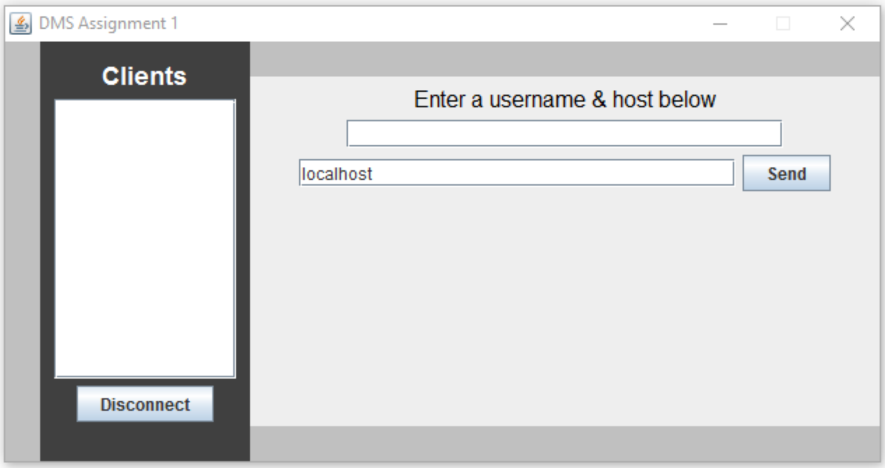
TCP/UDP Chat Application

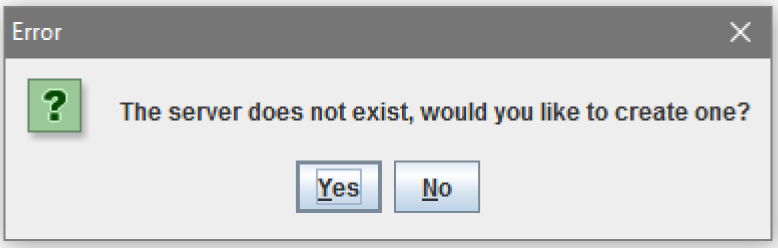
This piece of software is a client-server messaging system that uses TCP to send messages, and UDP to update a list of connected clients. Through GUI events the user can connect to or create a server, message individual clients, and broadcast messages to all clients.

User Instructions

1. Run the client application. Enter a username and host address in the respective text fields and press the 'Send' button.



   a. If the server exists, the chat client will be loaded.
   b. If the server does not exist, you will be given the option to start a server on your machine.



The server does not exist, would you like to create one?

2. When another client connects to the server, they will appear in the list on the left side of the client GUI.
3. Click on the list item to start or view a chat with a client. Type a message and press the 'Send' button.
4. To broadcast a message to all clients, select the "All clients" list option, type a message and press the 'Broadcast' button. This will be delivered to the corresponding private chats of all clients connected to the server.
5. Press the 'Disconnect' button to disconnect from the server and close the client application.

Network Protocols

TCP Protocol

- When the client application first runs, a socket connection is established in the Client class to connect to the server, and an ObjectOutputStream is created.
- The Server class accepts the socket connection and starts a thread to deal with all future communication with this socket.
- The client's username is then sent as a Message object to the server
- The server thread reads the username, and creates a key value pair in a HashMap where the key is the username, and the value is an ObjectOutputStream.
- The client starts a thread to receive messages, which iterates through a while loop waiting for objects to be received through the socket input stream.

UDP Protocol

- When the client has successfully connected, a thread is started to handle the updating of the clients list. It first creates a DatagramPacket and sends it over a DatagramSocket to the server.
- The server side also runs a thread, waiting to receive requests from the client through a DatagramSocket. The thread uses the received packet to send back a list of connected clients.
- When the client thread receives the packet, it updates the list in the GUI with the list it receives over the DatagramSocket.
- The client thread will sleep and then continue to loop through the above process, constantly updating the list.