COMP 416 COMP416: Computer Networks Network Programming Project

Client-Server Game Application via TCP Sockets, Çisem Özden-69707

ReadMe

Execution of the code:

Option 1: From an IDE of your choice, first run the HotColdServer project then run the HotColdClient project and follow the prompts written in the console.

Option 2: Open a terminal and go inside the HotColdServer project and run the following commands in which the HotColdServer.java file is located i.e., inside src.

- javac .\HotColdServer.java
- java HotColdServer

Open another terminal and do the same operations this time for the HotColdClient project and run the following commands. Then, follow the prompts written on the terminals.

- javac .\HotColdClient.java
- java HotColdClient

P.S. Server-side codes and output are bordered with blue, Client-side codes and output are bordered with red.

Simulation of the Game along with the Code

1. HotColdServer is started. The program executes until the line 22 and then waits for the user input. (Random x and y values are determined here, and the relevant variables are defined.)

```
7    public static void main(String[] args) throws IOException{
8        int port;
9        String player1;
10       String player2;
11       int player1Score = 0;
12       int player2Score = 0;
13
14       Random rand = new Random();
15       int x = rand.nextInt(256);
16       int y = rand.nextInt(256);
17
18       System.out.println("Enter welcoming socket's port");
19
20       try(BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in))){
21           String userInput;
22           userInput = stdIn.readLine();
```

```
Enter welcoming socket's port
10000
Waiting for client to connect...
```

2. After the port number is entered, Server creates the welcoming socket (line 26 in the number 4) and waits for the client to connect. Then, HotColdClient is started. The program executes until the line 16 and waits for the user input.

```
6    public static void main(String[] args) throws IOException{
7        int port;
8        String host = "localhost";
9        String player1;
10       String player2;
11
12       System.out.println("Enter server socket's port");
13
14       try(BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in))){
15           String userInput;
16           userInput = stdIn.readLine();
```

3. After the port number is entered, the client creates the socket and now the connection is established between the server and the client. The socket of the server is shown, the code executes until line 24, and it is the server's turn.

```
17              port = Integer.valueOf(userInput);
18
19              try(Socket echoSocket = new Socket(host, port);
20                      PrintWriter out = new PrintWriter (echoSocket.getOutputStream(), true);
21                      BufferedReader in = new BufferedReader (new InputStreamReader(echoSocket.getInputStream()));
22                      ){
23                  System.out.println("Server socket: " + echoSocket.getRemoteSocketAddress());
24                  player1 = in.readLine();
```

```
Enter server socket's port
10000
Server socket: localhost/127.0.0.1:10000
```

4. Also at the server side, socket of the client is shown, and the code executes until the end of line 33 and waits for the player 1 to enter her name.

```
23              port = Integer.valueOf(userInput);
24              System.out.println("Waiting for client to connect...");
25
26              try(ServerSocket serverSocket = new ServerSocket(port);
27                      Socket clientSocket = serverSocket.accept();
28                      PrintWriter out = new PrintWriter(clientSocket.getOutputStream(), true);
29                      BufferedReader in = new BufferedReader(new InputStreamReader(clientSocket.getInputStream()));
30                      ){
31                  System.out.println("Client socket: " + clientSocket.getRemoteSocketAddress());
32                  System.out.println("Player 1 please enter your name:");
33                  player1 = stdIn.readLine();
```

5. After the first player enters her name, the following output appears, and the server waits for the opponent's name that will be come from the client.

```
Client socket: /127.0.0.1:53141
Player 1 please enter your name:
Çisem
Waiting for player 2 name...
```

```
33                  player1 = stdIn.readLine();
34                  out.println(player1);
35                  System.out.println("Waiting for player 2 name...");
36                  player2 = in.readLine();
37                  System.out.println("You are playing with " + player2);
```

