INSTITUTE OF ENGINEERING & MANAGEMENT

IEM

Institute of Engineering & Management

NAME OF THE STUDENT : Abhishek Anand

SEMESTER : 6th

ROLL NO. : 57

ASSIGNMENT OF DEPARTMENT : Computer Networks Lab

DATE OF EXPERIMENT / PROJECT : 07/04/2022

DATE OF SUBMISSION :

TITLE : Socket programming with TCP & UDP

OBJECT : Socket programming with TCP & UDP.

Objective :- Socket programming with UDP & TCP.

Introduction :-

Socket programming is used to communicate between the application. In Java, we communicate between different JRE running applications. It can be connection oriented or connection less. The client in socket programming must know two information. ie →(a) IP address or Server.
(b) Port number of application process.

- TCP → (Transmission control protocol) is a network protocol that connect two hosts to exchange data streams. It guarantees the delivery of data & packets in the same order as they were sent.
  → It is connection oriented (Hand shaking procedure)
  → Reliable byte-stream.

- UDP → (User datagram protocol) is a lightweight data transport protocol that works on top of IP.
  → detects corrupt data in packets but not solve.
  → connectionless
  → Not Reliable data transfer.
  → uses DNS lookup to find the IP-address for "host name".

* Code →

TCP- client in java

```
import . java . io . * ;
import java . net . * ;
class TcpClient {
            public static void main ( string args[]) throws Exception
  {
     String line, newLine ;
     try {
         DataInputStream in = new DataInputStream (System.in);
         Socket cs = new Socket ("LocalHost", 6789) ;
         System . out . println (" client started ....") ;
DataInputStream inp = new DataInputstream (cs.getInputStream());
Data OutputStream out = new DataOutputStream(cs.getOutputStream());
while (true) {
            newLine = in . readLine() ;
    if ( newLine . equals ("q")) {
         out . writeBytes (" client is down ...." + '\n');
         return ;
        }
    else {
         Out . writeBytes ( newLine + '\n') ;
         }
      line = inp . readLine () ;
   System . out . println (" Received from server : " + line);
          }
      }
     catch (Exception e) {
                     }
         }
      }
```

TCP server in java :

```
import java.io.* ;
import java.io.* ;
class TcpServer {
    public static void main(string args[]) throws Exception {
        try {
            string line, newLine ;
            ServerSocket ss = new ServerSocket ( 6789) ;
        while (true) {
              Socket s = ss . accept () ;
          System.out.println(" server started ....");
    DataInputStream inp = new DataInputStream (s.getInputstream());
    DataInput Stream out = new DataOutputStream( s.getOutputstream( ));
    DataInputStream in = new Data Input stream( System. in);
    while (true){
      System.out.println(" Press 'q' if you want to exit server");
       line = inp.readLine ();
      System.out.println (" Received from client : " + line);
       newLine = in. read Line () ;
       if ( newLine.equals ( " q" )){
           Out.writeBytes (" Server is down..." + '\n');
            return ;
            }
          else {
                 out.writeBytes (newLine + '\n') ;
                 }
              }
           }
         }
      catch ( Exception e) {          }
        }
      }
```

Name : Abhishek Anand Roll No. : 57

Section : A Year : 3rd .

UDP client in java:

```
import java.io.* ;
import java.net.* ;
class UDPclient {
  public static void main (string args[]) throws Exception{
 Buffered Reader inFromUser = new BufferedReader (new InputStreamReader
                                            (System.in));
DatagramSocket clientSocket = new DatagramSocket();
InetAddress IPAddress = Inet Address .getByName ("hostname");
byte[] send Data = new byte [1024];
byte [] receive Data = new byte [1024];
String sentence = inFrom User . readLine ();
Send Data = sentence . getBytes ();
DatagramPacket send Packet = new DatagramPacket ( send Data,
                    Send Data .length , IPAddress, 9876);
 client Socket . send ( Send Packet );
Datagram Packet receivePacket = new Datagram Packet ( receiveData,
                          receive Data . length);
clientSocket . receive ( receive Packet);
String modefied sentence = new String (receive Packet.get data());
System.out . println (" FROM SERVER : " + modified sentence);
 client socket .close() ;
   }
 }
```

UDP server in Java.

```
import java.io.* ;
import java.net.* ;
Class UDP client {
      Public static void main( string args[]) throws Exception{
```

```
DatagramSocket ServerSocket = new DatagramSocket(9876);
 byte[] receiveData = new byte[1024];
 byte[] sendData = new byte[1024];
 while(true){
  Datagram Packet receivePacket = new DatagramPacket(receiveData,
                                         receiveData.length);
   ServerSocket.receive(receivePacket);
  String sentence = new string(receivePacket.getData());
  InetAddress IPAddress = receivePacket.getAddress();
  int port = receivePacket.getPort();
  String catitalizedSentence = sentence.toUpperCase();
  Send Data = capetalizedSentence.getBytes();
  DatagramPacket sendPacket = new DatagramPacket(sendData,
                    SendData.length,IPAddress, port);
   client Socket.Send (sendPacket);
 DatagramPacket receivePacket = new DatagramPacket(receiveData,
                                         receiveData.length);
    ServerSocket.Send(sendPacket);
     }
    }
}
```

Discussion:

For the socket programming with UDP & TCP, I communicated in same device as well as different devices for both UDP & TCP. In background I used wireshark to see the data packets and the communication. Firstly I run the Server side code, then I run the client side code & then communication established. In wireshark I got all the details related to TCP & UDP packets.