Messaging Client App

For Distrubuted and Mobile Systems

Prepared by: Adam Campbell (1311607) and Aziel Shaw (14847095)

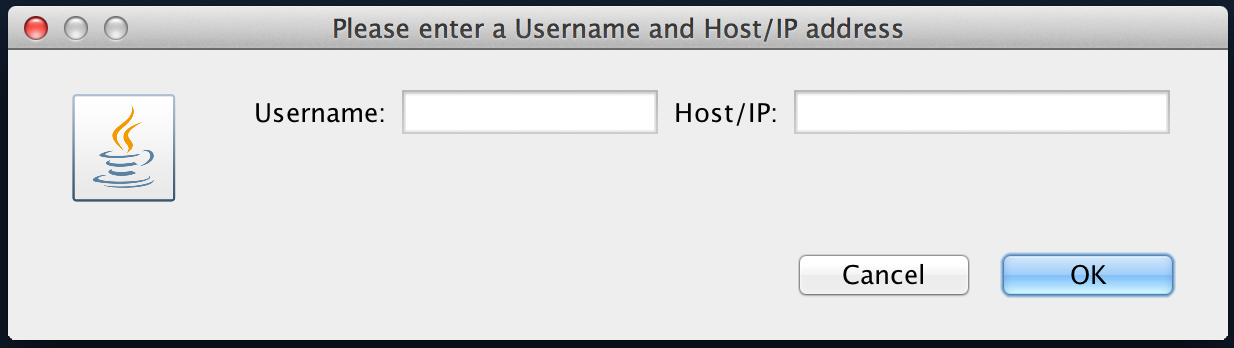
This project was prepared for Seth Hall for his Distributed and Mobile Systems paper. The goal of this Assignment was to create a client-server application using TCP and UDP to make an Instant Messaging system.

TCP is used for verifying connection to a server, verifying clients get their messages and verifies that the messages the client sends the server make it to the server.

UDP is used to contiously update clients about who is currently connected to the server and available to send messages to.

This Instant Messaging System is able to be used over multiple computers. The users are able to send messages to each other both globally and via a private message.

The Client starts off by establishing a TCP connection with the server. The server responds with a ConnectionResponse, which broadcasts, to all the other clients that are connected to the server via TCP. Once connection to the server is established the link between all clients is authorised. Clients can then send messages to each other by typing in the chat box and pressing either broadcast to send to all other users connected or by pressing the whisper button to send to another user privately. The client sends the messages to the server via a TCP connection. The server keeps track of all clients that are currently connected to the server and periodically sends out a list of who is currently connected to the server to all clients via a UDP broadcast. If the user wishes to disconnect from the chat room they can press the disconnect button. It sends a TCP DisconnectRequest to the server. Once received by the server it removes the user from the HashMap and sends a Disconnect Response to the client letting them know they have been successfully disconnected.