6. Once the second player enters her name, the Client waits for the player 1 to make her first guess, and the code executes until the line 32. Now, the game is controlled by the for loops in both the Server and the Client.

```
Player 2 you will be playing with Çisem, please enter your name:
Didem
Waiting for player 1 guess...
```

```
24                  player1 = in.readLine();
25                  System.out.println("Player 2 you will be playing with " + player1 + ", please enter your name:");
26                  player2 = stdIn.readLine();
27                  out.println(player2);
28
29                  for(int i=0; i<3; i++) {
30                      System.out.println("Waiting for player 1 guess...");
31
32                      String messagePrompt = in.readLine();
```

7. Game management in the Server and the message exchange between the Server and the Client can be seen as follows (Euclidean function calculates the distance between the actual number and the guesses):

```
                for(int i=0; i<3; i++) {
                    System.out.println(player1 + " please enter your x and y guesses, comma separated.");

                    String guess1 = stdIn.readLine();
                    int x1 = Integer.valueOf(guess1.split(",")[0]);
                    int y1 = Integer.valueOf(guess1.split(",")[1]);

                    System.out.println("Waiting for player 2 guess...");
                    out.println(player2 + " please enter your x and y guesses, comma separated.");

                    String guess2 = in.readLine();

                    int x2 = Integer.valueOf(guess2.split(",")[0]);
                    int y2 = Integer.valueOf(guess2.split(",")[1]);

                    if(euclidean(x1, x, y1, y) < euclidean(x2, x, y2, y)) {
                        player1Score++;
                        System.out.println("Winner for round " + (i+1) + " is " + player1);
                        out.println("Winner for round " + (i+1) + " is " + player1);
                    }else if (euclidean(x1, x, y1, y) > euclidean(x2, x, y2, y)){
```

```
                        player2Score++;
                        System.out.println("Winner for round " + (i+1) + " is " + player2);
                        out.println("Winner for round " + (i+1) + " is " + player2);
                    }else {
                        player1Score++;
                        player2Score++;
                        System.out.println("Winner for round " + (i+1) + " is Both players");
                        out.println("Winner for round " + (i+1) + " is Both players");
                    }
                }

                if(player1Score>player2Score) {
                    System.out.println("Game winner is " + player1);
                    out.println("Game winner is " + player1);
                }else if(player1Score<player2Score) {
                    System.out.println("Game winner is " + player2);
                    out.println("Game winner is " + player2);
                }else {
                    System.out.println("Game winner is Both players");
                    out.println("Game winner is Both players");
                }
            }
        }
    }

    public static double euclidean(int x1, int x2, int y1, int y2) {
        return Math.sqrt(Math.pow(x1-x2, 2) + Math.pow(y1-y2, 2));
    }
```

```
                for(int i=0; i<3; i++) {
                    System.out.println("Waiting for player 1 guess...");

                    String messagePrompt = in.readLine();
                    System.out.println(messagePrompt);

                    String guess2 = stdIn.readLine();
                    out.println(guess2);

                    String messageWinner = in.readLine();
                    System.out.println(messageWinner);
                }
                System.out.println(in.readLine());
```

8. Sample output from both sides (Server on the left, Client on the right):

```
Enter welcoming socket's port
10000
Waiting for client to connect...
Client socket: /127.0.0.1:58588
Player 1 please enter your name:
Çisem
Waiting for player 2 name...
You are playing with Meryem
Çisem please enter your x and y guesses, comma separated.
150,200
Waiting for player 2 guess...
Winner for round 1 is Meryem
Çisem please enter your x and y guesses, comma separated.
170,160
Waiting for player 2 guess...
Winner for round 2 is Meryem
Çisem please enter your x and y guesses, comma separated.
175,170
Waiting for player 2 guess...
Winner for round 3 is Çisem
Game winner is Meryem
```

```
Enter server socket's port
10000
Server socket: localhost/127.0.0.1:10000
Player 2 you will be playing with Çisem, please enter your name:
Meryem
Waiting for player 1 guess...
Meryem please enter your x and y guesses, comma separated.
180,180
Winner for round 1 is Meryem
Waiting for player 1 guess...
Meryem please enter your x and y guesses, comma separated.
180,150
Winner for round 2 is Meryem
Waiting for player 1 guess...
Meryem please enter your x and y guesses, comma separated.
160,170
Winner for round 3 is Çisem
Game winner is Meryem
```