then run Client. Their output will be in two separate console windows, so either line them up side-by-side or just switch between them. Some simple messages are hard-coded in each part, so you should see them printed in the console.

2. In the Client, change the hard coded message: prompt the user (that's you) to enter a String message, get it with a keyboard scanner; the code to send it to the server remains the same. Do the same on the Server side: after a message is received, prompt the user (that's still you) to enter a reply; the code to return it to the client stays the same.

3. Now put both parts in loops. On the client side, if the user enters "QUIT", send that message, and then exit the loop. On the server side, when "QUIT" is received, print that out and exit the loop.

4. In the server code, add: ***`int port = 8001;`*** and change the parameter in the ServerSocket constructor to ***`port`*** instead of hard-coding 8001; test that. Then add this code right before the ServerSocket constructor call:

```
if (args.length == 1) {
   port = Integer.parseInt(args[0])
}
```

Next, go to Run->Edit Configurations. (If nothing shows up, run the server, then try Run->Configurations; same with the client.) Make sure Server is chosen on the left. In the Configuration tab on the right, type 8001 in the Program Arguments text box. Test it.

5. Now try the same thing with the client code – create two variables, ***`String address`*** and ***int port***, set to the default values "localhost" and 8001; use them as the two parameters in the Socket constructor. Then add similar code as in #4 right before the Socket constructor, but the if test should be args.length == 2, and you'll set address to args[0] and port to args[1], converting it to an int. Go to Run->Edit Configurations; make sure Client is chosen, and type "localhost" and 8001 in the Program Arguments text box. Test it.

6. Create this method in the server:
`public static void handleClient(Socket clientConnection) throws IOException`.
Copy and paste the code from the line after accept() to the end of the loop into this method; just in case, comment out the code you've copied in main instead of deleting it. Put a call to handleClient(clientConnection) in its place. Make sure it runs correctly.

The deliverable is the code up to this point. Zip the two files and upload them to Canvas.

7. Find another person to pair up with. One person will run their server code, and the other person will run their client code. The person running the Server code should look up their IP address. On a Mac, click the Apple icon (upper left hand corner), choose System Preferences, and click the Network icon. The IP address should be listed. On Windows, click the Windows icon (lower left hand corner), click Settings, click Network&Internet, and click View Network Properties.The IPv4 address should be listed (don't use the /# at the end, just the four-part number). Tell your partner that address – they'll need to copy that number into the address field in #5. Does it work?

8. Now switch roles and try #7 again.