Network Programming [CACS355] BCA 6th Sem

Er. Sital Prasad Mandal

(Email : info.sitalmandal@gmail.com)
Bhadrapur, Jhapa, Nepal

https://networkprogam-mmc.blogspot.com/

Unit-1
Introduction

1.1.Network Programming Features and Scope

1.2.Network Programming Language, Tools & Platfoms

1.3.Client and Server Applications

1.4.Client Server model and Software Design

Unit-1 Introduction

- Network Programming involves writing programs that communicate with other programs across a computer network.
- A server is an application that provides a "service" to various clients who request the service.
- There are many client/server scenarios in real life:
  - Bank tellers (server) provide a service for the account owners (client)
  - Waitresses (server) provide a service for customers (client)
  - Travel agents (server) provide a service for people wishing to go on vacation (client)

Unit-1
Introduction

Java Networking Programming:

- Java Networking Programming is a concept of connecting two or more computing devices together so that we can share resources with the help of Coding.
- Java socket programming provides facility to share data between different computing devices.

Advantage of Java Networking

1. sharing resources
2. centralize software management

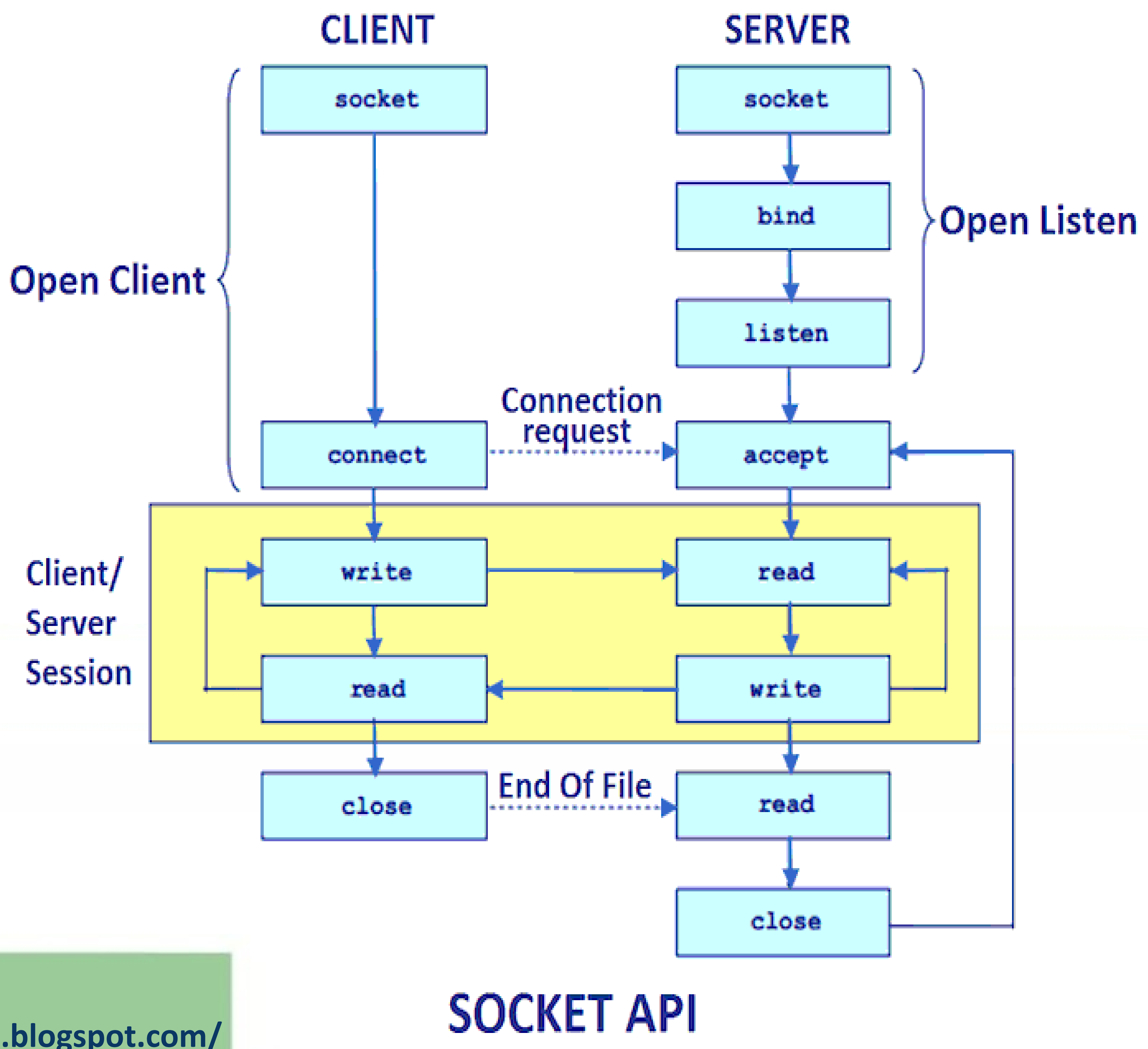
1.3.Client and Server Applications

- Java Socket programming is used for communication between the applications running on different JRE.
- Java Socket programming can be connection-oriented or connection-less.
- Socket and ServerSocket classes are used for connection-oriented socket programming and DatagramSocket and DatagramPacket classes are used for connection-less socket programming.

The client in socket programming must know two information:

1. **IP Address of Server, and**
2. **Port number.**

1.3.Client and Server Applications



