Comp 416 PA-1 Report

Ali Taylan Akyürek
64229

ReadMe:

This code provides communication among 2 programs in same computer. First, server class should be executed. After execution, server creates a port number indicated via command line arguements. Then the server prints the port number. Then client binds that port via commandd line arguements. After that, client asks input from the user and transmit that to server, server prints client's message and asks input from the user. Than client prints that message and procedure starts again until server or client writes quit as message. There is also a timer that measures the time elapsed and after 20 seconds, program terminates and prints message "Timeout".

Communication

Server:

```
        ServerSocket serverSocket= new ServerSocket(port);
                                Socket clientSocket= serverSocket.accept();
                                PrintWriter out= new PrintWriter(clientSocket.getOutputStream(), true);
                                BufferedReader in = new BufferedReader(new
InputStreamReader((clientSocket.getInputStream())));
                            BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in));
```

With these lines, server opens the welcoming socket and creates input output streams. To be more specific, server asks user input with "strReader", "strReader" transforms bytes to string, then with "out.println", it writes the message to client's "in" stream.

Client:

```
                            Socket communicationSocket = new Socket(host, port);
                            PrintWriter out = new PrintWriter(communicationSocket.getOutputStream(), true);
                            BufferedReader in = new BufferedReader(new
InputStreamReader(communicationSocket.getInputStream()));
                                BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in));
```

With these lines, client binds to socket and creates input output streams. . To be more specific, client asks user input with "strReader", "strReader" transforms bytes to string, then with "out.println", it writes the message to server's "in" stream.

Now communication has builded. After that, i will explain the procedure step by step via writing description near the lines in code snippets.

Client:

```
System.out.println("Type some text");
String userInput;
userInput=strReader.readLine(); This line asks user input and transform
bytes to string
out.println(userInput);        Writes the input to output stream, this
message is now in server's "in" stream.
if(userInput.equals("quit"))   This terminates the program if the message
is quit
{
        communicationSocket.close();

        break;
}
```

Server:

```
inputLine = in.readLine();    This line writes the input from "in" stream
to inputLine.
if(inputLine.equals(null)) {  This terminates the program if inputLine is
null
        serverSocket.close();
        break;
}
if(inputLine.equals("quit")) This terminates the program if the message is
quit
{
        serverSocket.close();
        break;
}
System.out.println("client says: " + inputLine); prints client's message
```

Client:

```
String inputLine= in.readLine(); This line writes the input from "in"
stream to inputLine.
if(inputLine.equals(null)) {  This terminates the program if inputLine is
null
        communicationSocket.close();
        break;
}
if(inputLine.equals("quit")). This terminates the program if the message
is quit

{
        communicationSocket.close();

        break;
}
```

```
System.out.println("server says: "+inputLine); prints server's message
```

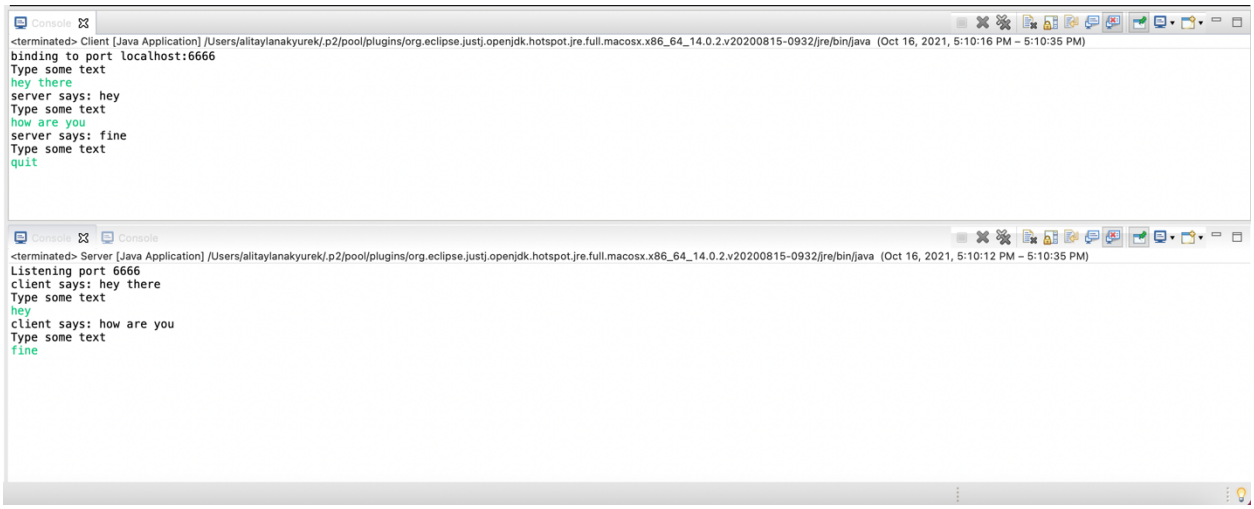
Server:

```
System.out.println("Type some text");
String userInput;
userInput=strReader.readLine(); This line asks user input and transform
bytes to string

out.println(userInput); Writes the input to output stream, this message is
now in client's "in" stream.

if(userInput.equals("quit")) This terminates the program if the message is
quit

{
        serverSocket.close();
        break;
}
```



