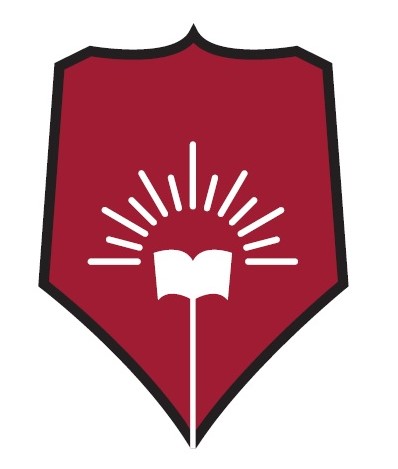
****

**Experiment No.: 5**

**Implement TCP or UDP Sockets**

1. **Aim:**Write a Programme to implement Socket Programming using TCP or UDP.
2. **Objectives:** To understand working of the transport layer protocol**.**
3. **Outcomes:** The learner will be able to

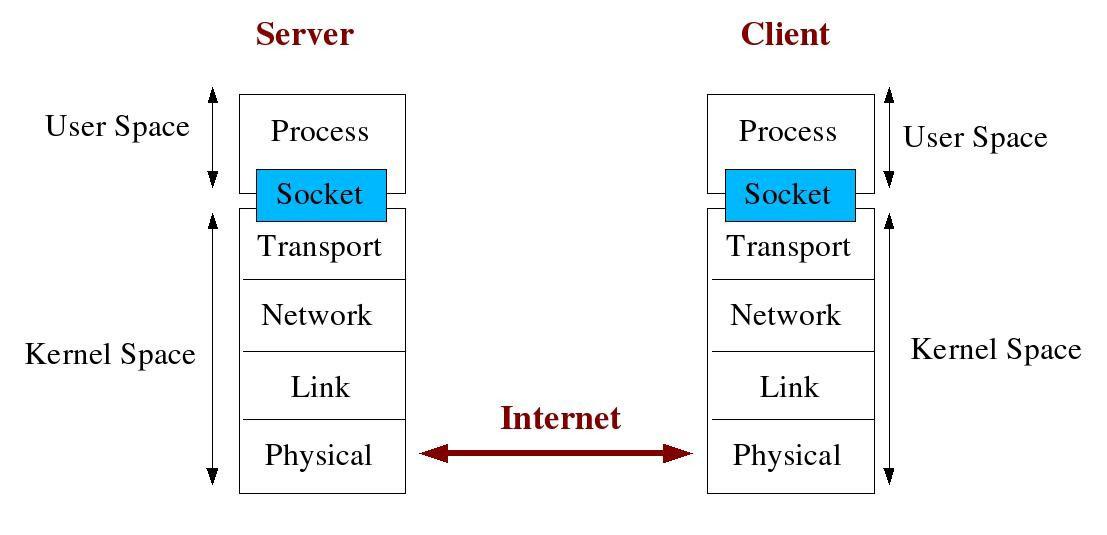
* Demostrate the transport layer with the help of socket programming.
* Recognize the need for sockets in life-long learning.

1. **Hardware/Software required:** JDK 1.8, Python 3.6
2. **Theory:**

What is a socket?

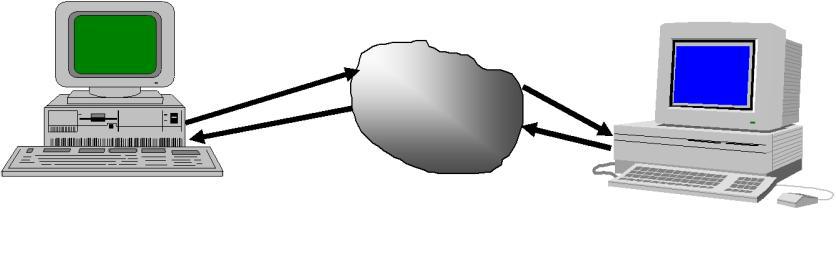
● Socket: An interface between an application process and transport layer. The application process can send/receive messages to/from another application process (local

or remote) via a socket.



**Client/Server Communication**

At a basic level, network-based systems consist of a server, client, and a media for communication as shown in Figure 1. A computer running a program that makes request for services is called client machine. A computer running a program that offers requested services from one or more clients is called server machine. The media for communication can be wired or wireless network.



In computer network system the communication is takes place only through themessage passing. Usually the communication takes place among client machine and server machine. The socket programming is used to communicate between client and server.

Here, we are going to make one-way client and server communication. In this application, client sends a message to the server, server reads the message and prints it. Here, two classes are being used: Socket and ServerSocket.

The Socket class is used to communicate client and server. Through this class, we can read and write message. The ServerSocket class is used at server-side. The accept() method of ServerSocket class blocks the console until the client is connected. After the successful connection of client, it returns the instance of Socket at server-side.

#Socket class

A socket is simply an endpoint for communications between the machines. The Socket class can be used to create a socket.

#ServerSocket class

The ServerSocket class can be used to create a server socket. This object is used to establish communication with the clients.

Creating Server:

To create the server application, we need to create the instance of ServerSocket class. Here, we are using 6666 port number for the communication between the client and server. You may also choose any other port number. The accept() method waits for the client. If clients connects with the given port number, it returns an instance of Socket.

ServerSocket ss=new ServerSocket(6666);

Socket s=ss.accept();//establishes connection and waits for the client

Creating Client:

To create the client application, we need to create the instance of Socket class. Here, we need to pass the IP address or hostname of the Server and a port number. Here, we are using "localhost" because our server is running on same system.

