INSTITUTE OF JAVA &SOFTWARE ENGINEERING

GRADUATE DIPLOMA IN SOFTWARE ENGINEERING

Socket Programming

ASSIGNMENT NO

01

STUDENT NAME: **Ishanka Dilshan**

NIC: **200015302141**

BATCH NO: **GDSE 56**

● What is Java Networking ?

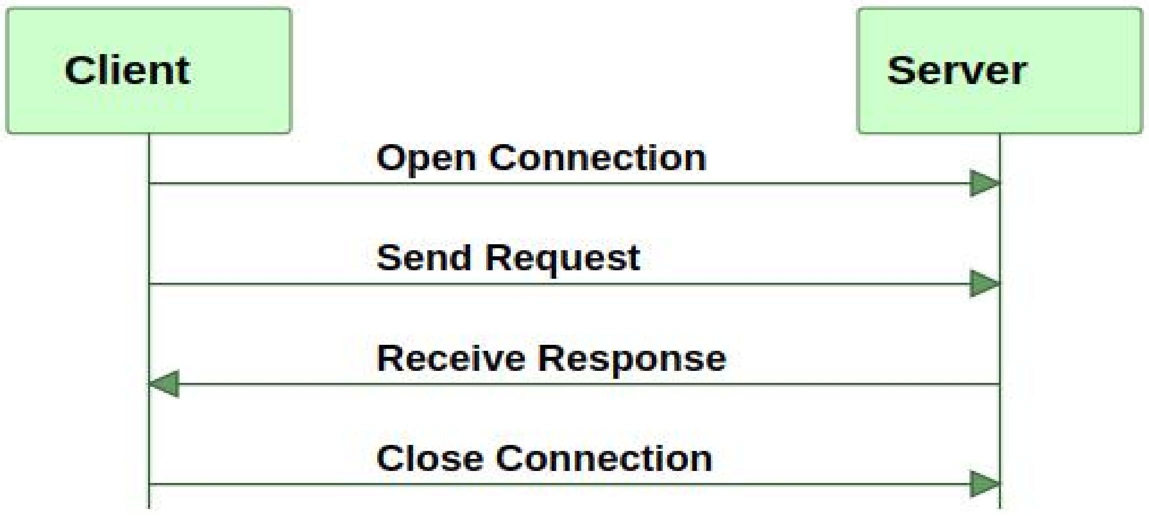
Networking is a concept of connecting two or more computing devices together so that we can share resources. **Java Socket** programming provides facility to share data between different computing devices in Java.

Java program communicates over the network at the application layer, java.net package is useful for all the java networking classes ans interfaces.

● Java Networking Basics

Typically a client opens a TCP/IP connection to a server. The client then starts to communicate with the server.

When the client is finished it closes the connection again.



A client may send more than one request through an open connection. In fact, a client can send as much data as the server is ready to receive. The server can also close the connection if it wants to.

● Advantages and Disadvantages in java networking.

Advantages	Disadvantages
Sockets are flexible and sufficient.	Security restrictions.
Can beeasily implemented for general communications.	Socket allow only raw data to be sent. (both client and server need to have mechanisms to interpret the data)
Only updated information can be send. (Java aplets can send only necessaty update information)	The re-use of socket based implementations is limited.
Low network traffic.	
centralize software management.	
sharing resources.	

- **Java Network Terminology**

The generally used java networking terminologies are :-

1. IP Address (Internet Protocol Address)
2. Protocol
3. Port Number
4. MAC Address (Media Access Control Address)
5. Skeleton
6. Stub
7. Connection-oriented & Connection-less protocol
8. Socket

