Client-Server Socket Program

M. Rafi Syafrinaldi 21/472772/PA/20335
Nashifa Ammara Fawziya Muchtar 21/472666/PA/20319
Almas Mirzandi Ramadhan 21/472900/PA/20357
Shreshta Adyaksa Hardono 21/478740/PA/20767

NOVEMBER 23, 2022

WHAT IS OUR PROGRAM?

A CHATTING PROGRAM WITH A TRIGGER TO PLAY A ROCK-PAPER-SCISSORS GAME!

HOW DOES IT WORK?

WE DEVELOPED A ROCK-PAPER-SCISSORS GAME SEPARATELY FROM THE CHATTING CLIENT, AND THAT CLIENT CAN JUST CALL THE RPS PROGRAM AS AN OPTION WITHIN IT.

CODE SNIPPETS

Keep in mind that these are just small bits of code from the program. The whole source code can be seen uploaded alongside this slide and later on during the demonstration.

Code Snippets

```python
def handshake(client):
    username = client.recv(1024).decode("utf-8")
    client.send(b"/sendGroupname")
    groupname = client.recv(1024).decode("utf-8")
    if groupname in groups:
        if username in groups[groupname].allMembers:
            groups[groupname].connect(username,client)
            client.send(b"/ready")
            print("User Connected:",username,"| Group:",groupname)
        else:
            groups[groupname].joinRequests.add(username)
            groups[groupname].waitClients[username] = client
            groups[groupname].sendMessage(username+" has requested to join the group.","PyconChat"
            client.send(b"/wait")
            print("Join Request:",username,"| Group:",groupname)
        threading.Thread(target=pyconChat, args=(client, username, groupname,)).start()
    else:
        groups[groupname] = Group(username,client)
        threading.Thread(target=pyconChat, args=(client, username, groupname,)).start()
        client.send(b"/adminReady")
        print("New Group:",groupname,"| Admin:",username)
```

```python
elif msg == "/disconnect":
    client.send(b"/disconnect")
    client.recv(1024).decode("utf-8")
    groups[groupname].disconnect(username)
    print("User Disconnected:",username,"| Group:",groupna
    break
elif msg == "/messageSend":
    client.send(b"/messageSend")
    message = client.recv(1024).decode("utf-8")
    groups[groupname].sendMessage(message,username)
elif msg == "/waitDisconnect":
    client.send(b"/waitDisconnect")
    del groups[groupname].waitClients[username]
    print("Waiting Client:",username,"Disconnected")
    break
```

These are some code screenshots for the multichat server. To the right is the code that notifies you of any changes within the group through the terminal. The handshake method is used to compile the client into one groupchat.

Code Snippets

```python
def main():
    if len(sys.argv) < 3:
        print("USAGE: python client.py <IP> <Port>")
        print("EXAMPLE: python client.py localhost 8000")
        return
    serverSocket = socket.socket(socket.AF_INET,socket.SOCK_STREAM)
    serverSocket.connect((sys.argv[1], int(sys.argv[2])))
    state["inputCondition"] = threading.Condition()
    state["sendMessageLock"] = threading.Lock()
    state["username"] = input("Welcome to PyconChat! Please enter your username: ")
    state["groupname"] = input("Please enter the name of the group: ")
    state["alive"] = False
    state["joinDisconnect"] = False
    state["inputMessage"] = True
    serverSocket.send(bytes(state["username"],"utf-8"))
    serverSocket.recv(1024)
    serverSocket.send(bytes(state["groupname"],"utf-8"))
    response = serverSocket.recv(1024).decode("utf-8")
```

```python
            data = pickle.loads(serverSocket.recv(1024))
            if data == set():
                print("No pending requests.*)
            else:
                print("Pending Requests:*)
                for element in data:
                    print(element)
        else:
            print(response)
    elif msg == "/approveRequest":
        serverSocket.send(bytes(".","utf-8"))
        response = serverSocket.recv(1024).decode("utf-8")
        if response == "/proceed":
            state["inputMessage"] = False
            print("Please enter the username to approve: *)
            with state["inputCondition"]:
                state["inputCondition"].wait()
            state["inputMessage"] = True
            serverSocket.send(bytes(state["userInput"],"utf-8"))
            print(serverSocket.recv(1024).decode("utf-8"))
        else:
            print(response)
    elif msg == "/disconnect":
        serverSocket.send(bytes(".","utf-8"))
        state["alive"] = False
        break
    elif msg == "/messageSend":
        serverSocket.send(bytes(state["userInput"],"utf-8"))
        state["sendMessageLock"].release()
    elif msg == "/allMembers":
        serverSocket.send(bytes(".","utf-8"))
        data = pickle.loads(serverSocket.recv(1024))
        print("All Group Members:*)
        for element in data:
            print(element)
    elif msg == "/onlineMembers":
        serverSocket.send(bytes(".","utf-8"))
        data = pickle.loads(serverSocket.recv(1024))
        print("Online Group Members:*)
        for element in data:
            print(element)
    elif msg == "/changeAdmin":
        serverSocket.send(bytes(".","utf-8"))
        response = serverSocket.recv(1024).decode("utf-8")
        if response == "/proceed":
            state["inputMessage"] = False
            print("Please enter the username of the new admin: *)
            with state["inputCondition"]:
                state["inputCondition"].wait()
            state["inputMessage"] = True
            serverSocket.send(bytes(state["userInput"],"utf-8"))
            print(serverSocket.recv(1024).decode("utf-8"))
        else:
            print(response)
    elif msg == "/whoAdmin":
```

These are some of the parts that make up the client. To the left is the main method that makes up the UI when we first start up the program. To the right, we can see the methods pertaining to its name (show all members, send message, etc.)

Code Snippets

```
elif msg == "/StartRPS":
    serverSocket.send(b"/StartRPS")
    serverSocket.recv(1024).decode("utf-8")
    cmd = 'cd /Users/rafisyafrinaldi/Desktop/multiclient && python gameServer.py'
    p1 = subprocess.run(cmd, shell=True)
    p1.returncode


elif msg == "/PlayRPS":
    serverSocket.send(b"/StartRPS")
    serverSocket.recv(1024).decode("utf-8")
    cmd2 = 'cd /Users/rafisyafrinaldi/Desktop/multiclient && python gameClient.py'
    p2 = subprocess.run(cmd2, shell=True)
    p2.returncode
```

This part of the code calls the RPS game.

Code Snippets

```
# MAIN GAME GUI
window_main = tk.Tk()
window_main.title("Game Client")
your_name = ""
opponent_name = ""
game_round = 0
game_timer = 4
your_choice = ""
opponent_choice = ""
TOTAL_NO_OF_ROUNDS = 5
your_score = 0
opponent_score = 0

# network client
client = None
HOST_ADDR = "192.168.0.76"
HOST_PORT = 8080
```

