Programming project 4

Shahd Mustafa

Implementation and design:

For this program I used the chapter 33 of the book to understand how to connect the client and server sockets and get them to communicate with one another. First, I made the interface that the client and server will store all the information and communicate through using fx. The interface for the client has a text field and text area, the text field is where the user will insert their desired message to send. And the text area will store all messages going in and out of the client. The server interface consists of just a text area, the user doesn't use that interface, it simply displays all actions that happen such as displaying when the client connects, the host name and IP address of the client and it also displays the client's conversation.

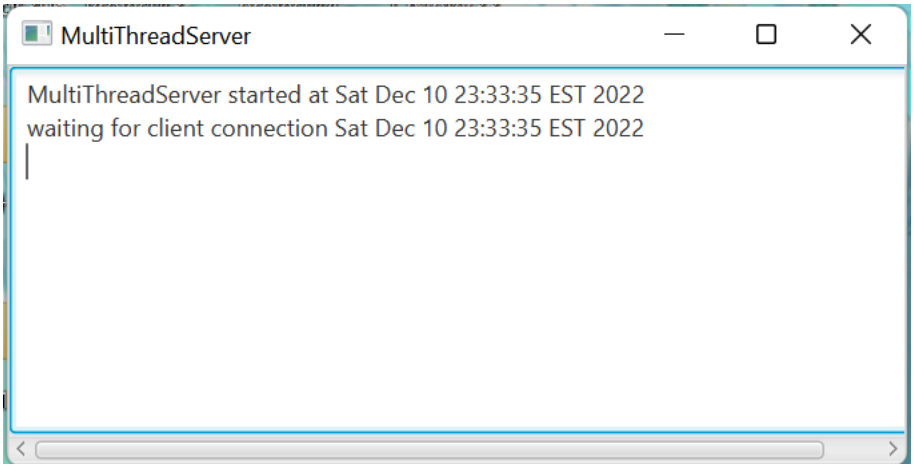
After the two interfaces were made, connecting, and creating the sockets was next. On the server side the socket is contained within a thread, it consists of just the port number. There are only 2 sockets in the program that are to be connected – client 1 and client 2. Both are connecting the same way, the socket is made and is accepting by the server socket, the output stream is also connected to the socket created for each client. Also, with each instance the server display to the user what is happening. This is all contained within a try/catch so if errors occur the user will be notified. (The try/catch al contains the second thread that handles the clients). On the clients side the sockets is created using the IP address and post number, the sockets are then connected to the input and output streams.

Lastly, how the messages are transferred over. On the server side the messages are handled in a class called Handle Client that has the parameters of the 2 sockets for the 2 clients. Each client also gets a corresponding input and output stream that they will communicate through. In a try/catch, if the client's input is available then the message is assigned as the input from the client, that's the assigned to the output of the other client with a out put message of what the "friend" sent. And the server will also display what the client sent. This created for both client 1 and client 2. On the clients side an action event takes in the messages that the user writes on the client's interface, this is then sent to the output server to be displayed. After a thread is created pull in the sent messages to the client. This the thread checks if the input is available, if it is then the message is set as the inputted message that the other client has sent over and displayed.

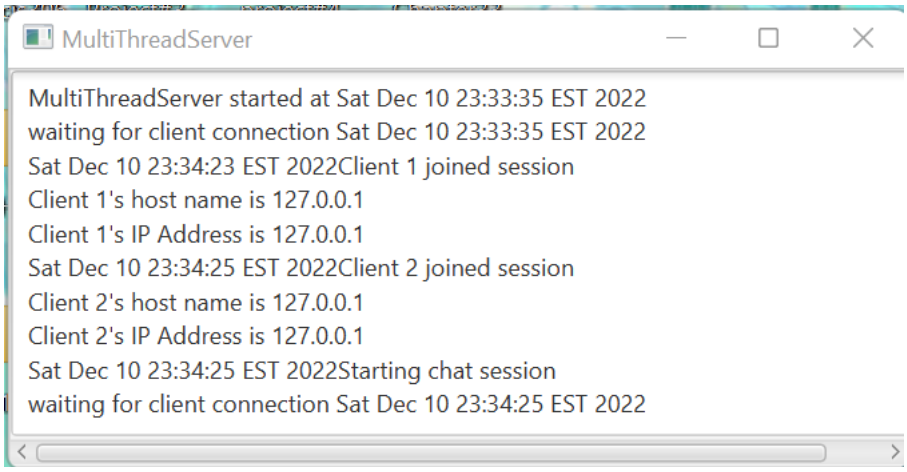
Through this the code is able to work!

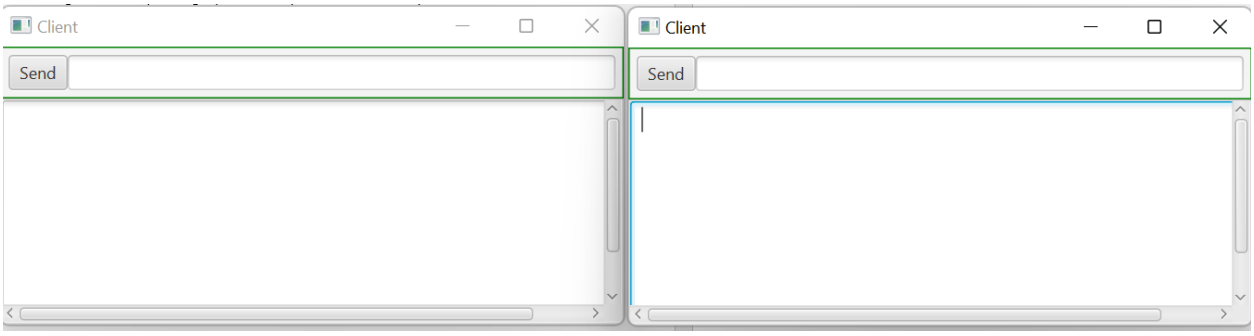
Screenshots:

Server starts:



Clients coming in:





Clients interacting and server log: