EXPERIMENT NO. 10

Aim :- Socket programming using TCP or UDP

Theory :-

Socket programming using TCP (Transmission Control Protocol) and UDP (User Datagram Protocol) involves establishing network communication between two devices or processes. Here's a theoretical overview of these two approaches :

1. TCP (Transmission Control Protocol):

Connection Oriented: TCP is a connection-oriented protocol, which means it establishes a reliable, ordered and error-checked connection between sender and receiver.

Reliability : TCP ensures that data is transmitted accurately and completely. It uses mechanisms like acknowledgements and retransmissions to guarantee data delivery.

Stream-Oriented: TCP is stream-oriented, which implies that it sends and receives data as a continuous stream. It provides a byte-stream service, ensuring data integrity but not preserving message boundaries.

Usage : UDP is suitable for applications where low overhead and speed are more important than data integrity, such as real-time multimedia streaming, online gaming, and DNS.

Socket Functions : In UDP socket programming, commonly used functions include socket(), bind(), sendto(), etc.

- In both TCP and UDP socket programming :-

Socket Creation : A socket is created using socket() function, specifying protocol (TCP or UDP) and addressing information.

Binding : The bind() function associates socket with a specific address and port.

- Listening (TCP only) : In TCP, the listen() function prepares socket to accept incoming connections.

Connection Establishment (TCP only) : The connect() function establishes a connection to another device in TCP.

Data Transfer : Data is sent using send() and received using recv() for TCP, while sendto() and recvfrom() are used for UDP.

Termination (TCP only): TCP sockets involve a connection termination process, typically initiated by close() function.

Conclusion :- Thus, we implemented socket programming using JAVA language.

A

18/10/23

```
import java.net.*;

import java.io.*;

import java.util.*;

class MyClient{

public static void main(String args[])throws Exception{

Socket s=new Socket("localhost",3333);

DataInputStream din=new DataInputStream(s.getInputStream());

DataOutputStream dout=new DataOutputStream(s.getOutputStream());

BufferedReader br=new BufferedReader(new InputStreamReader(System.in));


String str="",str2="";

while(!str.equals("stop")){

System.out.print("Enter number: ");

//str=br.readLine();

Scanner sc = new Scanner(System.in);

int n = 10;

n = sc.nextInt();

//dout.writeUTF(str);

//dout.writeUTF("Hello guys Chai Peelo");

dout.writeUTF(Integer.toString(n));

dout.flush();

str2=din.readUTF();

System.out.println("Server says: "+str2);

}


dout.close();

s.close();

}}


import java.net.*;
import java.io.*;
class Server{
public static void main(String args[])throws Exception{
ServerSocket ss=new ServerSocket(3333);
Socket s=ss.accept();
DataInputStream din=new DataInputStream(s.getInputStream());
DataOutputStream dout=new DataOutputStream(s.getOutputStream());
BufferedReader br=new BufferedReader(new InputStreamReader(System.in));

String str="",str2="";String num;
while(!str.equals("stop")){
num=din.readUTF();
//System.out.println("client says: "+str);
System.out.println("Number given by client: "+num);
int n = Integer.parseInt(num);
int square = n*n;
System.out.println("Square of Number given by client: "+square);
//str2=br.readLine();
dout.writeUTF("Square of Number given by you: "+square);
dout.flush();
}
din.close();
s.close();
ss.close();
}}
```

Output:

Client:

```
Microsoft Windows [Version 10.0.22621.2428]
```

```
(c) Microsoft Corporation. All rights reserved.

C:\Users\Rishab\OneDrive\Desktop\CN Experiments>javac MyClient.java

C:\Users\Rishab\OneDrive\Desktop\CN Experiments>java MyClient
Enter number: 4
Server says: Square of Number given by you: 16
```

Server:

```
Microsoft Windows [Version 10.0.22621.2428]
(c) Microsoft Corporation. All rights reserved.

C:\Users\Rishab\OneDrive\Desktop\CN Experiments>javac Server.java

C:\Users\Rishab\OneDrive\Desktop\CN Experiments>java Server
Number given by client: 4
Square of Number given by client: 16
```