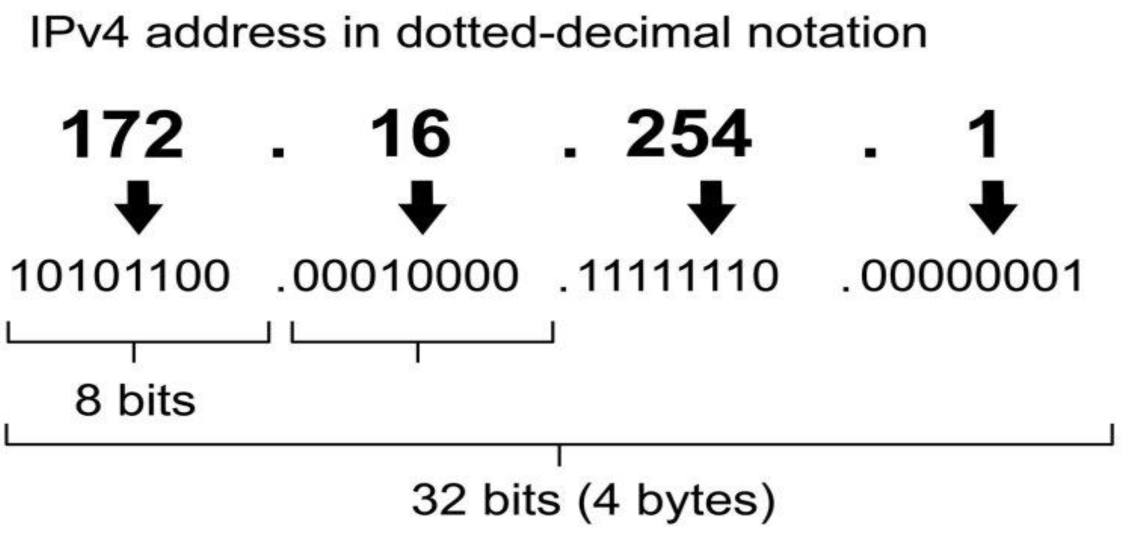
➢ **IP Address**

IP Address is a unique numerical label assigned to each device connected to a computer network.

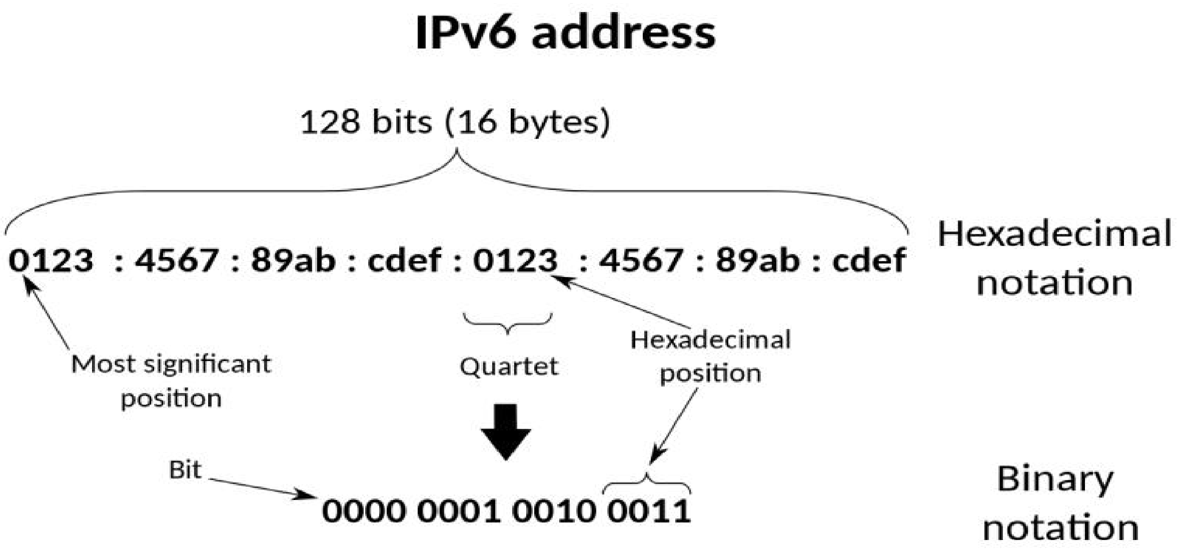
There are two standards for IP addresses

1. IP Version 4 (IPv4)
2. IP Version 6 (IPv6)

- IPv4 uses 32 binary bits to create a signle unique address on the netwrok. An IPv4 address is expressed by four numbers separated by dots. Each number is the decimal representation for an eight-digit binary number, also called an octet. For example : **216.27.61.137**

