Lab – Java – Network Programming – Basic Stream Sockets

Overview

Write an application that uses stream sockets. You will need to write two applications, one for the server and one for the client.

Create Server Application

Create a new Java console application for the server

Server Application Specifications

- Create a server socket.
- Listen for a client connection request on the server socket (this application will only allow for one client connection). The listening server socket will return a client socket instance when a connection request comes in.
- After it creates the client connection it should read data from the client socket and display it on screen.
- After reading data it should write data back to the client. For example, “Hello from the server”.
- End the program after sending the message to the client.

Create Client Application

Create another Java console application. This application will be for the client.

Client Application Specifications

- Create a socket.
- Make a connection to the server.
- Send a message to the server. For example, “Hello from the client”.
- After it sends the message it should read data from the server and display it on screen.
- End the program after reading the message from the server.

Test the Programs

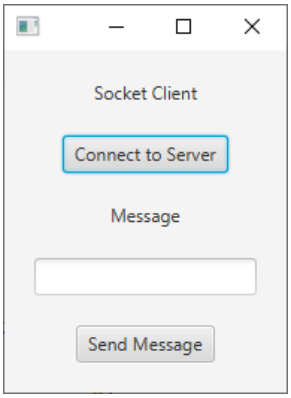
- Start the server application.
- Start the client application.
- Each program will have its own output windows. You should see the messages received in the respective output windows of each application.

Challenge - JavaFX GUI Client Application (only send message)

Write a JavaFX GUI front end for the client.

- The GUI should have a connect to server button, a message text field, and a send message button.
- Add member variables to the PrimaryController:
    - Socket
    - PrintWriter (initialize from the Socket variable right after the Socket is created)
- When the connect to server button is pressed it should connect to the server.
- When the send message button is pressed it should get the text from the text field and send that text to the server.

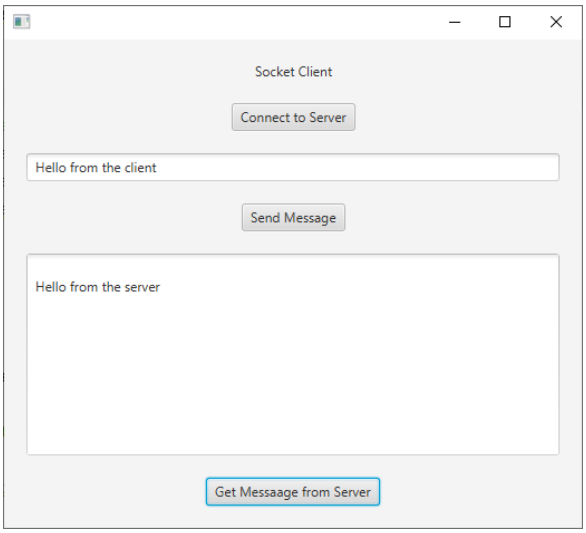
Here is a screenshot of the JavaFX GUI client:



Update Client GUI – Check for message from Server

- Add a button to check for a message from the server. The client GUI will block when reading from the input data stream until a message is received. You can add a TextArea to display the message received from the server.
- When running the client GUI you must press the buttons in the following order because of the way the server interacts with it:
    1. Connect to Server
    2. Send Message
    3. Get Message from Server

Here is a screenshot of the JavaFX GUI client after pressing buttons in the above order:



Update Client GUI – Get Message from Server Using Thread (console)

Use a thread to receive messages. The thread should start listening for messages right after the connection is made to the server. Messages will be printed on the console. Here are instructions:

- Create a class that implements the Runnable interface (for example ClientSocketListener).
    - Member variables: Socket, BufferedReader
    - Create a constructor that takes a Socket as a parameter. It should set the socket member variable. It should also setup the BufferedReader member variable from the socket (the BufferedReader will be the input stream for the socket).
    - run method. The override of run should have an infinite loop (while (true)) that keeps getting a message from the input stream and printing it on the console. Inside the loop you should also check if the value returned from the input stream is null. If it is null, it should break out of the loop (null means the other end of the socket was closed).
- Start Thread. The controller class should create a thread that runs your Runnable class after it connects to the server. You can start the thread however you want.
- Note: If you use an Executor it must be shutdown when the window closes. Do the following:
    - PrimaryController cleanup method. Add a cleanup method to the controller class. It should shut down the executor service.
    - You will need to update the code in the start method of the App class so that it automatically calls the cleanup method on the controller when the window is closing. Check in the Threads slides for an example of how to do this (the slide is named JavaFX GUI and Executor Service Shutdown).

Update Client GUI – Show Thread Received Message in GUI

Show the message received from the worker thread's socket inside the GUI. The worker thread does not have access to the GUI controls so you will need special code to write to the GUI. Here is what you should do:

- Update the controller class
    - Add a method that takes a message as a parameter and puts that message in the messages TextArea (this method will get called from the ClientSocketListener class).
- Update the Runnable class (ClientSocketListener)
    - Add a member variable to store a reference to the controller class.
    - Update your Runnable class constructor. Add a second parameter for the controller class. Set the controller member variable.
    - Call the method you just added to the controller class to print the message received (do this in the run method). You will need to call this method in a special way since other threads do not have access to the controls that are in the GUI thread. Check in the Threads slides for an example of how to do this (the slide is Update GUI from Other Thread).
- Remove the Get Message from Server Button (thread is always listening for messages).
    - You can remove the Get Message from Server button from the primary.fxml file.
    - Remove the button's event handler from the controller class.