1.3.Client and Server Applications

File: MyServer.java

```
import java.io.*;
import java.net.*;
public class MyServer {
public static void main(String[] args){
try{
ServerSocket ss=new ServerSocket(6666);
Socket s=ss.accept();//establishes connection
DataInputStream dis=new DataInputStream(s.get
InputStream());
String  str=(String)dis.readUTF();  //return utf to
string
System.out.println("message= "+str);
ss.close();
}catch(Exception e)
{System.out.println(e);}
}
}
```

File: MyClient.java

```
import java.io.*;
import java.net.*;
public class MyClient {
public static void main(String[] args) {
try{
Socket s=new Socket("localhost",6666);
DataOutputStream dout=new DataOutputStream(s.getOut
putStream());
dout.writeUTF("Hello Server");
dout.flush();
dout.close();
s.close();
}catch(Exception e)
{System.out.println(e);}
}
}
```

To execute this program open two command prompts and execute each program at each command prompt.
After running the client application, a message will be displayed on the server console. UTF-*Stands for "Unicode Transformation Format.*

1.3.Client and Server Applications

For example,

Among the **Constructors** for the *client-side* socket are the following:

- Socket (InetAddress, int) — creates a socket and connects it to the specified port on the host at the specified IP address.
- Socket (String, int) — creates a socket and connects it to the specified port on the host named in String.

Constructors on the *server side* include the following:

- ServerSocket (int) — creates a server socket and binds it to the specified port on the local host.

Do You Know ?

- How to perform connection-oriented Socket Programming in networking ?
- How to display the data of any online web page ?
- How to get the IP address of any host name e.g. www.google.com ?
- How to perform connection-less socket programming in networking ?