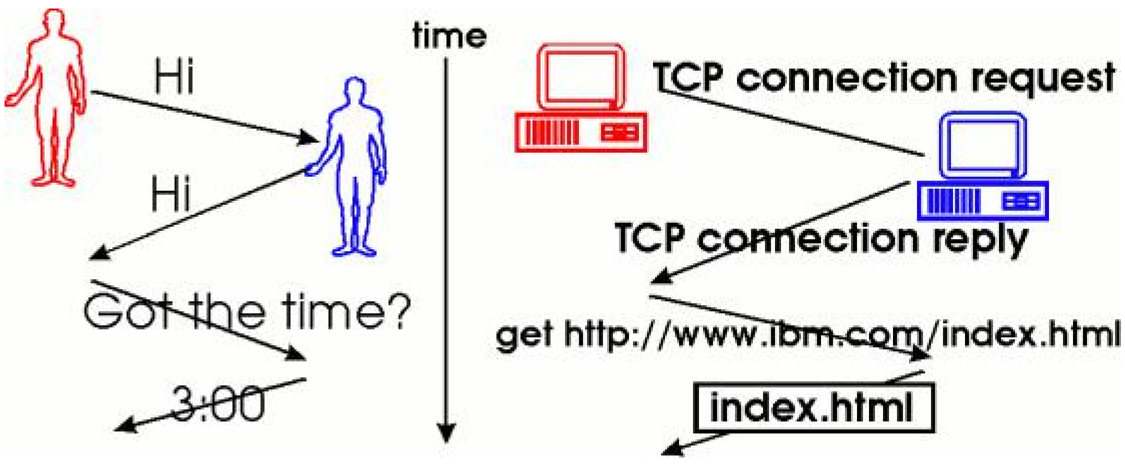


- IPv6 was designed by Internet Engineering Task Force (IETF) in 1998 with the purpose of superseding the IPv4 due to global exponentially growing internet users. An IPv6 address consists of eigth groups of four hexadecimal digits. For Example : **3001:0da8:75a3:0000:0000:8a2e:0370:7334**



➢ **Protocol**

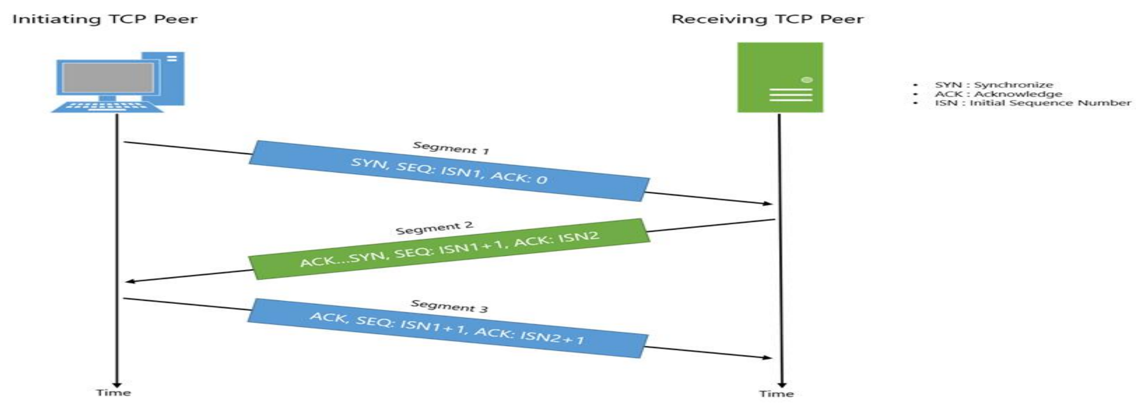
A protocol is a set of rules basically that is followed for communication. For example :



Protocol are used to define a 'blueprint of methods, properties, and other requirements that suit a particular task or piece of functionality'.

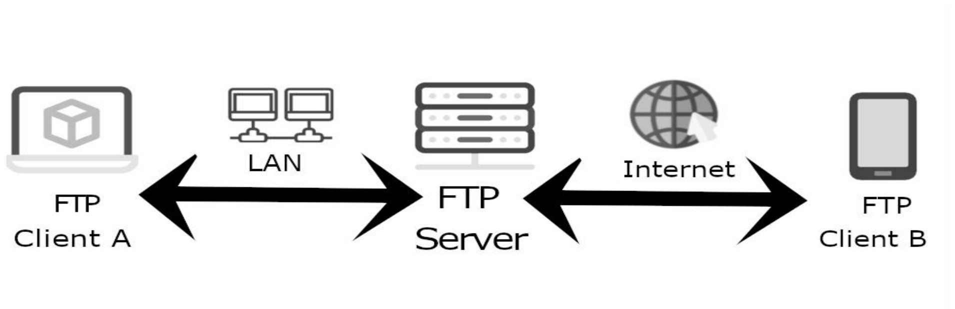
I. TCP - (Transmission Control Protocol)

TCP is a standard that defines how to establish and maintain a network conversation through which appilication can exchange data. TCP works with IP, which defines how computers send packets of data to each other.



