**1) client and server program input 2 number and the server will multiply it by two:**

**import** java.util.Scanner;

**import** java.io.IOException;

**import** java.io.PrintStream;

**import** java.net.Socket;

**import** java.net.UnknownHostException;

**public** **class** client {

 **public** **static** **void** main(String[] args) **throws** UnknownHostException, IOException {

 **int** number;

 **int** temp;

 Scanner sc = **new** Scanner(System.***in***);

 Socket s = **new** Socket("10.0.4.137", 546);

 Scanner sc1 = **new** Scanner(s.getInputStream());

 System.***out***.println("Enter any number: ");

 number = sc.nextInt();

 PrintStream p = **new** PrintStream(s.getOutputStream());

 p.println(number);

 temp=sc1.nextInt();

 System.***out***.println(temp);

 }

}

**import** java.util.Scanner;

**import** java.io.IOException;

**import** java.io.PrintStream;

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** Server {

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

 **int** number;

 **int** temp;

 ServerSocket s1 = **new** ServerSocket (546);

 Socket ss = s1.accept();

 Scanner sc = **new** Scanner(ss.getInputStream());

 number = sc.nextInt();

 temp=number \* 2;

 PrintStream p = **new** PrintStream(ss.getOutputStream());

 p.println(temp);

 }

}

2.

// SimpleServer.java: A simple server program.

**import** java.net.\*;

**import** java.io.\*;

**public** **class** SimpleServer {

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

 // Register service on port 1254

 ServerSocket s = **new** ServerSocket(456);

 Socket s1=s.accept(); // Wait and accept a connection

 // Get a communication stream associated with the socket

 OutputStream s1out = s1.getOutputStream();

 DataOutputStream dos = **new** DataOutputStream (s1out);

 // Send a string!

 dos.writeUTF("Hi there");

 // Close the connection, but not the server socket

 dos.close();

 s1out.close();

 s1.close();

 }

}

// SimpleClient.java: A simple client program.

import java.net.\*;

import java.io.\*;

public class SimpleClient {

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

// Open your connection to a server, at port 1254

 Socket s1 = new Socket("localhost",456);

// Get an input file handle from the socket and read the input

 InputStream s1In = s1.getInputStream();

 DataInputStream dis = new DataInputStream(s1In);

 String st = new String (dis.readUTF());

 System.out.println(st);

 // When done, just close the connection and exit

 dis.close();

 s1In.close();

 s1.close();

 }

}

3. message = Hello Server

**import** java.io.\*;

**import** java.net.\*;

**public** **class** Myclient1 {

 **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);

 }

 }

}

**import** java.io.\*;

**import** java.net.\*;

**public** **class** Myserver1 {

 **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);

 }

 }

}

4. Waiting for Clients... connection established

**import** java.net.ServerSocket;

**import** java.net.Socket;

**public** **class** EchoServer {

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

**try**

{System.***out***.println("Waiting for clients...");

ServerSocket ss = **new** ServerSocket(546);

Socket soc = ss.accept();

System.***out***.println("Connections established");

 }

**catch**(Exception e)

{

 e.printStackTrace();

}

}

}

**import** java.net.Socket;

**public** **class** EchoClient {

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

 **try**

 {

 System.***out***.println("Client started");

 Socket soc = **new** Socket("localhost", 546);

 }

**catch** (Exception e)

 {

 e.printStackTrace();

 }

 }

}

5.

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.InetAddress;

**import** java.net.SocketException;

**import** java.net.UnknownHostException;

**public** **class** UdpUnicastServer **implements** Runnable{

 **private** **final** **int** clientPort;

 **public** UdpUnicastServer (**int** clientPort) {

 **this**.clientPort = clientPort;

 }

@Override

**public** **void** run() {

 **try**(DatagramSocket serverSocket = **new** DatagramSocket (546)){

 **for** (**int** i = 0; i<3; i++) {

 String message = "Message number " + i;

 DatagramPacket datagramPacket = **new** DatagramPacket(

 message.getBytes(),

 message.length(),

 InetAddress.*getLocalHost*(),

 clientPort

 );

 serverSocket.send(datagramPacket);

 }

 }

**catch** (SocketException e) {

 e.printStackTrace();

 }

**catch** (UnknownHostException e) {

 e.printStackTrace();

 }

**catch** (IOException e) {

 e.printStackTrace();

 }

}

}

**import** java.io.IOException;

**import** java.net.DatagramPacket;

**import** java.net.DatagramSocket;

**import** java.net.SocketException;

**public** **class** UdpUnicastClient **implements** Runnable{

**private** **final** **int** port;

**public** UdpUnicastClient(**int** port) {

 **this**.port = port;

}

@Override

**public** **void** run() {

 **try**(DatagramSocket clientSocket = **new** DatagramSocket (port)){

 **byte**[] buffer = **new** **byte** [6550];

 clientSocket.setSoTimeout(3000);

 **while** (**true**){

 DatagramPacket datagramPacket = **new** DatagramPacket (buffer,0,buffer.length);

 clientSocket.receive(datagramPacket);

