# Networking

**By using java programming language we are able to prepare the following two types of applications:**

1.Standalone Application

2.Distributed Application

**1.Standalone Application:**

**If we prepare any java application with out using client-server arch. then that java application is called as "Standalone Application".**

**To prepare standalone applicatins we can use Core Libraries like java.io, java.util, java.lang,....**

**2.Distributed Application:**

**If we prepare any java application on the basis of Client-Server arch. then that java application is called as Distributed application.**

**To prepare Distributed applications, JAVA has provided the following set of technologies:**

1.**Socket Programming**

**2.RMI**

**3.CORBA**

**4.EJBs**

**5.WebServices**

**1.Socket Programming:**

**If we want to prepare distributed applications by using Socket Programming then we have to use Sockets between machines to transfer data from one machine to another machine.**

**Socket is a Channel or medium to transfer data from one machine to another machine.**

**In Socket programming we have to establish Sockets on the basis of System IP Address and Port Numbers.**

1. **What is the difference between IPAddress and Port Number?**

**Ans:**

**IP Address is an unique identity to each and every machine over the network and which is provided by network manager at the time of network configuration.**

**Port Number is an unique identity to each and every process being executed with in a single machine and it would be provided by local operating system.**

**To prepare distributed applications by using Socket Programming the required predefined library was provided by JAVA in the form of "java.net" package.**

**Socket programming Arch(heart of socket programming):**

**Diagram:**

**Steps to prepare Client Application**:

**1.Create Socket at client machine**:

**To create Socket class object we have to use the following constructor from java.net.Socket class.**

**public Socket(String server\_IP\_Addr, int server\_Port\_No)**

**EX: Socket s=new Socket("localhost", 4444);**

**NOTE: If server socket is available at the same machine then we are able to use "localhost" inplace of Server\_IP\_Addr.**

**2.Get OutputStream from Socket:**

**To get OutputStream from Socket we have to use the following method from java.net.Socket class.**

**public OutputStream getOutputStream()**

**EX: OutputStream os=s.getOutputStream();**

**3.Create PrintStream with OutputStream:**

PrintStream ps=new PrintStream(os);

1. **Send data to PrintStream:**

**String data="Hello";**

**ps.println(data);**

**NOTE: With the above steps , data will be send to Server, where Server will send response data to client.**

**4.Get InputStream from Socket:**

**To get InputStream from Socket we have to use the following method.**

**public InputStream getInputSteam()**

**EX: InputStream is=s.getInputStream();**

**5.Create BufferedReader with InputStream:**

BufferedReader br=new BufferedReader(new InputStreamReader(is));

**6.Read data from BufferedReader:**

**String data=br.readLine();**

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

**Steps To prepare Server Application:**

**1.Get InputStream from Socket:**

InputStream is=s.getInputStream();

**2.Create BufferedReader with InputStream:**

BufferedReader br=new BufferedReader(new InputStreamReader(is));

**3.Read data from BufferedReader:**

String data=br.readLine();

System.out.println(data);

**4.Get OutputStream from Socket:**

OutputStream os=s.getOutputStream();

**5.Create PrintStream with OutputStream:**

PrintStream ps=new PrintStream(os);

**6.Send Data to PrintStream:**

String data="Hai";

ps.println(data);

**NOTE: At Server machine, we have to create ServerSocket and it has to accept the request from client about to assign Socket from server machine inorder to establish connection .**

**EX: ServerSocket ss=new ServerSocket(4444);**

**Socket s=ss.accept();**

**The above application provides one time communication, but, if we want to provide infinite communication then we have to use infinite loops at both client application and Server application.**

**Applcation-1:**

ClientApp.java

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

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

**public class ClientApp**

**{**

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

**Socket s=new Socket("localhost", 4444);**

**OutputStream os=s.getOutputStream();**

**PrintStream ps=new PrintStream(os);**

**BufferedReader br1=new BufferedReader(new InputStreamReader(System.in));**

**InputStream is=s.getInputStream();**

**BufferedReader br2=new BufferedReader(new InputStreamReader(is));**

**while(true)**

**{**

**String data1=br1.readLine();**

**ps.println(data1);**

**String data2=br2.readLine();**

**System.out.println(data2);**

**if(data1.equals("bye") && data2.equals("bye"))**

**{**

**System.exit(0);**

**}**

**}**

**}**

**}**

**ServerApp.java**

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

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

**public class ServerApp**

**{**

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

**{**

**ServerSocket ss=new ServerSocket(4444);**

**Socket s=ss.accept();**

**InputStream is=s.getInputStream();**

**BufferedReader br1=new BufferedReader(new InputStreamReader(is));**

**OutputStream os=s.getOutputStream();**

**PrintStream ps=new PrintStream(os);**

**BufferedReader br2=new BufferedReader(new InputStreamReader(System.in));**

**while(true)**

**{**

**String data1=br1.readLine();**

**System.out.println(data1);**

**String data2=br2.readLine();**

**ps.println(data2);**

**if(data1.equals("bye") && data2.equals("bye"))**

**{**

**System.exit(0);**

**}**

**}**

**}**

**}**