CS3524 Assessment 1 Messenger

Name: Hasan Ahmedov

ID: 51768277

1. Description

For this assessment we had to build a Java Messenger application using Socket programming. It is a console-based program that is launched from the command line. The application consists of two parts: server and client. Each part can run independently on separate computers in the same network, e.g. Local Area Network (LAN).

There can be multiple clients connect to a server and they can chat to each other, just like in a chat room. After getting connected to the server, the user must provide his or her name to enter the chat. The server sends a command list and displays the names of currently online users to the new user. Moreover, a feedback message is sent to the client every time a command is used in the console.

Every user is notified when a new user has connected and when a user has quit. Each sent message is displayed with a prefix of the sender's username to keep track of who has sent the message.

A user of the client application can engage in conversations with either all of the other users, one single user (personal messages) or a particular group of users (group messages). The server can also handle users subscribing to topics of interest by forwarding messages according to clients subscribing to particular keywords in messages that represent topics of interest

The commands that a user can use inside the Messenger application are as follows:

- '**/users**' – prints out the names of currently registered users
- '**/all** <text>' – sends a text message to all currently available users

- ‘**/create** <group name>’ - creates a group which online users can join
- ‘**/join** <group name>’ - lets users join a group (if it exists)
- ‘**/leave** <group name>’ - allows users to leave a group (if they already members)
- ‘**/remove** <group name>’ - removes a group from the Messenger application
- ‘**/pm** <name> <text>’ - used when a user wants to send a personal message to an available user
- ‘**/gm** <group name> <text>’ - sends a text message to a group (if group exists)
- ‘**/groups**’ - prints out available groups
- ‘**/topic** <topic name>’ - lets users create topics of interest and whenever a message is sent by some client containing at least one of the subscribed keywords, it will be forwarded
- ‘**/topics**’ - prints out available topics of interest
- ‘**/sub** <topic name>’ - allows users to subscribe to a topic
- ‘**/unsub** <topic name>’ - unsubscribes a user from a topic (if already subscribed)
- ‘**/exit**’ - used to unregister the user and quit the Messenger application

The Java classes implemented for the Messenger application are:

- ‘**MessageServer**’ - starts the server, listening on a specific port. When a new client gets connected, an instance of ‘**ClientConnection**’ is created to serve that client. Since each connection is processed in a separate thread, the server is able to handle multiple clients at the same time.
- ‘**ClientConnection**’ - instantiated by the server whenever a client connects to the server and acts as the “receiver” for any messages received from such a client (calls the methods on the server side).

- ‘**MessageClient**’ - implements a client application that can connect to this server. It starts the client program, connects to a server specified by hostname/IP address and port number. Once the connection is made, it creates and starts two threads ‘**MessageReadThread**’ and ‘**MessageWriteThread**’.
- '**MessageReadThread**' for reading server's input and printing it to the console. It runs in an infinite loop until the client disconnects from the server.
- '**MessageWriteThread**' is responsible for reading user's input and sending it to the server. It runs in an infinite loop until the user uses the '**/exit**' command to quit.

2. How to run

To enable you to quickly run the Messenger application, a make file is provided that compiles all the required files. To run the make file you must launch a terminal window and navigate to the appropriate directory where the make file is located (where XXXX is cs3524_assessment_1_u01hba17/CGS_A5_A1 in our case):

➢ **your-linux% cd XXXX**

Once they are in the appropriate directory just run the following command in the console to compile the files (**messengerclean** to clean the folder):

➢ **make messenger**

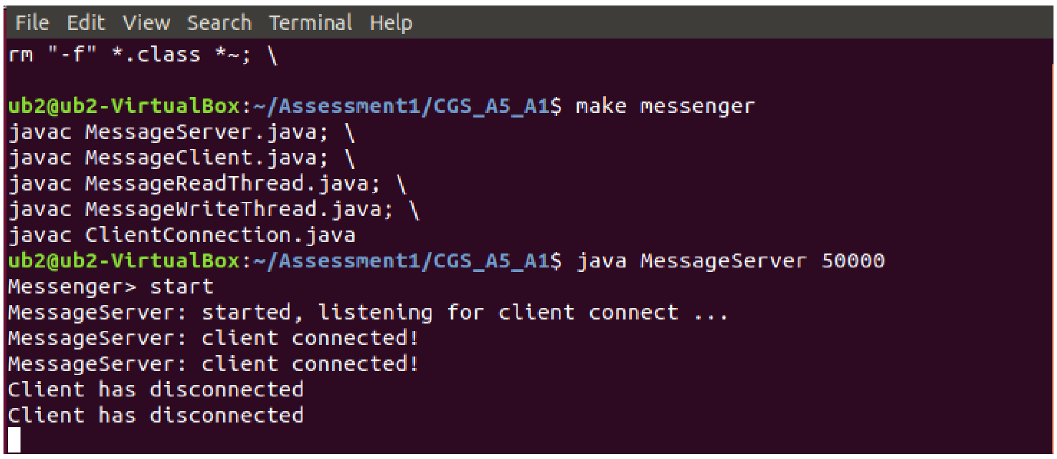
The server is started by providing a port number (50000 and above) at the command line:

➢ **java MessageServer 50000**

The client is started by specifying hostname and port of the server on separate terminal window:

