SW409 Advanced Programming in Java
Explanation About Structure Of Classes for Final Project
<Group Chatting Room>

Semester: Fall 2018

Name: Yifei Li

Due date: 12/13/2018 10:00pm

Description:

1. Features

The basic features:

- implement a text based chat program
- allow (at least) 2 of clients to communicate from different computers at a time (multithreaded)
- use sockets for connection in program

The additional features:

- create GUI for server and clients separately
- users could easily use the program though the GUI
- server could setup the maximum number of clients and port number each time when start
- client could setup his/her name before connecting to server
- all of GUIs display the online client list dynamically

2. Structure

My chatting room program has three java classes, including Server, Client, and User. Server class is responsible for setting up the server socket, invoking and closing the ServerThread and ClientThread, in which methods of sending and receiving messages are built. Client class is used for setting up a socket and sending connection requirements to ServerSocket. Also it contains methods for sending and receiving message. The User Class is created for clients used as objects in the other two classed.

Server class contains Server() constructor, in which the server's GUI is created. Functionalities are implemented by allowing the user to press the buttons on the GUI, such as "Start" button for serverStart(), "Stop" button for closeServer(), "Send" button for send() messages. Some small methods with few code are created for frequently usage in other methods, such as sendServerMessage(String message). Two main inner classes, ClientThread and ServerThread are also created in the Server class. Each of them has its own constructor for easy value reference.

Client class contains Client() constructor. It has similar function as the one in Server class. Client GUI, similar to Server is created in the constructor. All the

functionalities could be used only by clicking buttons. Pressing the “Start” button will call the connectServer(port, hostIp, name) function with three important parameters. “Stop” button will invoke closeConnection(). Different from Server class, Client class only need to deal with the MessageThread(BufferedReader reasder, JTextArea textAea). The first parameter in the constructor of MessageThread will pass the value of message strings for sending and receiving. The second parameter of MessageThread is telling the instantiated massage threads that all the messages are going to be added in the textArea, which is the place displaying all the messages. Synchronization solution is used for passively closeCon() of a client as exceptions might happen because of it.

User class has its own constructor with user’s name and user’s ip as parameters. The set() and get() methods are for the instantiated user object to refer the value. With this class, I could easily create HashMap list for all the online users and display their name in the GUI.

In conclusion, I decomposited the problem based on the requirements which makes my structure relatively high cohesion and low coupling.