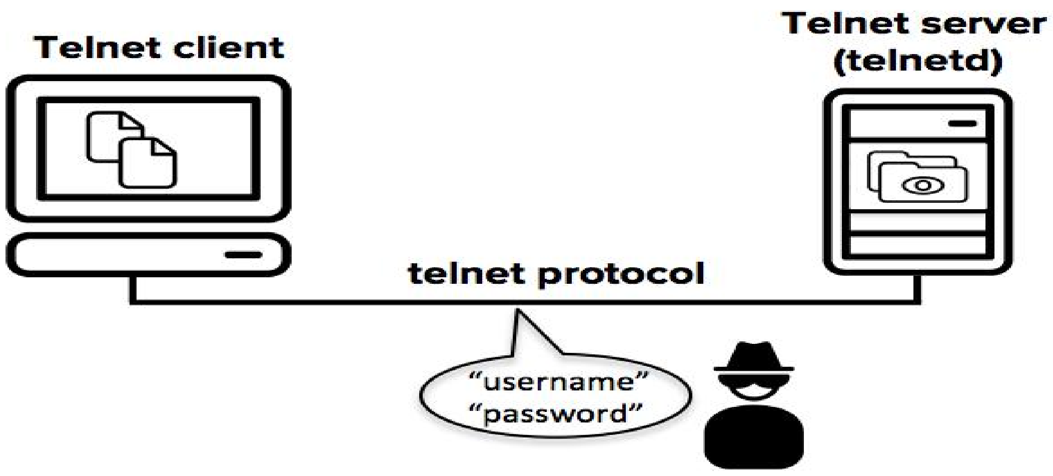
II. FTP - (File Transfer Protocol)

FTP is a standard network protocol used for the tansfer of computer files between client and server on a computer network. FTP is built on a client-server model architecture using seperate control and data connections between the client and server.



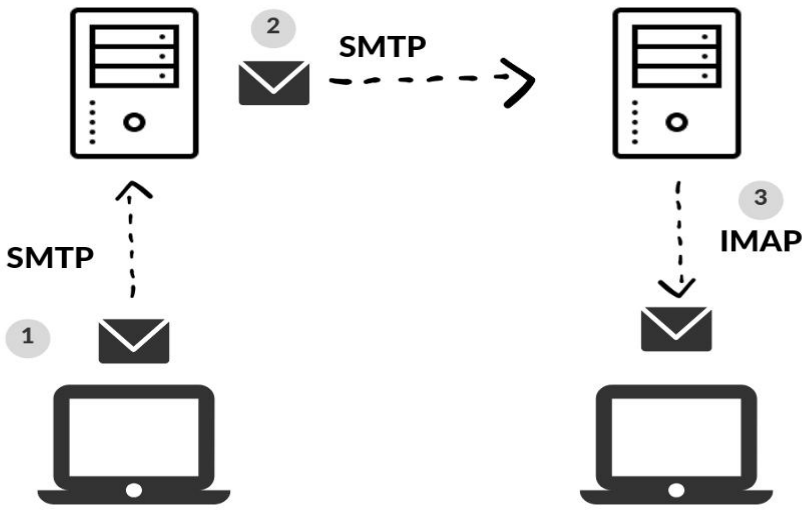
III. Telnet

Telnet is an application protocol used on the internet or local area network to provide a bidirectional interactive text-oriented communication facility using a virtual terminal connnection.



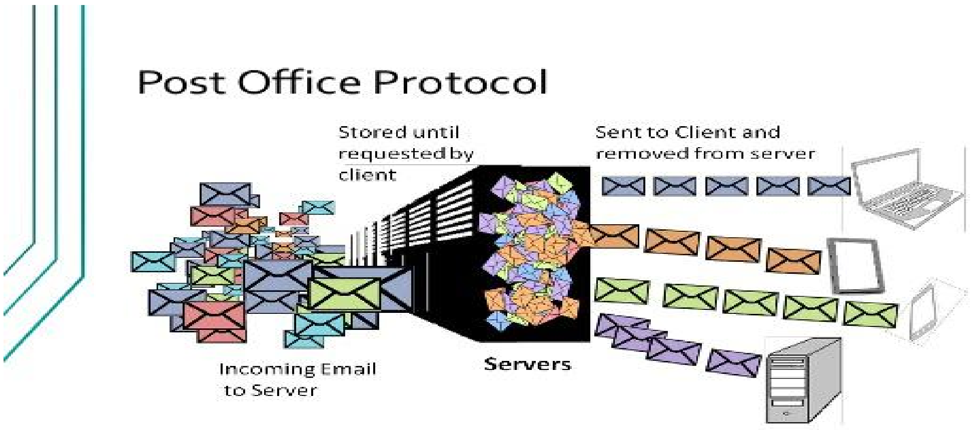
IV. SMTP - (Simple Mail Transfer Protocol)

SMTP is an application that is used to send, receive, and relay outgoing emails between senders and receivers. When an email is sent, it’s transferred over the internet from one server to another using SMTP. In simple terms, an SMTP email is just an email sent using the SMTP server.



