|  |  |
| --- | --- |
| http://logodatabases.com/wp-content/uploads/2012/03/java-logo-large.png  Networks 2a  A Multi-Threaded Client/Server Application using Sockets in Java | CSC3002F |
|  |  |

Jonathan Leyland LYLJON002

Alfred Mothapo MTHMAS016

Adam Meier MRXADA002

## Code Explanation

**ClientThread.java**

|  |  |
| --- | --- |
| **Method** | **Description** |
| **public ClientThread(String host, int port, String username)** | Constructor which creates a new connection, initialises input/output streams and sets the username of the client. |
| **public void run()** | Run function for the thread which runs after it has been started. Contains a while loop which controls the prompt to enter a message, read a message and add it to the history of the chat. |
| **public void writeToServer()** | This method contains a while loop which loops while the ClientThread is running. Upon message input, it receives it and sends it to the server. |
| **public void closeConnection()** | Called at the end of the thread’s life; closes input/output streams and closes the socket. |
| **public static void main(String [] args)** | Main method which accepts the username and server name as arguments. It then creates a new ClientThread using these parameters and calls the thread to start. |

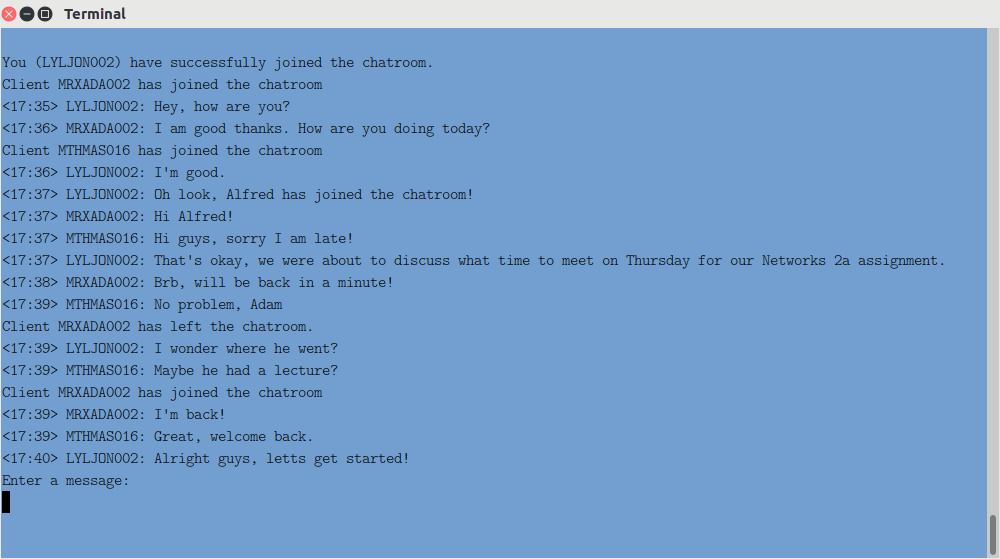
**Server.java**

|  |  |
| --- | --- |
| **Method** | **Description** |
| **public Server()** | A Simple constructor which creates a socket for the server and prints the fact that the server is awaiting connections. |
| **public void awaitingConnections()** | Contains a while loop which runs if the server is online. As soon as a client joins, it creates a new instance of ClSv.java and starts it. |
| **public static void main(String args[])** | Calls the constructor for Server.java and then tells the instance of the server to wait for connections. |
| **public static void clearScreen()** | Simple clearing method to clear the terminal and make it look neat |

