Issued to
Professor Solanki
CS 4390.003
Computer Networks
Spring 2023

Network Application Project Report

Due April 27, 2023

TEAM MEMBERS

Bondith Sovann - BXS210029

Kumar Gullapalli - SVG190000

Elliana Brown - ESB190002

PROTOCOL DESIGN

Message Format for sending and receiving math calculations:

Math expressions are input to as Strings composed of numbers and operators. This string is then passed to the Calculator.java file where it is broken down into characters and, in the case of digits, transformed into doubles. Once the expression is solved, the result returns to the TCPServer.java file as a double. The server then packages the result double into a string and gives it to the client that input the input the expression.

Message format for joining and terminating connection:

Once the server socket is created, it prints out that it was created, what port it was created on and that it is waiting for client connection. In order to connect to the server socket, a client socket must be created with the same port number. Once the client and server are connected, all information travels via an input and output stream. With these streams, the client sends their name as a string for the server to log (along with the local date time), in order to be completely connected. To terminate the connection, the client must input “E” instead of a math expression. This closes the input stream, the output stream, and the client socket. The server then logs the termination with the client’s name, the local date time, and the client’s Inet address host name.

Format for keeping logs of clients' activities at server side:

When a client connects, the server requests and logs the client's name as well as the local date time of the connection. The server logs the results of each entered math expression and logs the client's name, local date time, and the client's Inet address host name upon connection termination.

PROGRAMMING ENVIRONMENT

For this project, we used Windows 10 and Visual Studio Code during development. However, it will run properly on any operating system that has a Java Development Kit (JDK) installed.

HOW TO COMPILE AND EXECUTE PROGRAM

Compile with javac

To compile and execute the program, all files must first be compiled using the **javac *.java** command. Then, the server must be executed with the command **java TCPServer 1234** (1234 is the port number in this example). Finally, the client must be executed with the command **java TCPClient 1234** (1234 is the port number, corresponding to the server port number in this example).

Compile with Makefile

Another option for compiling is to use the makefile. This can be done using the **make -f Makefile** command. Once the code has been compiled, the server and client can be executed as described above.

PARAMETERS NEEDED

Port Number

The only parameter needed for a TCP client/server socket connection is the port number. It should be noted that it is essential that the client port number and the server port number are the same. For example, java TCPServer 1234 and java TCPClient 1234 where 1234 is the port number. However, while port number 1234 is used throughout this report as an example, it is not the only acceptable port number. The port number is set when the server and client is executed. It can be any available port number as long as the server and client are listening at the same port.

CHALLENGES FACED

This project wasn't too difficult as a very similar assignment regarding setting up sockets was given in the Operating System Concepts class. As such, the only real challenges involved the Calculator.java file. It took several sessions of troubleshooting and debugging to make the file properly accept so many different kinds of input.

WHAT YOU HAVE LEARNED

While much of the project was something of a review on socket connection, there were still many valuable topics that can be learned from this project. For instance, as we have covered in class, it is vital that the server socket be created and made to listen for the client before any connection is attempted on the client's part. Without this, the client would have nothing to connect to. Another important lesson involves the use of threads for the server to be able to service multiple clients at a time. Each client has their own independent thread, so they can run concurrently..

OUTPUT SCREENSHOTS OF APPLICATION

Server Side

```
C:\Users\sahit>cd Documents/School/"Spring 2023"/CompNetProject

C:\Users\sahit\Documents\school\spring 2023\CompNetProject>java TCPServer 1234
Server started on port 1234
Waiting for client connection...
Waiting for client connection...
Kumar is connected! on 2023/04/22 15:52:35Waiting for client connection...
David is connected! on 2023/04/22 15:52:56
Kumar requested solution for (1 + 4)
David requested solution for (5 * 4) + 20
David requested solution for 5*5
Kumar requested solution for 12*96
Kumar is disconnected on 2023/04/22 15:54:23: 127.0.0.1
David is disconnected on 2023/04/22 15:54:26: 127.0.0.1
```

Client1 Side

```
C:\Users\sahit\Documents\school\spring 2023\CompNetProject>java TCPClient 1234
What is Your Name ? Kumar
Kumar, You are connected, Successfully!

Please Enter the Math Expression to evaluate or E to exit: (1 + 4)
Your result is: 5.0
Please Enter the Math Expression to evaluate or E to exit: 12*96
Your result is: 1152.0
Please Enter the Math Expression to evaluate or E to exit: E

C:\Users\sahit\Documents\school\spring 2023\CompNetProject>
```

Client2 Side

```
C:\Users\sahit\Documents\school\spring 2023\CompNetProject>java TCPClient 1234
What is Your Name ? David
David, You are connected, Successfully!

Please Enter the Math Expression to evaluate or E to exit: (5 * 4) + 20
Your result is: 40.0
Please Enter the Math Expression to evaluate or E to exit: 5*5
Your result is: 25.0
Please Enter the Math Expression to evaluate or E to exit: E

C:\Users\sahit\Documents\school\spring 2023\CompNetProject>
```