V. POP - (Post Office Protocol)

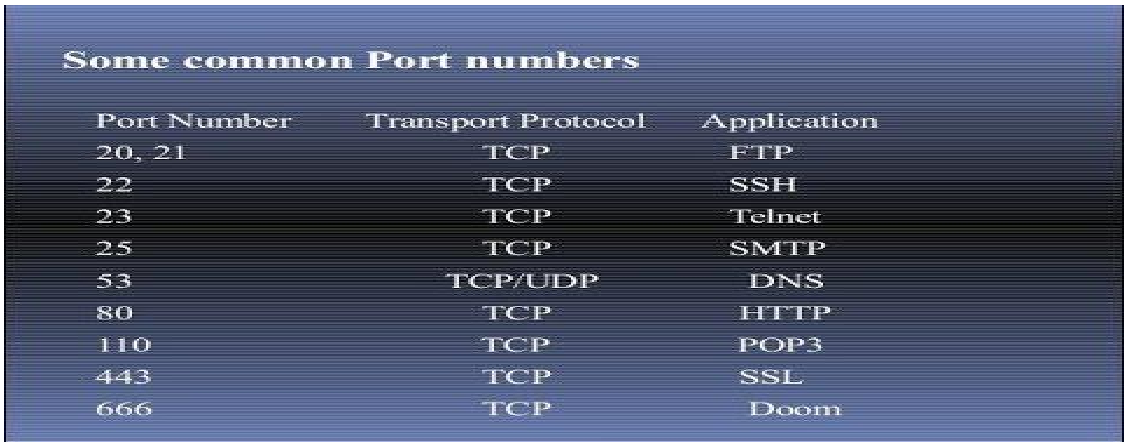
In computing, the POP is an application-layer internet standard protocol used by email client to retrieve email from mail server. POP version 3 is the version in common use.


Post Office Protocol
Stored until requested by client
Sent to Client and removed from server
Incoming Email to Server
Servers

➢ **Port Number**

A port is a number used to uniquely identify a transaction over a network by specifying both the host, and the server. They are necessary to differentiate between many different IP services, such as web service (HTTP), mail service (SMTP), and file transfer (FTP).

A port number is a 16 bit unsigned integer, thus ranging from 0 to 65535.

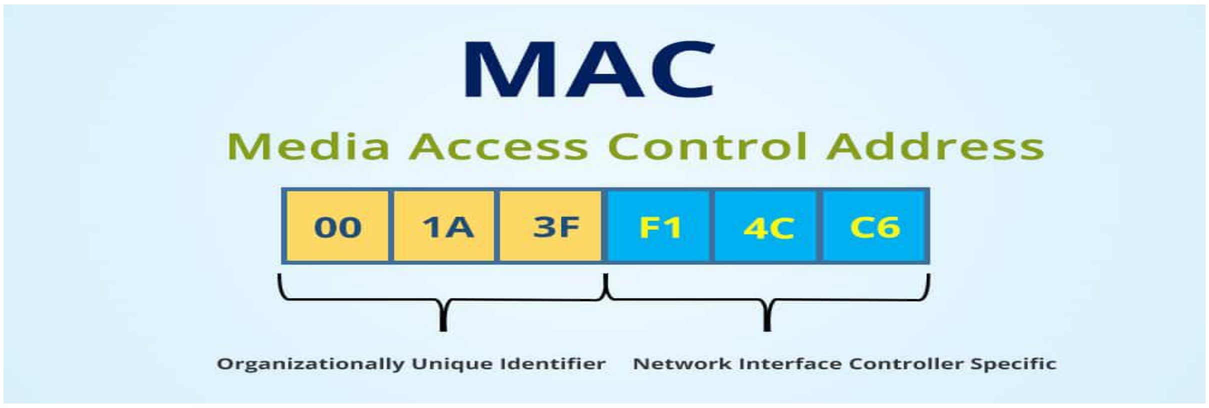


Some common Port numbers

Port Number	Transport Protocol	Application
20, 21	TCP	FTP
22	TCP	SSH
23	TCP	Telnet
25	TCP	SMTP
53	TCP/UDP	DNS
80	TCP	HTTP
110	TCP	POP3
443	TCP	SSL
666	TCP	Doom

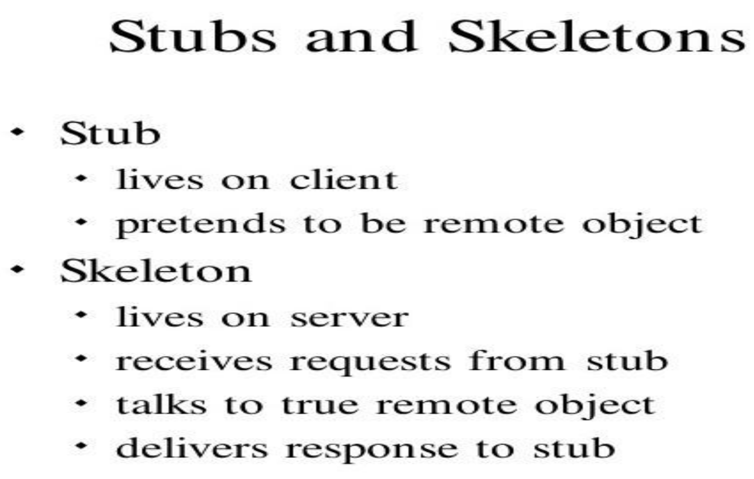
➢ **MAC Address**

MAC Address is a unique identifier assigned to a network interface controller for use as a network address in communications within a network segment. This use is common in most IEEE 802 networking thecnologies, including Ethernet, Wi-Fi and Bluetooth.


