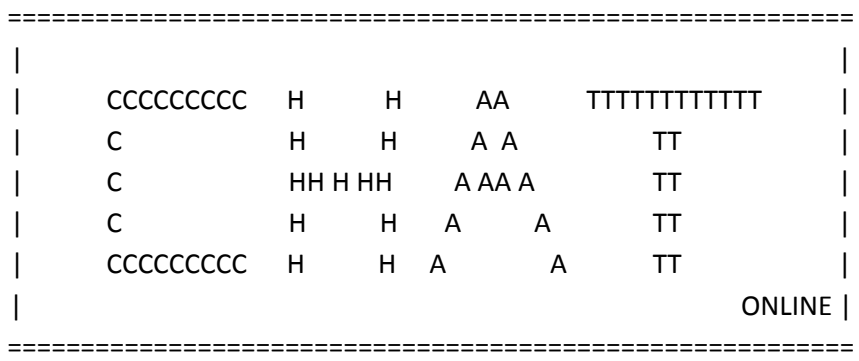
Chat Application by Adrian Salgado Lopez and Karina Pascual :D

Computer Networking Protocols | CS 4470 - 01 | Professor Senhua Yu



README

DESCRIPTION

This simple server and chat chat application was created by Adrian and Karina using python 3. To get started with our chat application you will need to run chat.py then client.py to have a running server and client to be able to use most of our application commands. See Functionality for more details. Overall our chat application functions like a Unix shell with several command options for clients.

PREREQUISITES

- Have python 3 installed
- The compressed chat.zip folder should contain the following files:
    - CHAT.PY - this files will runs the server side of the chat application
    - CLIENT.PY - this file will run the client side of the chat application
- To start up the server, it will take one command line parameter that includes a valid port number.
    - Ex:
        - python chat.py 80
- To start up the client, it will take one command line parameter that includes the port number of the server it is trying to connect to.
    - Ex:
        - python client.py 80
- Input ‘help’ command to get started.

CONTRIBUTIONS

Although we each contributed individually to this project, we approached it in an agile way, in a pair programming way. Where we both supported each other by sharing ideas while working on the same command, problem or bug. For example, when it came to sending messages there was a bug where the full message wasn't being sent out. How we approached this problem was how we approached developing the rest of the application, which is sharing 1 screen, sharing ideas, and developing our application.

I (Karina) first developed the foundation of the program for client, server, and functionality. I mostly focused on functionality. Initially, I made sure that a connection can be made from clients to the server. Then, focused on the following commands: myip, myport, help, list, send. I also shared/offered solutions to Adrian on functionality for "terminate", any bugs we encountered, and the logic of the program.

What I (Adrian) contributed to this application is to help Karina out whenever she got stuck, by sharing ideas to fix bugs when it came to creating the functionality of some of the commands she worked on. I created verifications for user inputs as well as making sure the application was handling errors appropiately. As well as create and finish up command terminate and exit. Also, I worked on creating the interface for the client side.

FUNCTIONALITY

The following list are commands that the program will accept to return information requested or action(s) executed:

- Help: Displays information about the available user interface options.
- Myip: Displays the ip of this process.
- Myport: Displays the port on which the process is listening for incoming connections.
- Terminate <connection id>: Terminates connection with <connection id>
- Send <connection id> <message>: Send <message> to connection with <connection id>
- Exit: Will close all connections and terminate the process.

SOURCES '-'y

---

https://www.geeksforgeeks.org/simple-chat-room-using-python/
https://www.geeksforgeeks.org/socket-programming-python/