John Dorman

S155252

Networking

Chris Janes

# IRC Protocol Definition Document

This document will specify a the application protocol definitions for a client to server chat system. The Chat system will have one server hosting multiple clients (users).

## 1.Client:

### 1.1 Client Id

Each client will need an unique Id which will be assigned to them as they connect to the server.

### 1.2 Client Username

Each client will need a unique username. After connecting to the server the client will be prompted to sign in or sign up running see *Client Command****, (2.1)***.

### 1.3 Client Permissions

Each client will either have a moderator or member status which will allow for access to different commands. As default all users are assigned into the member status.

## 2. Client Commands:

For a client to enter a command they will need to put a ‘\’ before their message.

### 2.1 User Sign up or Login

When a client connects to the server they will need to sign up or login to an account. A logon command will be sent to the server with the username and password as arguments. It then will be checked if username is stored and if they password is valid. If the username is not found a new user will be stored.If they username password pair is not a match to the stored data and invalid response will be given with a messaging detailing why.

**Client Sends**: Logon “*Username” “Password”*

**Server Responds:** *Valid | Invalid “msg”*

### 2.2 Help Command

When a client enters the “\help” it will print out a list of commands available to the client. The command will return query the server for the clients permissions. The server will respond with a the client's permission level which can then show a list of commands available to the client.

**Client Sends:** help

**Server Responds:** cmdList [*Client Permission*]

### 2.3 List Rooms

When a client enters the “\roomlist” command query the server for its list of rooms available. The server responds to the client with the list of rooms stored on the server.

**Client Sends:** roomlist

**Server Responds:** List of rooms

### 2.4 Join or Change Room

When a client enters the “\join” command they will need to pass in an additional “*roomName”* argument. The room name will be checked against the server’s room list to see if the request is valid. If so the server will connect the client to the room specified as the argument. If the room is not found then an invalid response will be given with a messaging detailing why.

**Client Sends:** join *“channelname”*

**Server Responds:** *Valid | Invalid “msg”*

### 2.5 Add or Remove Moderator

When a client enters the “\mod” command they will need to pass in two additional arguments “*Username”* and *“add/remove”*. The username will be checked by the server to see if it exists, and then the last argument will decide if that user is added or removed from the moderators. If the user cannot be found, if there are incorrect arguments, or if the arguments are incorrect the Invalid response will be given with a messaging detailing why. Else a Valid response will be given with a confirmation message.

**Client Sends:**mod “username” “add/remove”

**Server Responds:** *Valid | Invalid “msg”*

### 2.6 Kick User

When a client enters the “\kick” command they will need to pass in an additional “username” argument.The username will be checked by the server to see if it exists, and then the connection related with the user will be terminated. If the user is found and kicked the server will respond with a valid response and message, else an invalid response will be given with a messaging detailing why.

**Client Sends:** kick “username”

**Server Responds:** *Valid | Invalid “msg”*

### 2.7 Ban User

When a client enters the “\ban” command they will need to pass in an additional “*Username”*  argument. The server will search for the user and ban. The username will be checked by the server to see if it exists, then the connection related with the user will be terminated, and the connection information will be added to a blacklist. If the user is successfully found and added to the blacklist then the server will respond will a “Valid” response and a confirmation message, else an invalid response will be given with a messaging detailing why.

**Client Sends:** ban “username”

**Server Sends:** Valid | Invalid “msg”

### 2.8 Create rooms

When a client enters the “\room” command they will need to pass in an additional “roomname” argument. The room name will be checked against the server’s room list. If not found the server will create it and send a “Valid” response and a message, else if the room is found an invalid response will be given with a message detailing why.

**Client Sends:** room “roomName”

**Server Responds:** *Valid | Invalid “msg”*

### 2.9 Private Message

When a client enter the “\pm” command they will need to pass in two additional “username” and “message” arguments. The username will be checked by the server to see if it exists and if it currently connected. Then the message argument will be sent to them instead of being posted in the currently connect room. If user is found with a message argument a valid response will be given with a confirmation message, else an invalid response will be given with a message detailing why.

**Client Sends:** pm “*username” “message”*

**Server Responds:***Valid/Invalid “msg”*

## 3.Server:

### 3.1 Connections

The server will have a list of active connections and their associated username.

### 3.2 Users

The Server will have a list of users that contains a nickname, password, and permissions

### 3.3 Rooms

The server will have list of chat rooms.

## 4.Server Commands:

### 4.1 Backup

Every hour the server will schedule a backup for all of the user, room, message and blacklist data.

## 5. Messages

### 5.1 Chat Room Messages

A valid chat message that is sent from a client must be associated with a user, associated with a room, and must not start with a reserved character such as ‘\’ which is used for client commands. The server will receive this message and broadcast it out to all users connected to the room.

### 5.2 Error Messages

When a client command causes and invalid response from the server, the server will attach an error message to the response. A few common examples of these are “Err: UserNotFound”, “Err: RoomNotFound”, or “Err: CommandNotFound”.