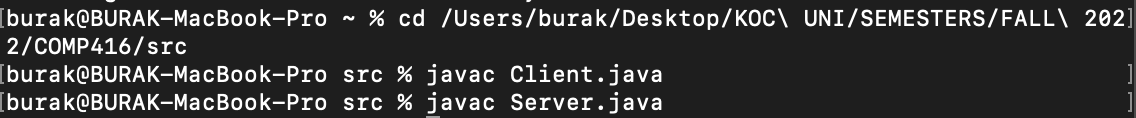
Comp416 Project 1.1

**ReadMe**

Firstly, open up a terminal window and change the directory to the project’s directory by cd command. Compile the Client.java and Server.java files using “javac Server.java” and “javac Client.java” commands. Then execute the compiled files with necessary arguments using “java Server <port-number> <timeout-value>” and “java Client <port-number>”. An example is given below.



**Execution**

Text

Description automatically generatedText

Description automatically generated

In the upper screenshots, left hand side is the server side and the right hand side is the client side.

1. Server side code is executed with port number as 1111 and timeout given as 35000 which refers to 35 seconds.
2. After establishing the connection, both sides show the socket addresses.
3. The timeout setting of the connected socket is shown for both sides.
4. An example chat between server and the client can be seen in the screenshots.

Text

Description automatically generatedText

Description automatically generated

1. When client or server side writes “quit” keyword chat is ended and connections are closed.

Text

Description automatically generatedText

Description automatically generated

Text

Description automatically generatedText

Description automatically generated

1. In case of a timeout like in the above screenshots, chat is ended and server closes the connection.

Text

Description automatically generated with medium confidence

In the Client file, I get the port value from the user as an argument. Then, I initialized the socket, readers, and writers for the client-server interaction. After establishing the connection, I printed out the remote socket address and the timeout value. I initialized userInput and inputLine variables for getting the output of the client and receiving the message of the server respectively. Firstly, timeout value that is sent from the server is read and the timeout value is set to that specific value. Infinite loop that can only interrupted by exceeding the timeout limit or a quit message of either client or server side. In the loop, client and server writes a messages one by one. Client writes a message then it sends to the server. After that, the message of the server is received by the client. The procedure continuous until an interruption that I mentioned above. In case of an interruption, I inform the user about a quit operation or a timeout operation. In both cases, I close all the sockets, readers, and writers in the end.

Text

Description automatically generated with medium confidence

In the server side, it is very similar to the client side. The port number and the timeout value are given as first and second argument from the user respectively. Server socket, client socket, writers, and readers are initialized. Timeout value is set for the socket and sent to the client. After establishing the connection, remote socket address and timeout values are printed out. Then, I defined inputLine and serverInput for getting the message of the client and giving the input of the server respectively. It waits for the client to send a message. After receiving the message of the client, the server is prompted to write a message. In the infinite loop, it continuous until an interruption. An interruption can only occur due to exceeding the timeout limit or a quit message of either client or server side. In case of a timeout exception it is handled by informing by printing out the exception. In both cases all the socket connections, readers, and writers are closed in the end.