MAC
Media Access Control Address
00
1A
3F
F1
4C
C6
Organizationally Unique Identifier
Network Interface Controller Specific

➢ **Skeleton and Stub**

In the java remote method invocation (Java RMI) nomenclature, a **Stub** communicates on the client side with a **Skeleton** in the server side. A class skeleton is an outline od a class that is used in software engineering.



Stubs and Skeletons

- Stub
  - lives on client
  - pretends to be remote object
- Skeleton
  - lives on server
  - receives requests from stub
  - talks to true remote object
  - delivers response to stub

➢ **Connection-oriented and Connection-less Protocol**

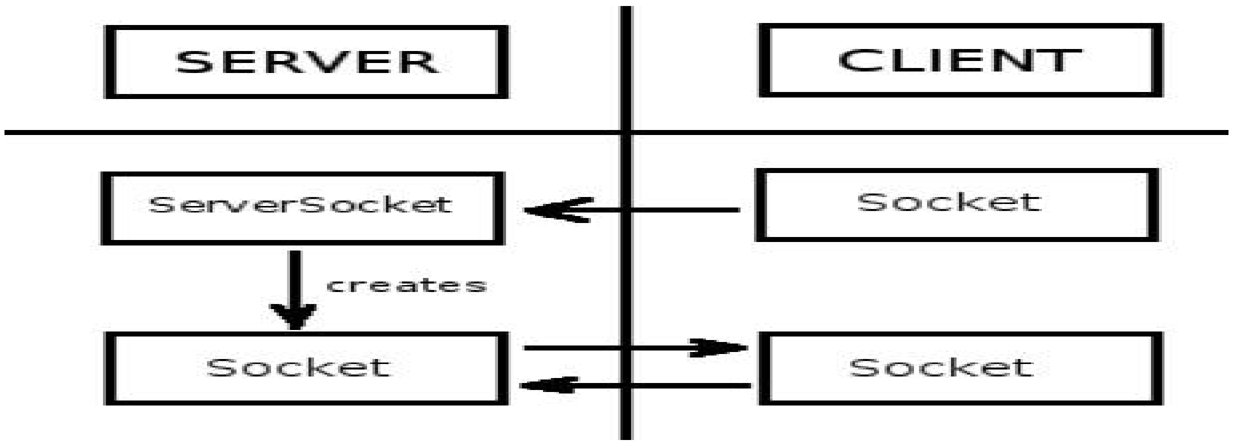
Both Connection-oriented service and Connection-less Protocol service are used for the connection establishment between two or more than two devices. These type of services are offered by network layer.

In Connection Oriented service, Handhake method is used to establish the connection between sender and receiver.

Connection less service is related to the postal system. It does not include any connection establishment and connection termination.

Difference between Connection oriented and Connectionless service

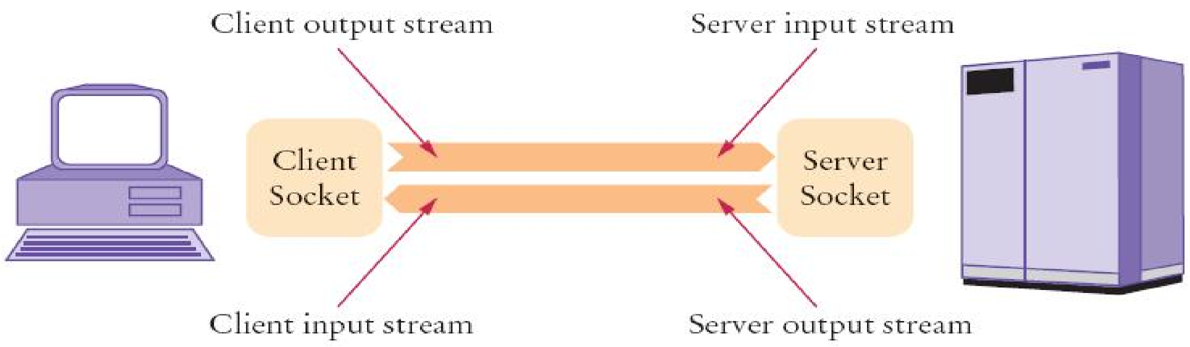
Connection oriented	**Connectionless**
Authentication is needed.	does not need any authentication
Guarantees message delivery	Not guarantees a message delivery
More reliable	Less reliable
Stream based	message based



- **Socket**

What is a Socket?