➢ **java MessageClient localhost 50000**

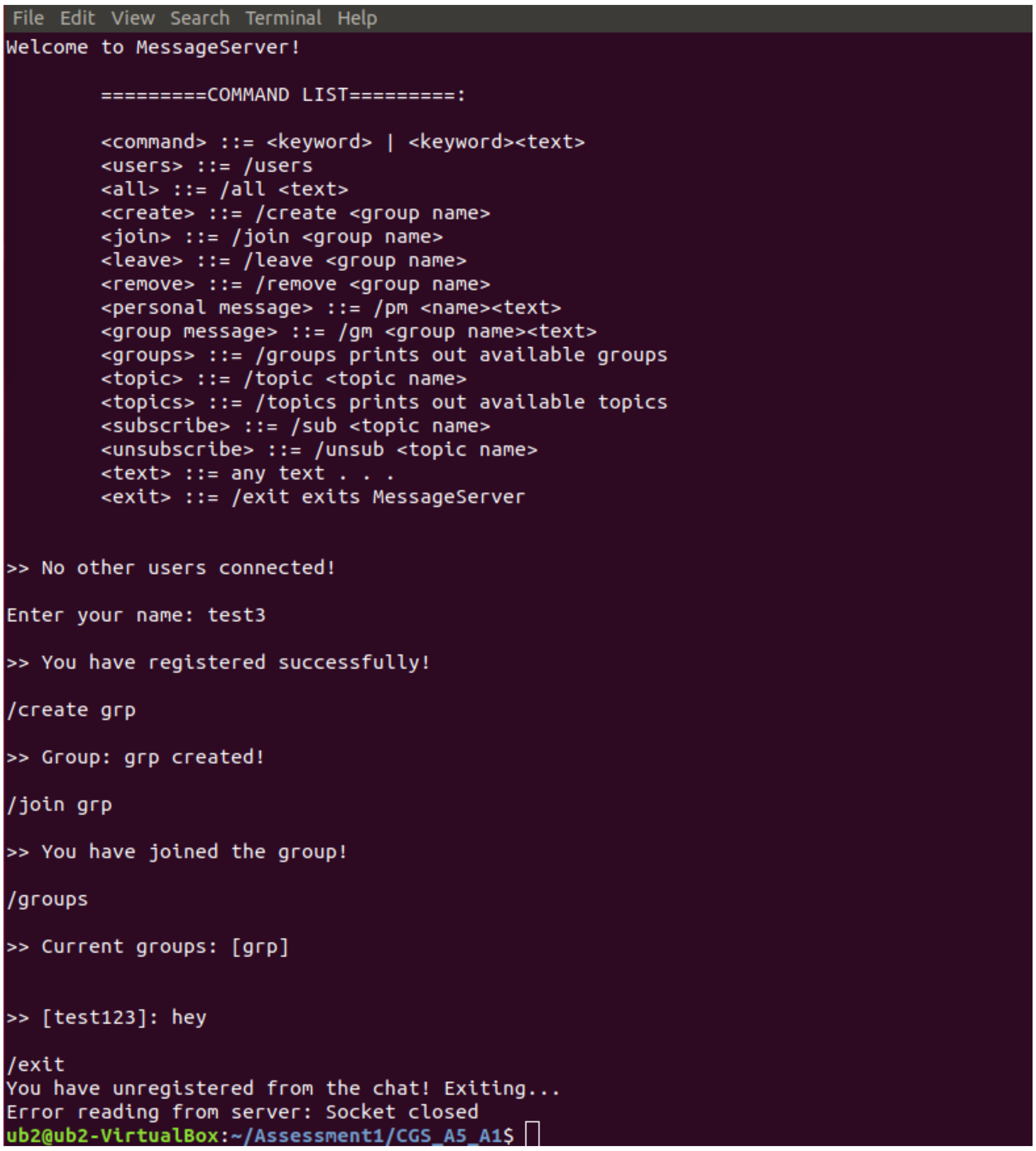
3. Messenger sample output (execution)



```
rm "-f" *.class *~; \

ub2@ub2-VirtualBox:~/Assessment1/CGS_A5_A1$ make messenger
javac MessageServer.java; \
javac MessageClient.java; \
javac MessageReadThread.java; \
javac MessageWriteThread.java; \
javac ClientConnection.java
ub2@ub2-VirtualBox:~/Assessment1/CGS_A5_A1$ java MessageServer 50000
Messenger> start
MessageServer: started, listening for client connect ...
MessageServer: client connected!
MessageServer: client connected!
Client has disconnected
Client has disconnected
```

Figure 1. *Running the make file and message server.*



```
Welcome to MessageServer!

        =========COMMAND LIST=========:

        <command> ::= <keyword> | <keyword><text>
        <users> ::= /users
        <all> ::= /all <text>
        <create> ::= /create <group name>
        <join> ::= /join <group name>
        <leave> ::= /leave <group name>
        <remove> ::= /remove <group name>
        <personal message> ::= /pm <name><text>
        <group message> ::= /gm <group name><text>
        <groups> ::= /groups prints out available groups
        <topic> ::= /topic <topic name>
        <topics> ::= /topics prints out available topics
        <subscribe> ::= /sub <topic name>
        <unsubscribe> ::= /unsub <topic name>
        <text> ::= any text . . .
        <exit> ::= /exit exits MessageServer


>> No other users connected!

Enter your name: test3

>> You have registered successfully!

/create grp

>> Group: grp created!

/join grp

>> You have joined the group!

/groups

>> Current groups: [grp]


>> [test123]: hey

/exit
You have unregistered from the chat! Exiting...
Error reading from server: Socket closed
ub2@ub2-VirtualBox:~/Assessment1/CGS_A5_A1$
```

Figure 2. *Starting a message client and trying out some commands.*