**Establishing TCP Connections**

Client Class:

1. import java.io.\*;
2. import java.net.\*;
4. public class Client{
6. public static void main(String[] args) throws Exception{
7. Socket s = new Socket("LocalHost",6666);
8. DataOutputStream dos = new DataOutputStream(s.getOutputStream());
9. System.out.println("Connected Successfully");
10. dos.writeUTF("Connection Has Been Established Successfully");
11. s.close();
12. }
13. }

Server Class:

1. import java.net.\*;
2. import java.io.\*;
4. public class SocketServer{
5. public static void main(String[] args) throws Exception{
6. ServerSocket serverSocket = new ServerSocket(1212); //1212 is the port number
7. Socket s = serverSocket.accept(); //Accepting connection with the server socket
8. DataInputStream ins = new DataInputStream(s.getInputStream());
9. String str = (String) ins.readUTF();
10. System.out.println(str);
11. }
12. }

**URL Class**

1. import java.net.\*;
2. import java.io.\*;
4. class URLClass{
5. public static void main(String[] args) throws Exception{
6. URL url = new URL("http://pratyayDhond.me/Self-Assess");
7. String protocol = url.getProtocol();
8. String hostName = url.getHost();
9. int port = url.getPort(); // If port is left as it is -1 by default
10. String fileName = url.getFile();
12. System.out.println("Protocol : " + protocol);
13. System.out.println("Host Name : " + hostName);
14. System.out.println("Port : " + port);
15. System.out.println("File Name: " + fileName);
16. }
17. }

**InetAddress Class**

1. import java.net.\*;
2. import java.io.\*;
4. class InetAddressProgram{
5. public static void main(String[] args) throws Exception{
6. InetAddress inet = InetAddress.getByName("www.pratyayDhond.me");
7. System.out.println("Domain Name : " + inet.getHostName());
8. System.out.println("IP Address : " + inet.getHostAddress());
9. }
10. }

Client Class:

import java.net.\*;

import java.io.\*;

import java.util.Vector;

public class Client{

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

DatagramSocket ds = new DatagramSocket();

InetAddress inet = InetAddress.getLocalHost();

//For storing input and sending all input directly

Vector<String> s = new Vector<String>();

//for taking input from the user

InputStreamReader isr = new InputStreamReader(System.in);

BufferedReader br = new BufferedReader(isr);

//for sending data in form of packets

int count = 0;

do{

String str = new String();

str = br.readLine();

s.add(str);

count++;

}while(!s.elementAt(count-1).equals("-1"));

for(int i = 0;i < s.size()-1;i++){

byte[] b = s.elementAt(i).getBytes();

DatagramPacket dp = new DatagramPacket(b,b.length,inet,1234);

System.out.println(dp.getData());

ds.send(dp);

}

String str = "END\_CONNECTION";

byte[] b =str.getBytes();

DatagramPacket dp = new DatagramPacket(b,b.length,inet,1234);

ds.send(dp);

isr.close();

br.close();

ds.close();

System.out.println("Connection closed successfully");

}

}

Server Class:

1. import java.net.\*;
2. import java.io.\*;
4. class Server{
6. final static String END\_CONNECTION = "END\_CONNECTION";
7. public static void main(String[] args) throws Exception{
8. DatagramSocket ds = new DatagramSocket(1234);
10. do{
11. byte[] b = new byte[1024];
12. DatagramPacket dp = new DatagramPacket(b,b.length);
13. ds.receive(dp);
14. String str = new String(dp.getData()).trim();
16. if(str.trim().contains(END\_CONNECTION))
17. break;
18. else
19. System.out.println(str.trim());
20. }while(true);
22. }
23. }

**One Way communication:**

Client Class:

1. import java.net.\*;
2. import java.io.\*;
4. class Client {
5. public static void main(String[] args) throws Exception{
6. Socket s = new Socket("Localhost",1234);
8. DataOutputStream dout = new DataOutputStream(s.getOutputStream());
10. InputStreamReader isr = new InputStreamReader(System.in);
11. BufferedReader br = new BufferedReader(isr);
13. String str;
14. while(true){
15. str = br.readLine();
17. if(str.equals("-1"))
18. break;
19. else
20. dout.writeUTF(str);
21. }
22. dout.writeUTF("END\_CONNECTION");
24. dout.close();
25. s.close();
26. br.close();
27. isr.close();
28. System.out.print("\nConnection Closed Succesfully");
29. }
30. }

Server Class:

1. import java.net.\*;
2. import java.io.\*;
4. class Server {
6. final static String END\_CONNECTION = "END\_CONNECTION";
8. public static void main(String[] args) throws Exception{
9. ServerSocket serverSocket = new ServerSocket(1234);
10. Socket s = serverSocket.accept();
11. System.out.println("Connection Established\n");
12. DataInputStream din = new DataInputStream(s.getInputStream());
14. while(true){
15. String str = din.readUTF();
17. if(str.contains(END\_CONNECTION))
18. break;
19. else
20. System.out.println(str);
21. }
23. System.out.println("\n" + "Connection Closed Successfully");
24. }
25. }

**Two Way communication:**

Client Class:

1. import java.io.\*;
2. import java.net.\*;
4. class Client{
5. public static void main(String[] args) throws Exception{
6. System.out.println("--------------------Client Program-------------------- \n");
7. Socket s = new Socket("LocalHost",2003);
8. DataInputStream din = new DataInputStream(s.getInputStream());
9. DataOutputStream dos = new DataOutputStream(s.getOutputStream());
11. BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
13. String str1 = "",str2 = "";
15. while(!str1.equals("-1") && !str2.equals("-1")){
16. System.out.print("> ");
17. str1 = br.readLine();
18. dos.writeUTF(str1);
19. str2 = din.readUTF();
20. if(!str2.equals("-1"))
21. System.out.println("Server: " + str2);
22. else{
23. dos.writeUTF("-1");
24. break;
25. }
26. }
27. din.close();
28. dos.close();
29. s.close();
30. System.out.println("Connection closed");
31. }
32. }

Server Class:

1. import java.io.\*;
2. import java.net.\*;
4. class Server{
5. public static void main(String[] args) throws Exception{
6. System.out.println("--------------------Server Program-------------------- \n");
7. ServerSocket ss = new ServerSocket(2003);
8. Socket s = ss.accept();
10. DataInputStream din = new DataInputStream(s.getInputStream());
11. DataOutputStream dos =new DataOutputStream(s.getOutputStream());
13. BufferedReader br = new BufferedReader(new InputStreamReader(System.in));
14. String str1="",str2="";
16. while(!str1.equals("-1") && !str2.equals("-1")){
17. str1 = din.readUTF();
18. if(!str1.equals("-1"))
19. System.out.println("Client: " + str1);
20. else{
21. dos.writeUTF("-1");
22. break;
23. }
24. System.out.print("> ");
25. str2 = br.readLine();
26. dos.writeUTF(str2);
27. }
29. dos.close();
30. din.close();
31. ss.close();
32. s.close();
33. System.out.println("Server disconnected successfully");
34. }
35. }