A socket is one endpoint of a two-way communication link between two programs running on the network. A socket is bound to a port number so that the TCP layer can identify the appilication that data is destined to be send to. An endpoint is a combination of an IP address and a port number.



<u>Socket Class</u>

A socket is simply and endpoint for communications between the machines.

The Socket class can be used to create a scoket.

Important methods of Socket Class

Method	**Description**
public InputStream getInputStream()	returns the InputStream attached with the socket
public OutputStream getOutputStream()	returns the OutputStream attached with socket
public synchronized void close()	closes this socket

<u>Server Class</u>

The SeverSocket class can be used to create a server socket.

This object is used to establish communication with the clients.

Important methods of Server Class

Method	**Description**
public Socket accept()	returns the socket and establish a connection between server and client
public synchronized void close()	closes the server socket

- **Client-side programming & Server-side Programming**

➢ There has mainly two communication Types.

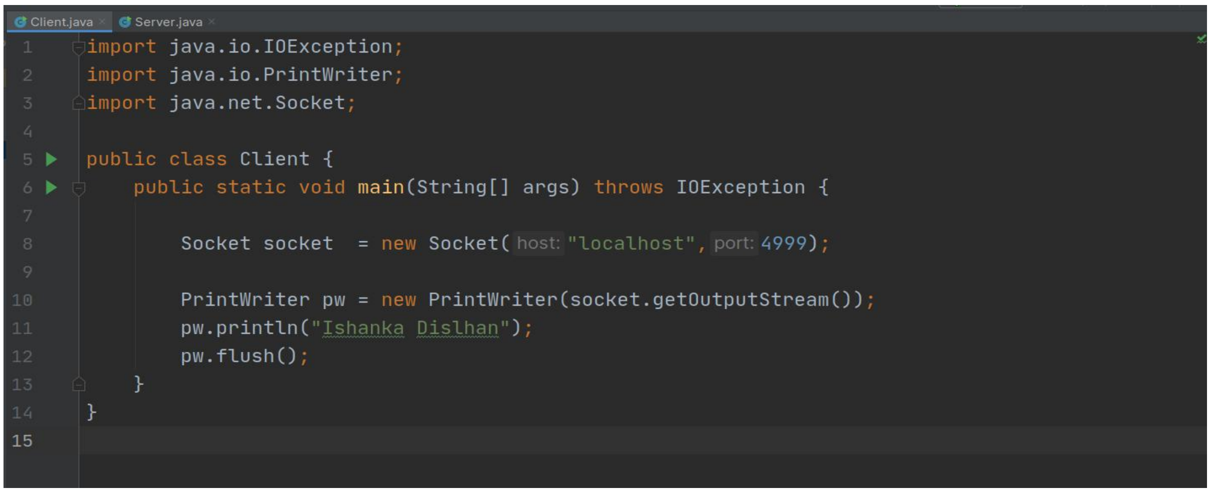
1. One Way Communication (Client to Server)
2. Two Way Communucation (Client to Server and Server to Client)

➢ **Client:** A client is a party that requests pages from the server and displays them to the end-user. In general a client program is a web browser.

➢ **Server:** The Server is responsible for serving the web pages depending on the client/end-user requirement. It can be either static or dynamic.

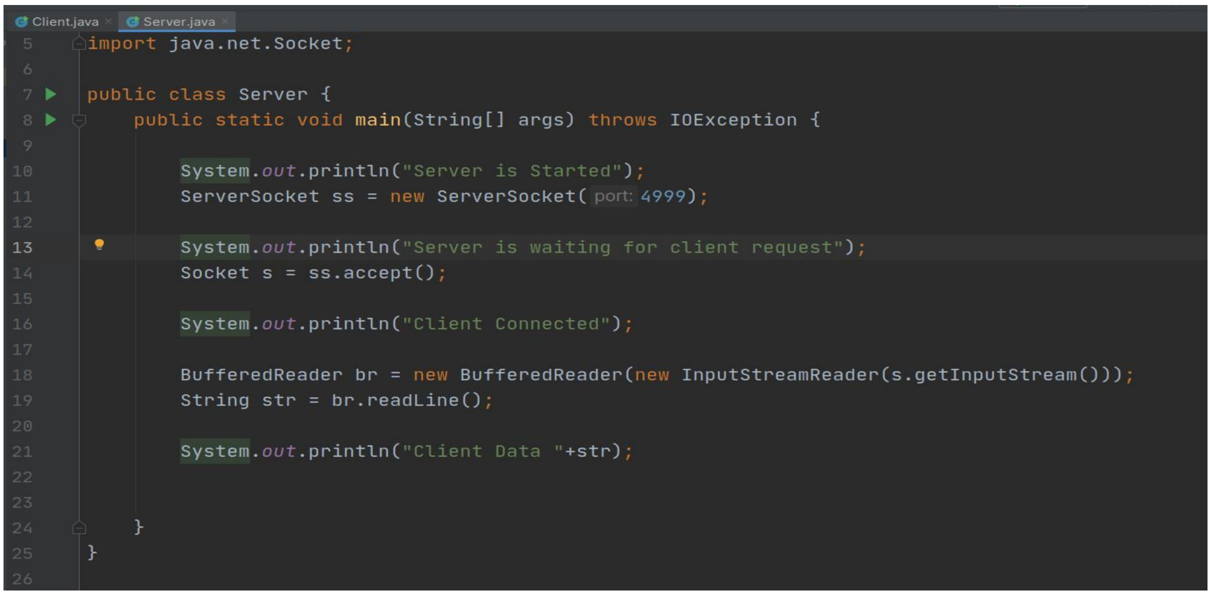
<u>One Way Communication (Client to Server)</u>