**ClSv.java**

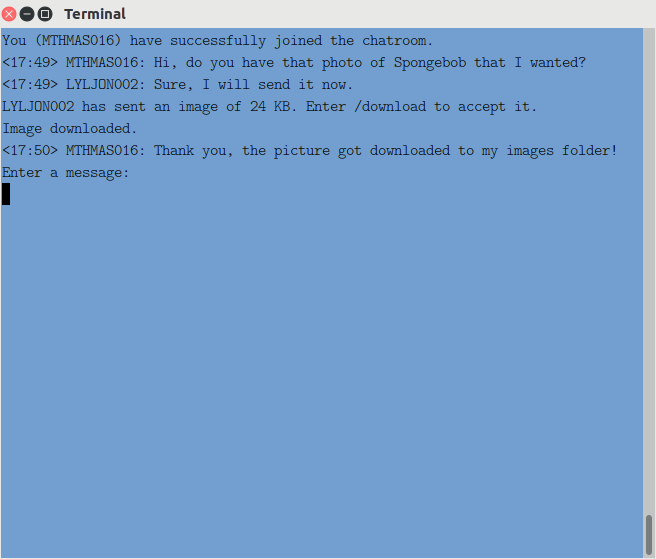
|  |  |
| --- | --- |
| **Method** | **Description** |
| **public ClSv(Socket clientSocket, ArrayList<ClSv> clSvArrayList)** | A constructor which accepts the socket from the client, and an array list which contains other clients on the server. The input and output streams are taken from the client. |
| **public String getUsername()** | Returns the username of the client. |
| **public void closeConnection()** | Closes the input and output streams that were accepted during the constructor |
| **public void run()** | The run method which is run after Server.java calls start. This method is responsible for writing any message to all other clients. Firstly, it tells other users if someone has joined the chatroom. It also tells the other users if somebody leaves the chatroom. Finally, it takes any message a client has sent and displays it for all other clients to see. |

## Screenshots



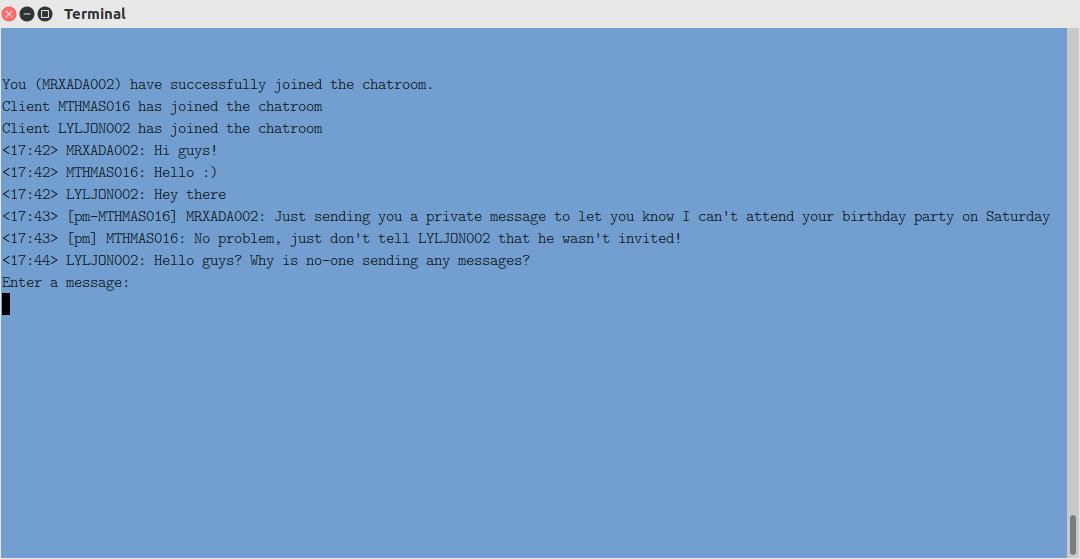
**Figure 1.1 – Three users engaging in a text chat after they join the server.**

Users can join and leave the chatroom as they wish. A prompt is displayed on other client’s screens upon a user connecting or disconnecting. Multiple messages can be sent at once. The username can be changed by choosing a different one in the constructor. A user can disconnect from the server by calling “/quit.” They reconnect by calling the constructor.



**Figure 1.2 – A user requesting to send a picture to another client, and the picture sending once the other client has accepted.**

A user can send a picture by calling “/send” followed by name of the image with its type extension. The picture should exist inside the folder called “images” for it to work. Other uses will receive a prompt with the image name as well as the size of the image. Other users can call “/download” to save the image, where it will be saved to their images folder.



**Figure 1.3 – Two users in a chatroom with three people in to sending private messages to each other.**

A user can send a private message to another user by calling “/<USERNAME> <message>”. The message shows up with [pm] to indicate that no other clients can read that message.