1.3.Client and Server Applications

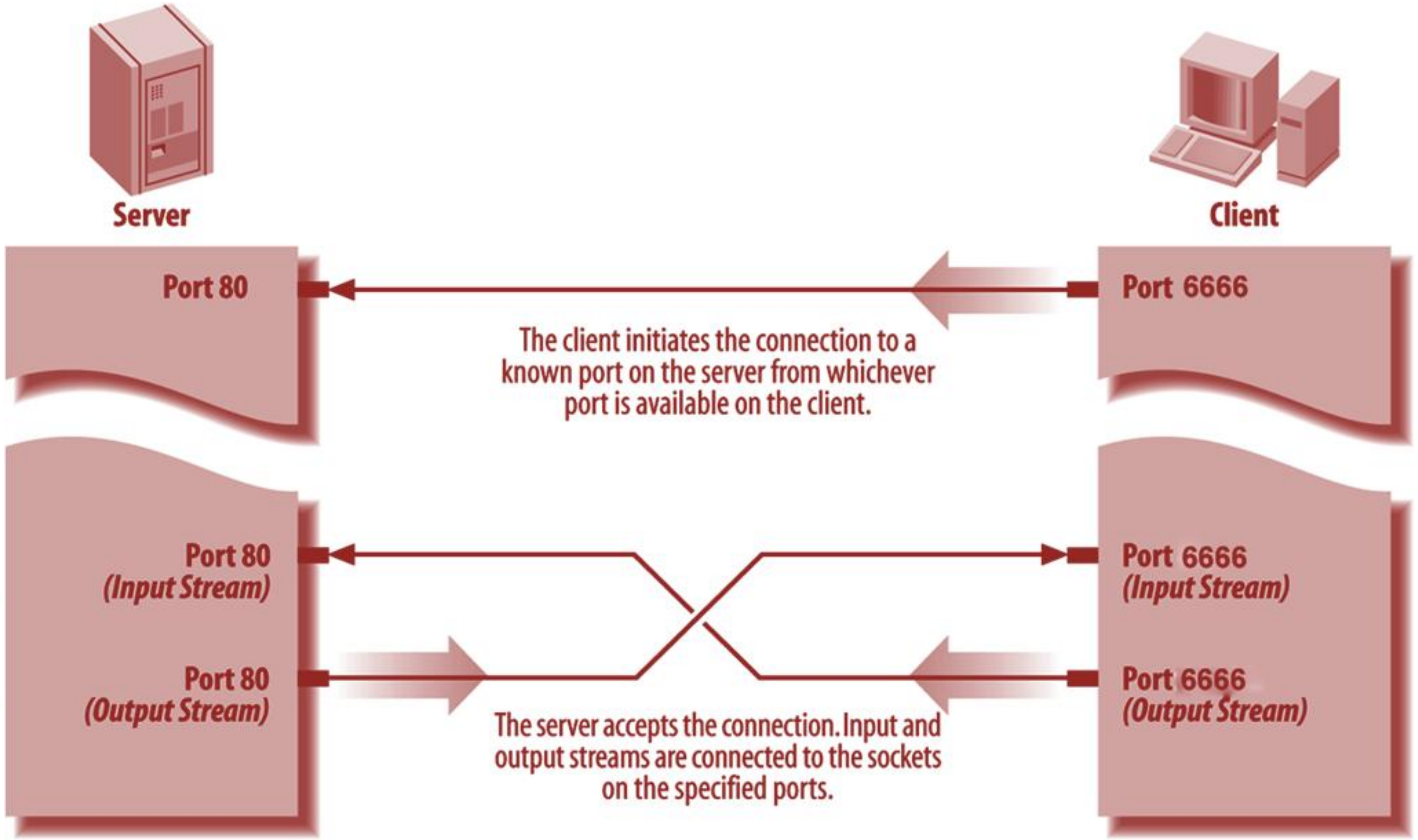

Server
Client
Port 80
Port 6666
The client initiates the connection to a known port on the server from whichever port is available on the client.
Port 80 (Input Stream)
Port 6666 (Input Stream)
Port 80 (Output Stream)
Port 6666 (Output Stream)
The server accepts the connection. Input and output streams are connected to the sockets on the specified ports.

Figure . A client/server connection

Java Networking Terminology

The widely used java networking terminologies are given below:

1. **IP Address**
2. **Protocol**
3. **Port Number**
4. **MAC Address**
5. **Connection-Oriented And Connection-Less Protocol**
6. **Socket**

Java Networking Terminology

1) IP Address

IP address is a unique number assigned to a node of a network e.g. 192.168.0.1 . It is composed of range from 0 to 255. It is a logical address that can be changed.

2) Protocol

A protocol is a set of rules basically that is followed for communication. For example:

• http • TCP • FTP • Telnet • SMTP • POP etc.

3) Port Number

The port number is used to uniquely identify different applications. It acts as a communication endpoint between applications.

The port number is associated with the IP address for communication between two applications.

Java Networking Terminology

4) MAC Address

MAC (Media Access Control) Address is a unique identifier of NIC (Network Interface Controller). A network node can have multiple NIC but each with unique MAC.

5) Connection-Oriented And Connection-Less Protocol

In Connection-Oriented Protocol, acknowledgement is sent by the receiver. So it is reliable but slow. The example of connection-oriented protocol is TCP.

But, in Connection-Less Protocol, acknowledgement is not sent by the receiver. So it is not reliable but fast. The example of connection-less protocol is UDP.

6) Socket

A socket is an endpoint between two way communication.