Enter the IP

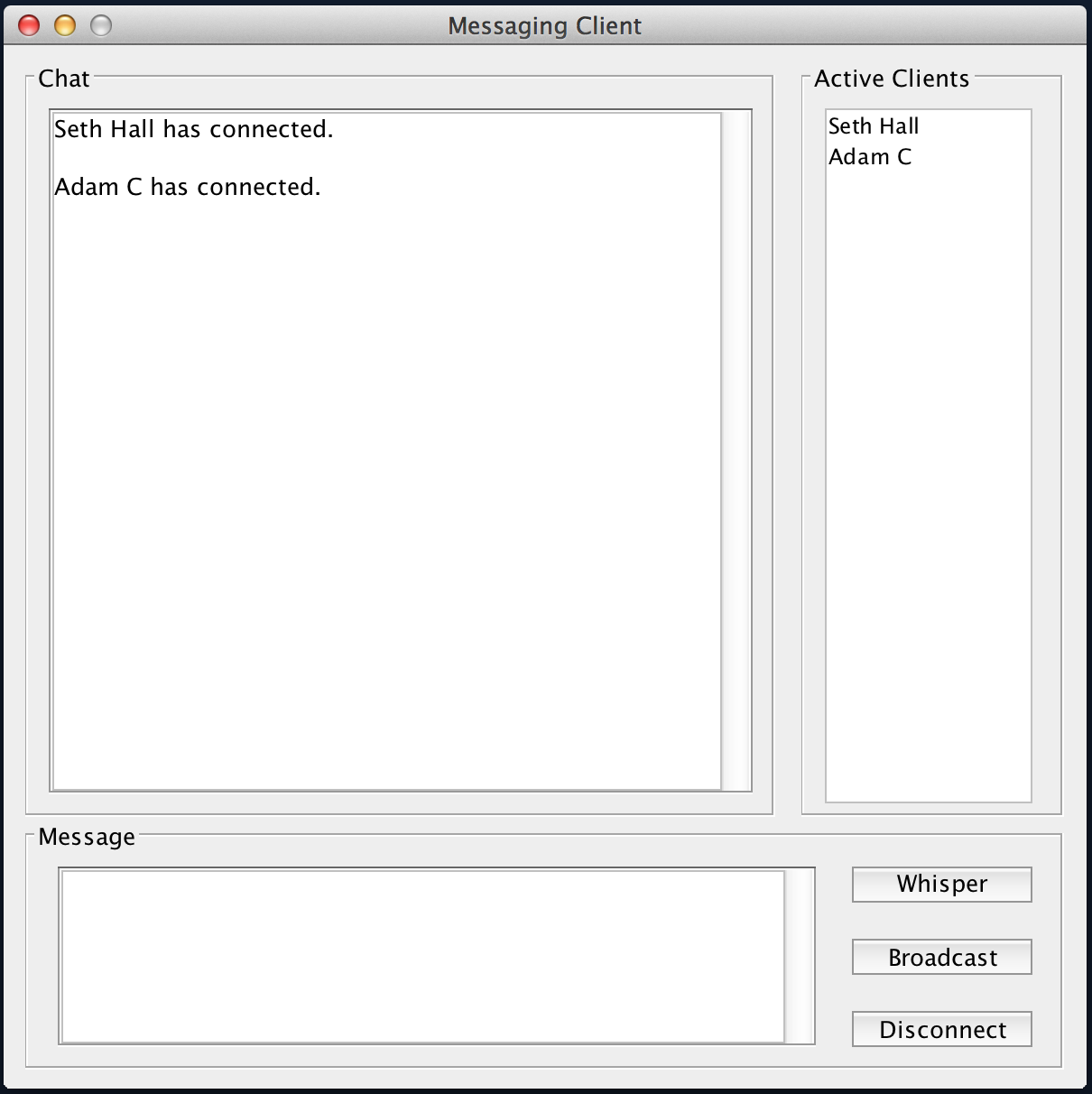
of the server

here.

Enter who you

want everyone else to be see

you as here.



Whisper to another

User on the server.

(Select their name

on the Active Client

list).

Broadcase your

message to all

users connected

to the server.

Disconnect

from the server.

Type the

message

you want to

send here.

A list of who

else is on the

server with

you.

Record of all

chat messages

you have

received.

# Classes

|  |  |
| --- | --- |
| Name | Description |
| ActiveClientsMessage | This class handles the current list of users who are connected to the server. |
| Client | This class handles the Client side information. Asks for a Username, and an IP to connect to. It has a GUI for the client to interact with and handles the sending and receiving of messages. |
| ClientData | This class is hosted on the server and belongs to a specific user currently connected to the server. And holds the message list of all the messages that have been sent to them via various whisper and broadcast messages. |
| ConnectionRequest | This class is for the Server to handle a user connecting from the server. Upon receiving this message the server will add the Client to the HashMap which contains the list of all the users on the server. |
| ConnectionResponse | This class is generated when a ConnectionRequest is received, whats generated is a specific response based on the users input. |
| DisconnectRequest | This class is for the Server to handle a user disconnecting from the server. Upon receiving this message the server will remove the Client from the HashMap containing the list of all the users on the server. |
| DisconnectResponse | This class is generated when a DisconnectRequest is received, whats generated is a specific response based on the users input. |
| Message | An abstract class that holds the blueprint for how messages between the server and client work. Extended by: ActiveClientsMessage, ConnectionRequest, ConnectionResponse, DisconnectRequest, DisconnectResponse, and Whisper. |
| Server | This class hosts the server. It is where all the clients connect to. The server handles all the clients messages list, and holds a HashMap that stores all the clients names and their inbox of messages. |
| Whisper | This class holds a recipient and and message to be delivered to a particular client on the server. It will be delivered to their inbox assuming they are currently connected to the server. |