**CSCI 4311**

**Program Assignment 1 Report**

**Shijun Jiang**

**Code Overview**

**Server.java** implements a multi-threaded server for handling multiple client connections and message delivery. The server listens to a specified port through ServerSocket and waits for the client’s connection. The server maintains two key data structures: Set<String> userNames and Set<PrintWriter> writers. userNames is used to ; the writers collection stores the output streams of all clients, allowing the server to broadcast messages to all connected clients. The Handler class implements the Runnable interface and represents the processing logic for each client connection. It reads messages sent by clients and broadcasts them to all other clients.

*store all user names of the current connection*

*stores the output streams of all clients*

文本

中度可信度描述已自动生成

*Clock display with time zone, output to console*

图形用户界面, 文本, 应用程序

描述已自动生成

**Username processing**

Get the current time and format it

***Create a PrintWriter object (out) for sending data to the client***

***Create a BufferedReader object (in) for reading input from the client***

**4 branch situations and their processing:**

1. If the read message is **null**, it means that the client has disconnected, and the loop will jump out through break to end the current client’s processing.
2. If the message sent by the client is **Bye(bye)**, the server will broadcast the user’s away message to all clients and end the current thread.
3. If the client requests a list of all currently connected users (by sending the **AllUsers** command), the server will build and send a message containing a list of all currently connected users and their connection times. This list is constructed by synchronously accessing the userNames collection and the userConnectionTimes map.

**文本, 应用程序

描述已自动生成**



(4)If the message received is **NOT** the special command above, the server broadcasts the message to all connected clients. This is accomplished by looping through the writers collection and sending a formatted message to each PrintWriter.

**Client.java** implements client logic for connecting to the server and sending and receiving messages. The client establishes a connection with the server through Socket and communicates with the server using input and output streams.

There are two main threads: one that reads the user’s input and sends it to the server, and the other that continuously receives messages from the server and prints them to the console. In order to achieve real-time user interaction, the client does not need to wait for the server's response before sending a message to the server.

文本, 应用程序

描述已自动生成



***Used to print the total running time, end time, final used memory and other information at the end of the program***

***Create a Socket to connect to the server***

图形用户界面, 文本, 应用程序

描述已自动生成



**Start a new thread to read messages from the server asynchronously**

*If the user enters bye, print a disconnect message and set the running flag to false to end the main loop.*

Output:

**1）Server.java**

文本

描述已自动生成

After the user triggers the bye command on the client and disconnects the connection, the server will output a confirmation command to disconnect the connection.

**2) Client.java**

**User1-Shijun**

文本

描述已自动生成

User1 input

Print the welcome message after entering the username

Execute the bye command to trigger the disconnection

When the program ends, summary information is printed.

**User2-happy2024**

文本

描述已自动生成

The server broadcasts user1’s bye information to user2.

**User2 input**

Execute the bye command to trigger the disconnection

Print the welcome message after entering the username

Execute the allusers command to print the currently active user names and their connection times in chronological order.