Higher Diploma in Science (Software Development)

Network Programming Project

Project Name: NetworkChatProj

Student Name: Edivagner Ribeiro

Student ID: G00411275

In this menu-driven program, we use a simple command-line user interface for the server and another one for the client. The implementation of the web-based chat application uses the Java Socket API and to support multiple users we use a thread pool to handle the connections.

When a server runs on a specific socket that is bound to a specific port number. The server just waits, listening to the socket for a client to make a connection request [1, 2]. To establish communication between two machines/software it is necessary that each one knows the characteristics of the other, that is, in this case, the localhost port they are working on [3].

Since this is a classic networking problem, there are millions of tutorials and references. Among the many consultations carried out during this development, the main guidelines are the references [3-6]. In particular, Jenkov, J. [6] presents a good example of how to handle server exceptions.

To run this program, we need two or more terminals.

- Server:
  - Compile all files in folder src: **javac ie/atu/sw/server/***
  - To run the server via terminal: **java ie.atu.sw.server.ServerRunner**
- Client:
  - Compile all files in folder src: **javac ie/atu/sw/client/***
  - To run the client via terminal: **java ie.atu.sw.client.ClientRunner**

In this project, for the server interface and the client interface, we created a menu, where on both sides we can check which ports are working as a server. After checking the open ports, it is possible to configure a port to establish the server-client connection by a socket.

The Server class implements Runnable to start the server using a thread pool. The server starts up and waits for socket connections on a specific port user-defined or with default port 13. * The ConnectionsHandler class handle multiple client connections and the exchange of messages via socket. When a client enters, they receive the message to define a name. The server sends the message to every other connection a new client has entered.

Three functions are defined for communication between the client and server:

- "\chatConnect" : returns the message that it is connected;
- "\name" : the next entry will be the client's new name;
- "\q" : close the connection with the server;

When there are no more connections to the server, it will shut down.

The client interface has a ClientMenu class that creates a menu that provides the user with 4 different options:

- Search for servers on localhost
- Set port for localhost Chat Client
- Run Chat Client
- Quit

When the user enters a valid port for the chat server, the user receives a message from the server to enter a name and thus can start chatting with the other clients in the chat room.

If the user set a not valid port, he will receive a message that communication cannot be established, and the application will be closed.

References:

[1] Docs, O. (no date) *What is a socket?*, *What Is a Socket? (The Java™ Tutorials > Custom Networking > All About Sockets)*. Available at: https://docs.oracle.com/javase/tutorial/networking/sockets/definition.html (Accessed: January 15, 2023).

[2] Docs, O. (2022) *Java development kit version 17 API specification*, *java.net (Java SE 17 & JDK 17)*. Available at: https://docs.oracle.com/en/java/javase/17/docs/api/java.base/java/net/package-summary.html (Accessed: January 15, 2023).

[3] Sierra, K. *et al.* (2022) “17 - Make a Connection,” in *Head first java*. Cambridge: O'Reilly, pp. 590–641.

[4] Liao, J. (2017) *JOD EP1: Building a multi-user chat application in Java - Part 1: Serversocket*, *YouTube*. YouTube. Available at: https://www.youtube.com/watch?v=cRfsUrU3RjE&t=774s (Accessed: January 15, 2023).

[5] *Simple TCP chat room in Java* (2021) *YouTube*. YouTube. Available at: https://www.youtube.com/watch?app=desktop&v=hIc_9Wbn704 (Accessed: January 15, 2023).

[6] Jenkov, J. (2014) *Thread pooled server - jenkov.com*. Available at: https://jenkov.com/tutorials/java-multithreaded-servers/thread-pooled-server.html (Accessed: January 15, 2023).