 String receivedMessage = **new** String (datagramPacket.getData());

 System.***out***.println(receivedMessage);

 }

 }**catch** (SocketException e) {

 e.printStackTrace();

 }

 **catch**(IOException e) {

 System.***out***.println("Timeout, Client is closing");

 }

}

}

**import** java.util.concurrent.ExecutorService;

**import** java.util.concurrent.Executors;

**public** **class** Main {

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

 **int** port = 50001;

 UdpUnicastServer server = **new** UdpUnicastServer(port);

 UdpUnicastClient client = **new** UdpUnicastClient(port);

 ExecutorService executorService = Executors.*newFixedThreadPool*(2);

 executorService.submit(client);

 executorService.submit(server);

 }

}

6. // UDPServer.java: A simple UDP server program.

**import** java.net.\*;

**import** java.io.\*;

**public** **class** UDPServer {

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

 DatagramSocket aSocket = **null**;

 **if**(args.length<1)

 { System.***out***.println("Usage: java UDPServer ");

 System.*exit*(1);

 }

 **try** { **int** socket\_no = Integer.*valueOf*(args[0]).intValue(); aSocket = **new** DatagramSocket(socket\_no); **byte**[] buffer = **new** **byte**[1000];

 **while**(**true**) {

 DatagramPacket request = **new** DatagramPacket(buffer, buffer.length); aSocket.receive(request);

 DatagramPacket reply = **new** DatagramPacket(request.getData(), request.getLength(),request.getAddress(), request.getPort());

 aSocket.send(reply); }

 } **catch** (SocketException e) {

 System.***out***.println("Socket: " + e.getMessage()); }

 **catch** (IOException e) { System.***out***.println("IO: " + e.getMessage()); }

 **finally** { **if** (aSocket != **null**)

 aSocket.close();

 }

 }

}

// UDPClient.java: A simple UDP client program.

**import** java.net.\*;

**import** java.io.\*;

**public** **class** UDPClient {

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

// args give message contents and server hostname

 DatagramSocket aSocket = **null**;

 **if** (args.length < 3) {

 System.***out***.println( "Usage: java UDPClient ");

 System.*exit*(1);

}

 **try** {

 aSocket = **new** DatagramSocket();

 **byte** [] m = args[0].getBytes();

 InetAddress aHost = InetAddress.*getByName*(args[1]); **int** serverPort = Integer.*valueOf*(args[2]).intValue(); DatagramPacket request =

 **new** DatagramPacket(m, args[0].length(), aHost, serverPort); aSocket.send(request);

 **byte**[] buffer = **new** **byte**[1000];

 DatagramPacket reply = **new** DatagramPacket(buffer, buffer.length); aSocket.receive(reply);

 System.***out***.println("Reply: " + **new** String(reply.getData())); }

 **catch** (SocketException e) {

 System.***out***.println("Socket: "+ e.getMessage()); }

 **catch** (IOException e) {

 System.***out***.println("IO: " + e.getMessage()); }

 **finally** { **if** (aSocket != **null**) aSocket.close();

 }

 }

 }

7. Input - chatting

**import** java.net.\*;

**import** java.io.\*;

**class** MyclientInput {

 **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")) {

 str = br.readLine();

 dout.writeUTF(str);

 dout.flush();

 str2 = din.readUTF();

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

 }

 dout.close();

 s.close();

 }

}

**import** java.net.\*;

**import** java.io.\*;

**class** MyServerInput {

 **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 = "";

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

 str = din.readUTF();

 System.***out***.println("client says: " + str);

 str2 = br.readLine();

 dout.writeUTF(str2);

 dout.flush();

 }

 din.close();

 s.close();

 ss.close();

 }

}

8. InetAddress

**import** java.io.\*;

**import** java.net.\*;

**public** **class** InetDemo {

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

 **try** {

 InetAddress ip = InetAddress.*getByName*("www.javatpoint.com");

 System.***out***.println("Host Name: " + ip.getHostName());

 System.***out***.println("IP Address: " + ip.getHostAddress());

 } **catch** (Exception e) {

 System.***out***.println(e);

 }

 }

}

9. Sending and Receiving DatagramPacket

//DSender.java

**import** java.net.\*;

**public** **class** DSender {

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

 DatagramSocket ds = **new** DatagramSocket();

 String str = "Welcome java";

 InetAddress ip = InetAddress.*getByName*("127.0.0.1");

 DatagramPacket dp = **new** DatagramPacket(str.getBytes(), str.length(), ip, 3000);

 ds.send(dp);

 ds.close();

 }

}

//DReceiver.java

**import** java.net.\*;

**public** **class** DReceiver {

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

 DatagramSocket ds = **new** DatagramSocket(3000);

 **byte**[] buf = **new** **byte**[1024];

 DatagramPacket dp = **new** DatagramPacket(buf, 1024);

 ds.receive(dp);

 String str = **new** String(dp.getData(), 0, dp.getLength());

 System.***out***.println(str);

 ds.close();

 }

}