Socket s=**new** Socket("localhost",6666);

**Code:**

MyServer.java file

import java.io.\*; import java.net.\*; public class MyServer

public static void main(String[] args){ try

{

ServerSocket ss=new ServerSocket(6666); Socket s=ss.accept();//establishes connection

DataInputStream dis=new DataInputStream(s.getInputStream()); String str=(String)dis.readUTF();

System.out.println("message= "+str); ss.close();

}

catch(Exception e){System.out.println(e);}

}

}

MyClient.java file

import java.io.\*; import java.net.\*; public class MyClient

{

public static void main(String[] args)

{

try

{

Socket s=new Socket("localhost",6666);

DataOutputStream dout=new DataOutputStream(s.getOutputStream()); dout.writeUTF("Hello Server");

dout.flush();

dout.close();

s.close();

}catch(Exception e){System.out.println(e);}

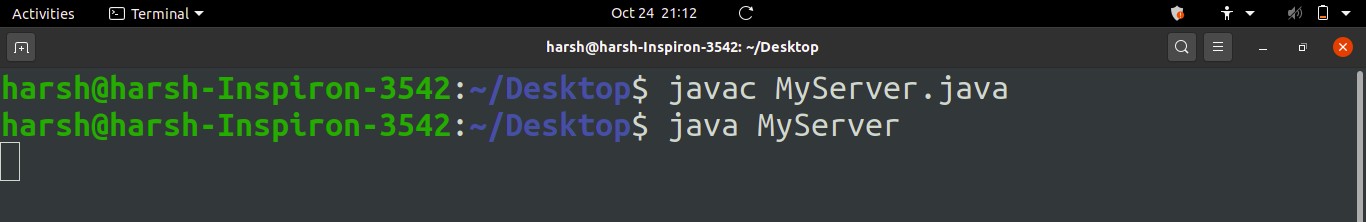
}

}

**Output:**

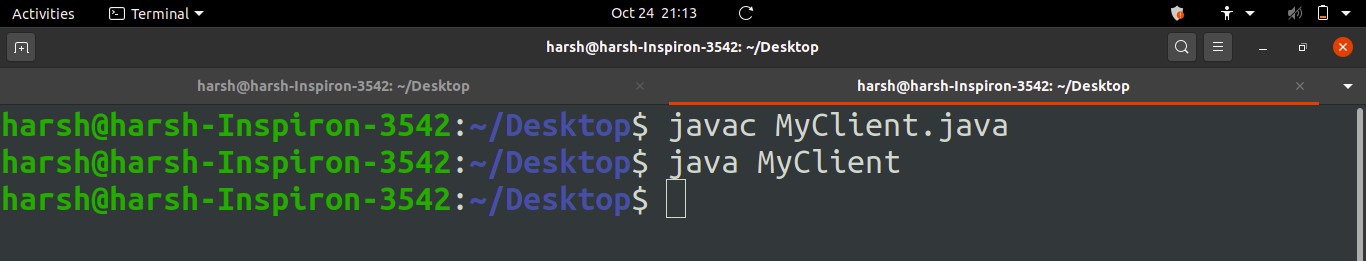
To execute this program open two command prompts and execute each program at each command prompt as displayed in the below figures.

First run Myserver.java file in terminal/cmd,



Running MyServer.java

Then in new terminal/cmd run MyClient.java file,



Running MyClient.java

As soon as you run MyClient program a message is sent to server and displayed in MyServer Terminal/CMD as shown below,

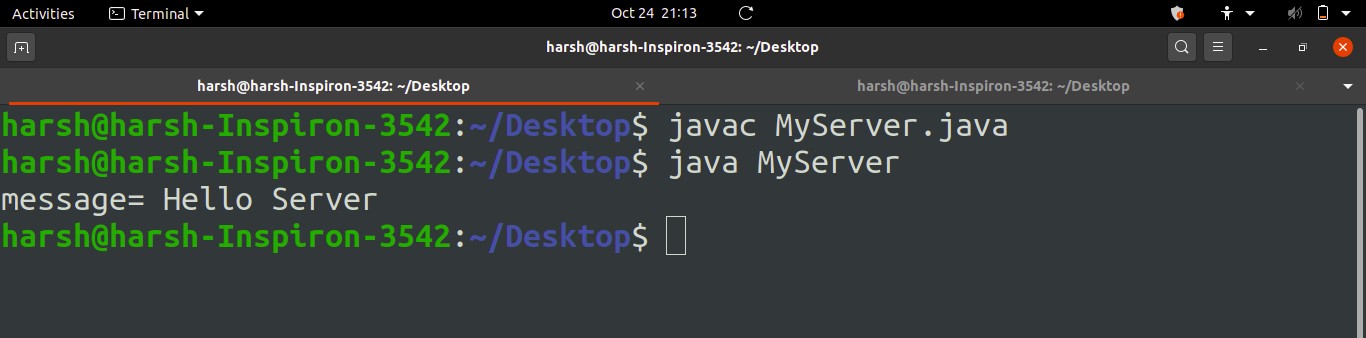


Fig. Message displayed in MyServer after running MyClient

Output

Client and server communication output window.

1. **Output Analysis:**

(Students should write output analysis based on the working of different topology and different networking devices used in simulation. Specify each scenario explicitly with output analysis)

1. **Additional Learning:**

(Students should write additional learning on their own based on what additionally they learnt after performing the experiment)

1. **Conclusion :**

(Students should write conclusion on their own)

1. **Viva Questions:**

* State the promitives used for communication between client and server.
* Communication is always initiated by client, mention Jusification.
* Server has to be in passive mode. Justify.
* Explain the conept of socket in detail.

1. **References:**
   1. A.S. Tanenbaum, “Computer Networks”, Pearson Education, (4e).
   2. B.A. Forouzan, “Data Communications and Networking”, TMH (5e).
   3. James F. Kurose & K W Ross: Computer Networking: A Top Down Approach, Pearson Education (LPE).