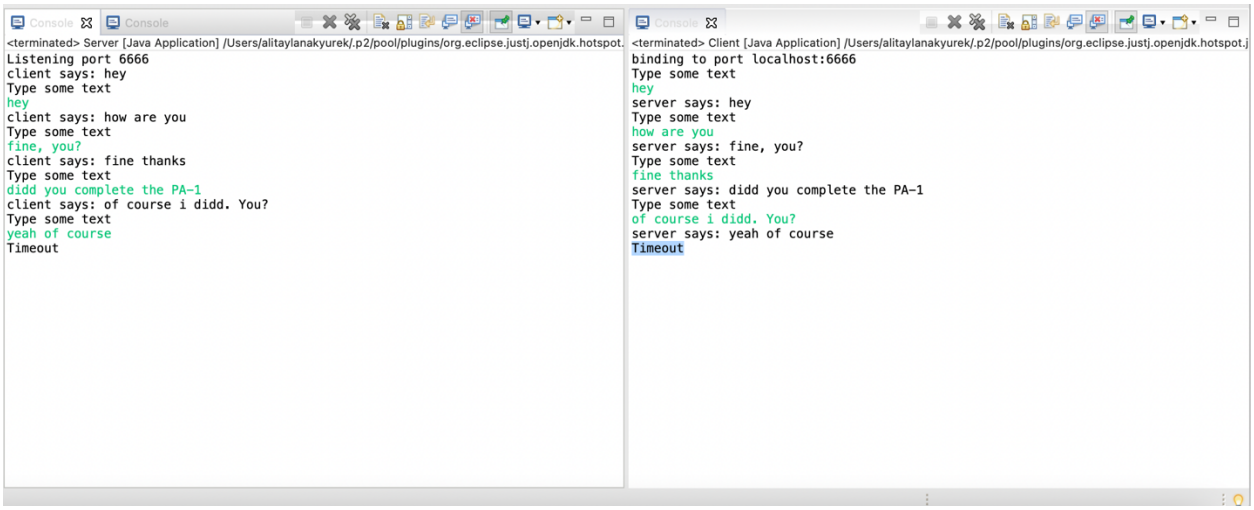
Timeout:

```
long start = System.currentTimeMillis(); this starts the time

timeOut=(System.currentTimeMillis()-start)/1000; this measures the elapsed time

if(timeOut>20)                          this terminates the program after 20 seconds
        {
            System.out.println("Timeout");

                break;
        }
```


Console
Console
<terminated> Server [Java Application] /Users/alitaylanakyurek/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.
Listening port 6666
client says: hey
Type some text
hey
client says: how are you
Type some text
fine, you?
client says: fine thanks
Type some text
didd you complete the PA-1
client says: of course i didd. You?
Type some text
yeah of course
Timeout
Console
<terminated> Client [Java Application] /Users/alitaylanakyurek/.p2/pool/plugins/org.eclipse.justj.openjdk.hotspot.j
binding to port localhost:6666
Type some text
hey
server says: hey
Type some text
how are you
server says: fine, you?
Type some text
fine thanks
server says: didd you complete the PA-1
Type some text
of course i didd. You?
server says: yeah of course
Timeout