BLG 433E
Computer Communications

Homework I
Python Socket Programming

Muhammed Burak Buğrul
150140015

Introduction

In this assignment, we are wanted to implement a chat room program with using python3.7 both for server and client.

Environment

Project implemented in an OSX environment and tested in two different OSX computers connecting each other on a local network. Project codes are written in Python 3.7.0

Requirements

Only requirement is Python 3.6+. Codes in this project uses only built-in functions and modules(socket, datetime, tkinter).

Project Architecture

server.py: Server script, it always waits for new clients to connect(up to a given limit) and handles sending messages to every client.

Client.py: Client script, connects a specific server given by user and receives/send messages to it.

Server Class Functions:

- **__init__:** Default constructor function.
- **serve:** Starts server to listen incoming clients.
- **wait_connections:** Waits for connections infinitely. When a connection made a new thread(with *communicate_client* function) assigned to that connection in order to receive its messages.
- **communicate_client:** In the beginning of the function server ask client for his/her username. It asks continuously until a valid username(not used by other clients) come. Then waits a client to send a message infinitely. When a message is received sends it to all clients. If the message is *quit* message, it deletes client from the client dictionary then terminates the thread.
- **broadcast:** Sends given message to all clients.

- **close:** Closes the server(terminates litening).

Client Class Functions:

- **__init__:** Default constructor function. It also starts the gui.
- **connect:** Connects client to server.
- **receive:** Infinitely waits for a message from the server(With help of a thread). When a message comes, shows it on the gui and adjusts the scrollbar.
- **Send:** Sends a message to server(to all clients). If message is the *quit* message, then closes the connection, closes the gui and terminates the thread.
- **close_callback:** Listens the gui close event(users quit even on the computer). If it occurs, callback functions send *the quit message* to server then terminates the connection, gui and thread.

Usage

First server.py should be run with command:

python3 server.py

After that, it asks for host and ports for server and connection limit, you can leave them empty for default settings. Then clients can run and connect to the server. Clients should be on the same local network unless server IP is static. Client run command is:

python3 client.py

After that client asks for host and port of the server, you can leave them for default settings. If client and the server are on the same machine you can leave the host part empty, otherwise you should enter the host address(IP). When connection to the server successfully occurs, server asks for username. It will ask it until you enter a valid(not used by other users) one. Now you can use the chat room. For exiting the chat room, you can type *the quit message* '---quit---'(without quotes) or simply click closing button of the gui.

Conclusion

It is a good practice for threads and sockets in python.