ECEN489 Mobile Applications with Android MP 3 - Due Date: 01/25/2019

TCP Sockets in Java

Lecturer: Srinivas Shakkottai *Scribe: Self*

1 Introduction

Please read all sections of this document before you begin coding. The purpose of this machine problem is to familiarize you with network programming. You will obtain, compile and run a simple network program on the Smart Phone Laboratory workstations, then hand in some files. The extensions to the code will introduce you to one method of framing data into individual messages when using a byte stream abstraction such as TCP for communication. Refer to The Java Tutorials [2] for sample code on how to create sockets and send messages using them.

2 Problem Statement

In this assignment, implement a server and a client using Java sockets. The server and client must use a TCP connection to communicate with each other.

Server

The server must have the capability to handle multiple clients simultaneously. The server takes as console input the port number on which clients would connect to it, the maximum number of clients that can connect simultaneously, and the message to be passed as a string. The server should create a thread for each connected client. Messages should be passed in a character buffer.

```
>./server <server port> <max. clients> <message>
```

During your demo, we will use our own version of the client that will attempt to connect to your server. It must successfully receive a 50 character message from your server.

Testing your Server Using Telnet (Optional)

You can use telnet to verify if your server can correctly receive messages. Once connected you can send messages and verify if your server receives them.

```
>telnet <server ip> <server port>
```

The other option is to test using the client that you develop below.

Client

The client takes as console input the IP and the port number of the server in order to connect and closes the connection when the input "\disconnect" is typed on the console. The client should ignore any other messages.

```
>./client <server ip> <server port>
```

During your demo, we will use our own version of the server. Your client must connect with the server, which will attempt to send a 50 character message to your client. The client must successfully receive the message and display it on the console. Use a buffered reader to receive the message.

Extension

TCP sets up a two-way connection. Hence, after connecting to a server, the client can also send it messages. Modify your server to handle messages sent by the clients and display them with the IP address of the client in the console at the server.

```
<client ip>:<client port>: <message>
```

3 Evaluation Guidelines

1. Your server will be checked for correctness with our client and vice versa.
2. Enter arbitrary strings as input to the client/server to test for error handling.

References

[1] Head First Java. HFJ Chapter 15

[2] http://docs.oracle.com/javase/tutorial/networking/sockets/index.html