

# CL-IX Assignment 1A

Roll No. 43104  
Ajinkya Tikhe

## Assignment 1A - TCP Java Sockets

**Aim:** To develop any distributed application through implementing client-server communication programs based on TCP Java Sockets.

**Problem Statment:** String Manipulation: Design a distributed application that consists of client-server communication using TCP, UDP & RMI techniques in Java. Multiple clients can simultaneously connect to the server and multiple clients submit two strings to the server and the server returns the concatenation of the given strings.

File 1:

### Server.java

// A Java program for a Server

```
import java.net.*;
```

```
import java.io.*;
```

```
public class Server
```

```
{
```

```
    //initialize socket and input stream
```

```
    private Socket          socket = null;
```

```
    private ServerSocket server = null;
```

```
    private DataInputStream in      = null;
```

```
    // constructor with port
```

```
    public Server(int port)
```

```
    {
```

```
        // starts server and waits for a connection
```

```
        try
```

```
        {
```

```

server = new ServerSocket(port);
System.out.println("Server started");

System.out.println("Waiting for a client ...");

socket = server.accept();
System.out.println("Client accepted");

// takes input from the client socket
in = new DataInputStream(
    new BufferedInputStream(socket.getInputStream()));

String line = "";
String line1 = "";

for(int n=0; n<2; n++)
{
    try
    {
        line1 = in.readUTF();
        line = line+line1;

    }
    catch(IOException i)
    {
        System.out.println(i);
    }
}

System.out.println("Concatenation of the given strings : " + line);
System.out.println("Closing connection");

// close connection
socket.close();
in.close();
}
catch(IOException i)
{
    System.out.println(i);
}

```

```

    }
}

public static void main(String args[])
{
    Server server = new Server(5000);
}
}

```

---

File 2:

### **Client.java**

// A Java program for a Client

```
import java.net.*;
```

```
import java.io.*;
```

```
public class Client
```

```
{
```

```
    // initialize socket and input output streams
```

```
    private Socket socket          = null;
```

```
    private DataInputStream input = null;
```

```
    private DataOutputStream out   = null;
```

```
    // constructor to put ip address and port
```

```
    public Client(String address, int port)
```

```
    {
```

```
        // establish a connection
```

```
        try
```

```
        {
```

```
            socket = new Socket(address, port);
```

```
            System.out.println("Connected");
```

```
            // takes input from terminal
```

```
            input = new DataInputStream(System.in);
```

```

        // sends output to the socket
        out = new DataOutputStream(socket.getOutputStream());
    }
    catch(UnknownHostException u)
    {
        System.out.println(u);
    }
    catch(IOException i)
    {
        System.out.println(i);
    }

```

```

// string to read message from input
String line = "";

```

```

System.out.println("Enter the input - ");
for(int n=0; n<2; n++)
    {
        try
        {
            System.out.print("String " + (n+1) + " : ");
            line = input.readLine();
            out.writeUTF(line);
        }
        catch(IOException i)
        {
            System.out.println(i);
        }
    }

```

```

// close the connection
try
{
    input.close();
    out.close();
    socket.close();
}
catch(IOException i)
{

```

```

        System.out.println(i);
    }
}

public static void main(String args[])
{
    Client client = new Client("127.0.0.1", 5000);
}
}

```

---

## Client Console:

The screenshot shows the Eclipse IDE interface. The Project Explorer on the left lists various projects, including 'TCP Java Sockets'. The main editor displays the 'Client.java' file with the following code:

```

20  System.out.println(" Connected ");
21
22  // takes input from terminal
23  input = new DataInputStream(System.in);
24
25  // sends output to the socket
26  out = new DataOutputStream(socket.getOutputStream());
27  }
28  catch(UnknownHostException u)
29  {
30      System.out.println(u);
31  }
32  catch(IOException i)
33  {
34      System.out.println(i);
35  }
36
37  // string to read message from input
38  String line = "";
39
40  System.out.println("Enter the input - ");
41  for(int n=0; n<2; n++)
42  {
43      try
44      {

```

The Console window at the bottom shows the following output:

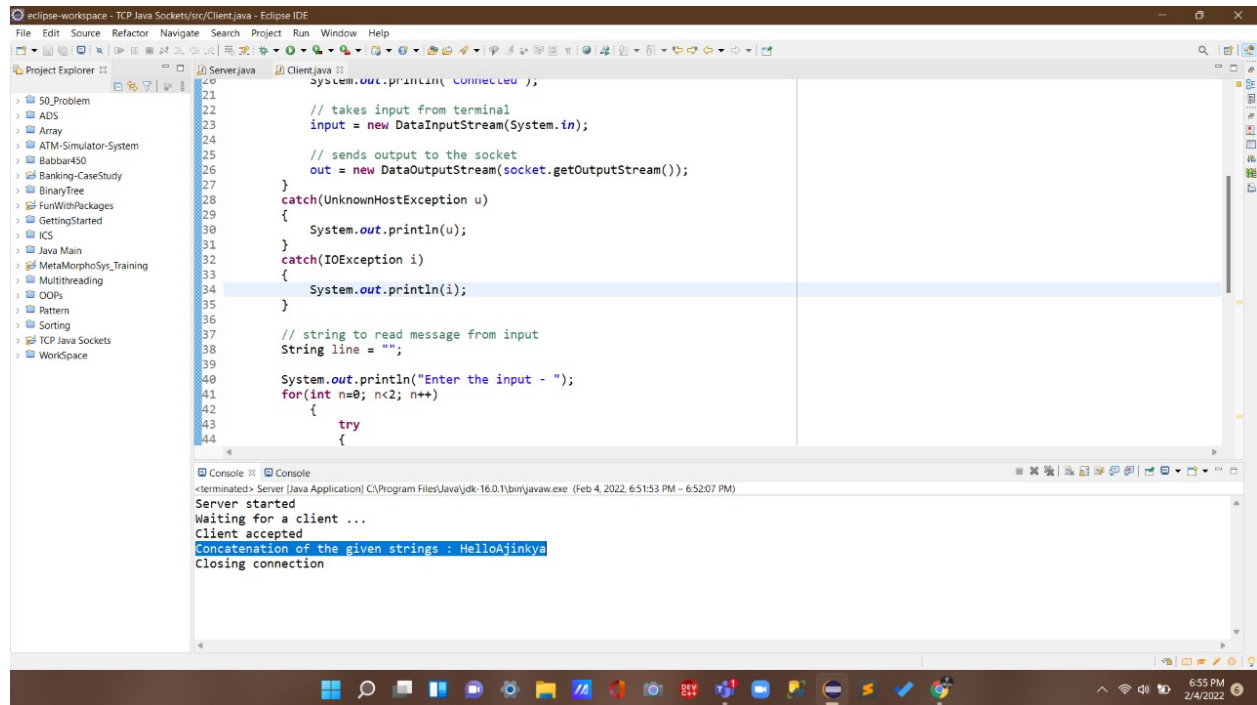
```

<terminated> client (1) [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (Feb 4, 2022, 6:51:57 PM - 6:52:07 PM)
Connected
Enter the input -
String 1 : Hello
String 2 : Ajinkya

```

The Windows taskbar at the bottom shows the time as 6:56 PM on 2/4/2022.

## Server Console:



```
21      System.out.println(" Connected ");
22
23      // takes input from terminal
24      input = new DataInputStream(System.in);
25
26      // sends output to the socket
27      out = new DataOutputStream(socket.getOutputStream());
28  }
29  catch(UnknownHostException u)
30  {
31      System.out.println(u);
32  }
33  catch(IOException i)
34  {
35      System.out.println(i);
36  }
37
38  // string to read message from input
39  String line = "";
40
41  System.out.println("Enter the input - ");
42  for(int n=0; n<2; n++)
43  {
44      try
45      {
```

Console

<terminated> Server [Java Application] C:\Program Files\Java\jdk-16.0.1\bin\javaw.exe (Feb 4, 2022, 6:51:53 PM - 6:52:07 PM)

Server started  
Waiting for a client ...  
Client accepted  
Concatenation of the given strings : HelloAjinkya  
Closing connection

---

**Conclusion:** Developed distributed application through implementing client-server communication programs based on TCP Java Sockets.

---