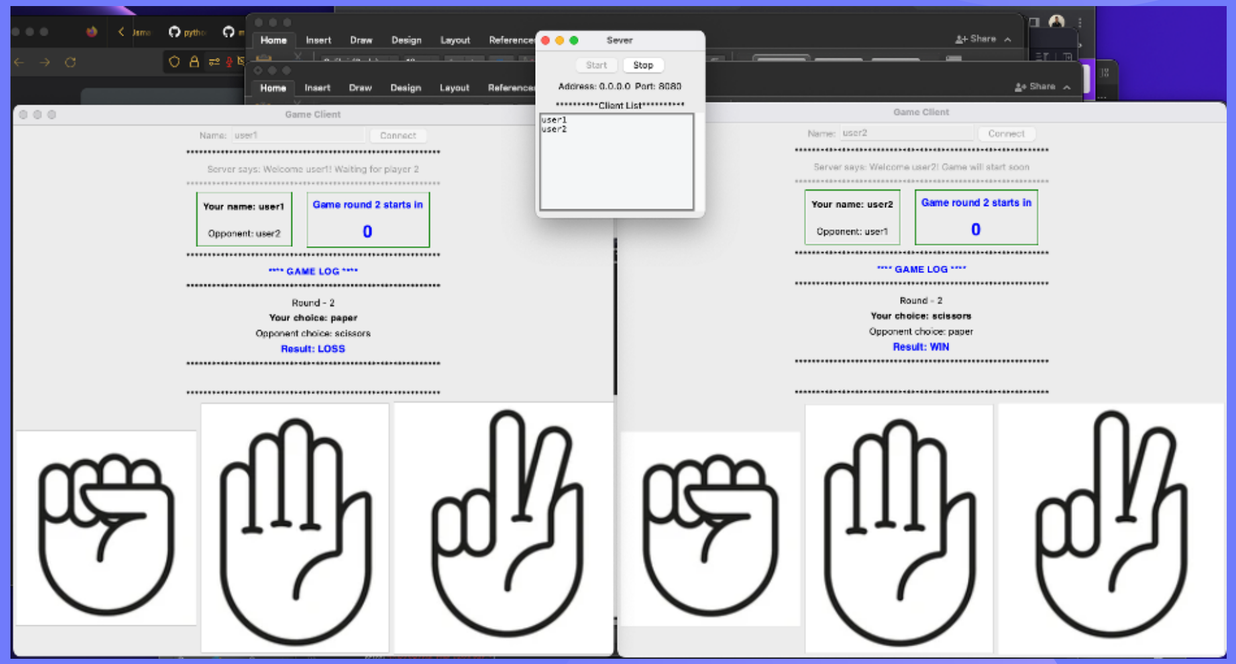
```
def game_logic(you, opponent):
    winner = ""
    rock = "rock"
    paper = "paper"
    scissors = "scissors"
    player0 = "you"
    player1 = "opponent"

    if you == opponent:
        winner = "draw"
    elif you == rock:
        if opponent == paper:
            winner = player1
        else:
            winner = player0
    elif you == scissors:
        if opponent == rock:
            winner = player1
        else:
            winner = player0
    elif you == paper:
        if opponent == scissors:
            winner = player1
        else:
            winner = player0
    return winner
```

```
    # we know two users are connected so game is ready to start
    threading._start_new_thread(count_down, (game_timer, ""))
    lbl_welcome.config(state=tk.DISABLED)
    lbl_line_server.config(state=tk.DISABLED)

elif from_server.startswith("$opponent_choice"):
    # get the opponent choice from the server
    opponent_choice = from_server.replace("$opponent_choice", "")

    # figure out who wins in this round
    who_wins = game_logic(your_choice, opponent_choice)
    round_result = " "
    if who_wins == "you":
        your_score = your_score + 1
        round_result = "WIN"
    elif who_wins == "opponent":
        opponent_score = opponent_score + 1
        round_result = "LOSS"
    else:
        round_result = "DRAW"
```

Some code that make up the RPS game. The logic is defined on code in the middle.

OUTPUT SCREENSHOTS

```
Welcome to PyconChat! Please enter your username: aksa
Please enter the name of the group: test
You have created the group test and are now an admin.
Available Commands:
/1 -> View Join Requests (Admins)
/2 -> Approve Join Requests (Admin)
/3 -> Disconnect
/4 -> View All Members
/5 -> View Online Group Members
/6 -> Transfer Adminship
/7 -> Check Group Admin
/8 -> Kick Member
/9 -> File Transfer
/10 -> Start Rock Paper Scissor Server
/11 -> Play Rock Paper Scissor
Type anything else to send a message
```



DEMO

REFERENCES

https://github.com/matchilling/RockPaperScissors.git

https://github.com/UsmanJafri/PyconChat

https://stackoverflow.com/questions/10672437/how-to-stop-multithread-chat-client\

THANKS!