Lab Report 1

Course Title : Sessional Based on CSE 3205
Course No. : CSE 3206

Submitted by

Name: Md. Shabir Khan Akash.

Class: 3rd year, Even semester.

Department : CSE

Roll: 1603108

Section: B

Submitted to

Dr. A. H. M. Sarowar Sattar

Professor

Department of CSE, RUET

Objectives:

To gather information about “Socket” and “Socket Programming” & its appications and also creating one way and two way server-client socket program.

Description :

Q. What is Socket?

Answer: Basically a socket is an internal endpoint of a two way communication link between two programs running on the same network. A socket is attached to a port number so that the TCP (Transmission Control Protocol) layer can identify that very application that the data is destined to be sent to. Sockets are bidirectional, meaning that either side of the connection is capable of both sending and receiving data.Usually an endpoint is a combination of IP (Internet Protocol) address and port number. Point to be noted that a socket is a virtual thing, not a hardware. Sockets are low-level connections. The client and the server both communicate through a stream of bytes written to the socket. The client and the server must agree on a protocol--that is, they must agree on the language of the information transferred back and forth through the socket.

Q. What is Socket Programming ?

Answer: Socket programming is a way of connecting two nodes on a network to communicate with each other. One socket (node) listens on a particular port at an IP, while other socket reaches out to the other to form a connection. Server forms the listener socket while client reaches out to the server. Socket programs are generally used to communicate between various processes usually running on different systems. It is mostly used to create a client-server environment. Socket programming can be done in many popular programming languages like C, C++, Python, Java. These are some primary socket API functions and methods given below:

1. socket() ,

2. bind(),

3. listen(),

4. accept(),

5. connect(),

6. connect_ex(),

7. send(),

8. recv(),

9. close()

Q. What are the applications of Socket Programming ?

Answer: The applications of socket programming are listed below :

- for creating servers like web servers, mail servers, ftp servers and other servers,
- For establishing communication between two processes,
- mostly used to create a client-server environment.
- for creating clients like browsers.
- for transfering file or message form one computer to another computer.
- for creating any sever, client applications. Like FTP, HTTP, DNS etc.

One way Server-Client program:

Step 1 : A java file was created named **MyServer.java** which was a server program.

Step 2: In the main function under public class **MyServer**, ServerSocket, Socket objects were declared to establish or initialize a serversocket and socket.

Step 3: Then an **InputStream, InputStreamReader** were declared to get the input that will be coming from the client program.

Step 4: Then a **BufferedReader** was declared to take the string data as a buffer stream in the input. A String variable **returnMessage** was declared to store message.

Step 5: Then **OutputStream, OutputStreamWriter, BufferedWriter** were used to show some messages on server side. Finally, **flush()** method was used on the **BufferedWriter** object to flush the stream.

Step 6: Then A java file was created named **MyClient.java** which was a client program.

Step 6: In the main function under public class **MyClientClass**, Socket object was declared to establish or initialize a socket.

Step 7: Then an **OutputStream, OutputStreamReader** were declared to send the input that will be passed to the server program.

Step 8: Then a **BufferedReader** was declared to take the string data as a buffer stream in the input. A String variable **number** was declared to take user input in the client program and write() method was used on the **BufferedReader** object.

Step 9: Finally, **flush()** method was used on the **BufferedWriter** object to flush the stream.

Two way Server-Client program:

Step 1 : A java file was created named **MyServer.java** which was a server program.

Step 2: In the main function under public class **MyServer**, ServerSocket, Socket objects were declared to establish or initialize a serversocket and socket.

Step 3: Then an **DataInputStream, DataOutputStream** were declared to get the input and output that will be coming from the client program and will be sent to client respectively.

Step 4: Then a **BufferedReader** was declared to take the string data as a buffer stream in the input. Two String variables str1, str2 were declared to store message.

Step 5: Then unless user types "exit", the server will take input from user using **readLine()** method and show input from client using **readUTF()** method. Finally, **flush()** method was used on the **DataOutputStream** object to flush the stream and **ServerSocket**, **Socket**, **DataInputStream** will be closed using **close()** method.

Step 6: Then A java file was created named **MyClient.java** which was a client program.

Step 7: In the main function under public class **MyClientClass**, Socket object was declared to establish or initialize a socket.

Step 8: Then an **DataInputStream, DataOutputStream** were declared to get the input and output that will be coming from the server program and will be sent to server respectively.

Step 9: Then a **BufferedReader** was declared to take the string data as a buffer stream in the input. Two String variables str1, str2 were declared to store message.

Step 10: Then unless user types "exit", the client will take input from user using **readLine()** method and show input from server using **readUTF()** method. Finally, **flush()** method was used on the **DataOutputStream** object to flush the stream and **Socket** will be closed using **close()** method.

