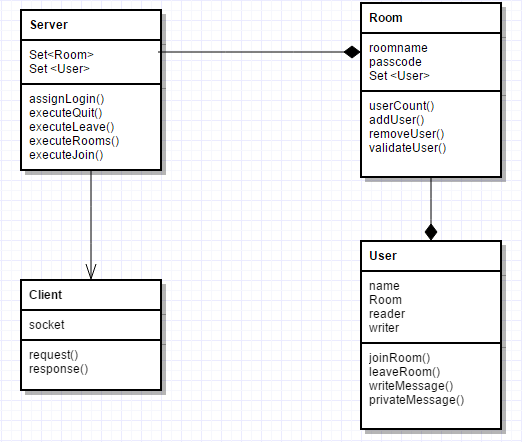
Chat Server

Chat serve allows users to communicate with each other. After connecting to the server, user is asked to pick a username. If the same username is not been used by other user then, that username is assigned to the new user. This functionality is synchronized so that a user can pick unique username irrespective of other users who might be requesting username at the same time. User can exit just after connecting to the server. Once username is been picked up by uses he can execute join, leave and rooms commands.

Once user has joined a room, he can broadcast a message or can send a private message to other user in the same room.

Class Diagram



Server Operations

* Display the list of available rooms.

Server has some default available rooms (two public rooms and one federation). User can join any room from this list.

New functionality to create a federation can be easily given to user.

Syntax- /rooms

* Join chat room

User joins the room. If the room name provided by the user is wrong then the error is sent and list of available rooms is displayed.

Syntax- /join <room name>

* Join federation

To join the federation user need to provide the pass code while joining, ‘-p’ argument is used to provide the pass code. This pass code is validated and then the user gets access to the federation.

Syntax- /join <room name> -p <pass code>

Example: /join fed -p weeby (‘fed’ is default available federation having pass code ‘weeby’)

* Leave the chat room or federation

User leaves the room if he has joined any. At a time user can join a single room. Users in the room get notification about the user leaving the room.

Syntax- /leave

* Exit from the server

If the user has not left the room before quitting then he is first left from the room.

Syntax- /quit

* Broadcast the message to all the users of the room.

Syntax- <message>

* Send a private message

After joining the room user can send a private message to another user from the same room.

If the provided another user is not present in the same room as sender then the sender gets an error message.

Syntax- <another username> : <message>

* Create a new chat room

User can create the new room.

The new federation can also be created easily, by passing a pass code which used as the code to join this new federation.

Syntax- /create <new room name>

Features

* **Private message**

Any user in the room can send a private message to another user in the same room.

* **Federation**

Before joining the federation, user authentication is provided. Also, communication is secured in federation.

* **High scalability**

As RoomBroadcaster handles the forwarding message to all the users in the same room, the Server can handle other requests at the same time. As number of client increases, performance increases because of this feature. Here RoomBroadcaster acts as a mini server, reducing the load of main server. The server creates a client handler for every request from the client. This is executed in a separate thread. This achieves high scalability.

* **Security**

Every message is encrypted before sending it to the user. Only federation communication can be made private.

* **Special emotions**

User can include the special emoticons in the broadcast and private messages.

Shortcomings

* As client is displaying the response from server to user and at the same time it is waiting for the input to be entered by the user, protocol accepting input from user on new line(=>) is not displayed properly in some cases. This occurs mostly when broadcast message is displayed at the client end. Client can still enter an input but (=>) prompt is not displayed sometimes.

Future Enhancements

* User creating a federation.
* User deleting a room.
* User deleting a federation.

The user who created a federation can only delete it if required.

* Making only communication in federation secure.

Instructions to execute the application

1. Copy the “ChatServer” folder to a location on the local disk.
2. Open the command prompts and navigate to the ChatServer folder. Now go to the “src” folder. All the java files are at this location.
3. In command prompt execute the compilation command as:

javac ChatServer.java Client.java ClientReader.java ClientWriter.java PublicRoom.java Room.java RoomBroadcaster.java User.java

1. Run the server by executing:

java ChatServer

1. To run the clients:
   1. Open other command prompt and go to the src folder in “ChatServer”.
   2. Run the client by executing: java Client localhost 9899
   3. To create more clients repeat steps ‘a’ and ‘b’