Conclusion:

In the first portion, the definition of Socket, Socket Programming and the applications of socket programming were described briefly. Then, In the one way server-client program procedure, an input was sent from client to server program and to check the connectivity of one way server client socket program. As it only sends from client to server that's why it is a one way server-client program. Lastly In the two way server-client program, an input was sent from client to server program along with from server to client and to check the connectivity of two way server client socket program. As it both sends from client to server and server to client that's why it is a two way server-client program.

Appendix:

One way Server-client program:

MyServer.java

```
import java.io.*;
import java.net.*;
public class MyServer{
        private static Socket socket;
        public static void main(String[] args)
        {
          try
          {
            int port = 25000;
            ServerSocket serverSocket = new ServerSocket(port);

            while(true)
            {
              socket = serverSocket.accept();
              System.out.println("Client has connected!");
              InputStream is = socket.getInputStream();
              InputStreamReader isr = new InputStreamReader(is);
              BufferedReader br = new BufferedReader(isr);
              String number = br.readLine();
              System.out.println("Message received from client is "+number);
              String returnMessage;
              try
              {
                int numberInIntFormat = Integer.parseInt(number);
                int returnValue = numberInIntFormat%1000;
                returnMessage = String.valueOf(returnValue) + "\n";
              }
              catch(NumberFormatException e)
              {
                returnMessage = "Please send your student ID\n";
              }

              OutputStream os = socket.getOutputStream();
              OutputStreamWriter osw = new OutputStreamWriter(os);
              BufferedWriter bw = new BufferedWriter(osw);
              bw.write(returnMessage);
              System.out.println("Message sent to the client is "+returnMessage);
              bw.flush();
            }
          }
          catch (Exception e)
          {
            e.printStackTrace();
          }
        }
}
```

MyClient.java

```
import java.io.*;
import java.net.*;
import java.util.Scanner;

public class MyClient{

        private static Socket socket;

        public static void main(String args[])
        {
                int flag = 0;
          Scanner input=new Scanner(System.in);
          while(true)
          {
            try
            {

                if(flag == 0){
                        System.out.println("Enter any text to test the connectivity of Server and Client: ");
                        flag = flag + 1;
                }

              String host = "localhost";
              int port = 25000;
              InetAddress address = InetAddress.getByName(host);
              socket = new Socket(address, port);
              System.out.println("You're now connected to the Server");

              OutputStream os = socket.getOutputStream();
              OutputStreamWriter osw = new OutputStreamWriter(os);
              BufferedWriter bw = new BufferedWriter(osw);
              String number;
              number=input.next();
              String sendMessage = number + "\n";
              bw.write(sendMessage);
              bw.flush();
              System.out.println("Message sent to the server : "+sendMessage);
              InputStream is = socket.getInputStream();
              InputStreamReader isr = new InputStreamReader(is);
              BufferedReader br = new BufferedReader(isr);
              String message = br.readLine();
              System.out.println("Message received from the server : " +message);
            }
            catch (IOException exception)
            {
              System.out.println("Server is still offline");

            }
          }
        }
}
```

Two way Server-client program:

MyServer.java

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

public class MyServer{
	public static void main(String args[])throws Exception{
		ServerSocket serversoc = new ServerSocket(3333);
		Socket soc = serversoc.accept();
		DataInputStream din = new DataInputStream(soc.getInputStream());
		DataOutputStream dout = new DataOutputStream(soc.getOutputStream());
		BufferedReader buffRead = new BufferedReader(new InputStreamReader(System.in));
		String str1 = "", str2 = "";
		System.out.println("~~~ (Enter 'exit' for exiting) ~~~");
		while(!str1.equals("exit")){
			str1 = din.readUTF();
			System.out.println("client is saying : "+str1);
			System.out.println("\nEnter a message that will be sent to client: ");
			str2 = buffRead.readLine();
			dout.writeUTF(str2);
			dout.flush();
		}
		din.close();
		soc.close();
		serversoc.close();
	}
}
```

MyClient.java

```
import java.net.*;
import java.io.*;
public class MyClient{
	public static void main(String args[])throws Exception {
		Socket soc = new Socket("localhost",3333);
		DataInputStream din = new DataInputStream(soc.getInputStream());
		DataOutputStream dout = new DataOutputStream(soc.getOutputStream());
		BufferedReader buffRead = new BufferedReader(new InputStreamReader(System.in));
		String str1 = "", str2 = "";
		System.out.println("~~~ (Enter 'exit' for exiting) ~~~");
		while(!str1.equals("exit")){
			System.out.println("\nEnter a message that will be sent to server: ");
			str1 = buffRead.readLine();
			dout.writeUTF(str1);
			dout.flush();
			str2 = din.readUTF();
			System.out.println("Server is saying : "+str2);
		}
		dout.close();
		soc.close();
	}
}
```