Client Side:-



```java
import java.io.IOException;
import java.io.PrintWriter;
import java.net.Socket;

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

        Socket socket  = new Socket( host: "localhost", port: 4999);

        PrintWriter pw = new PrintWriter(socket.getOutputStream());
        pw.println("Ishanka Dislhan");
        pw.flush();
    }
}
```

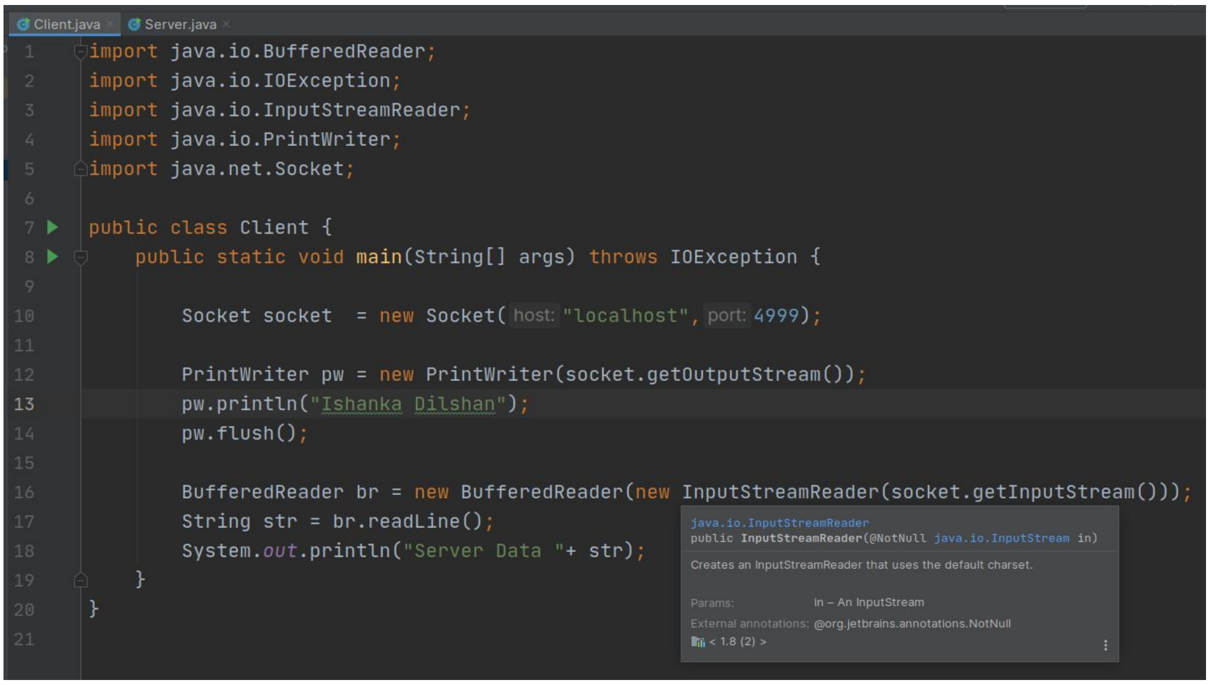
Server Side:-



```java
import java.net.Socket;

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

        System.out.println("Server is Started");
        ServerSocket ss = new ServerSocket( port: 4999);

        System.out.println("Server is waiting for client request");
        Socket s = ss.accept();

        System.out.println("Client Connected");

        BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
        String str = br.readLine();

        System.out.println("Client Data "+str);


    }
}
```

Two Way Communication (Client to Server and Server to Client)

Client Side:-

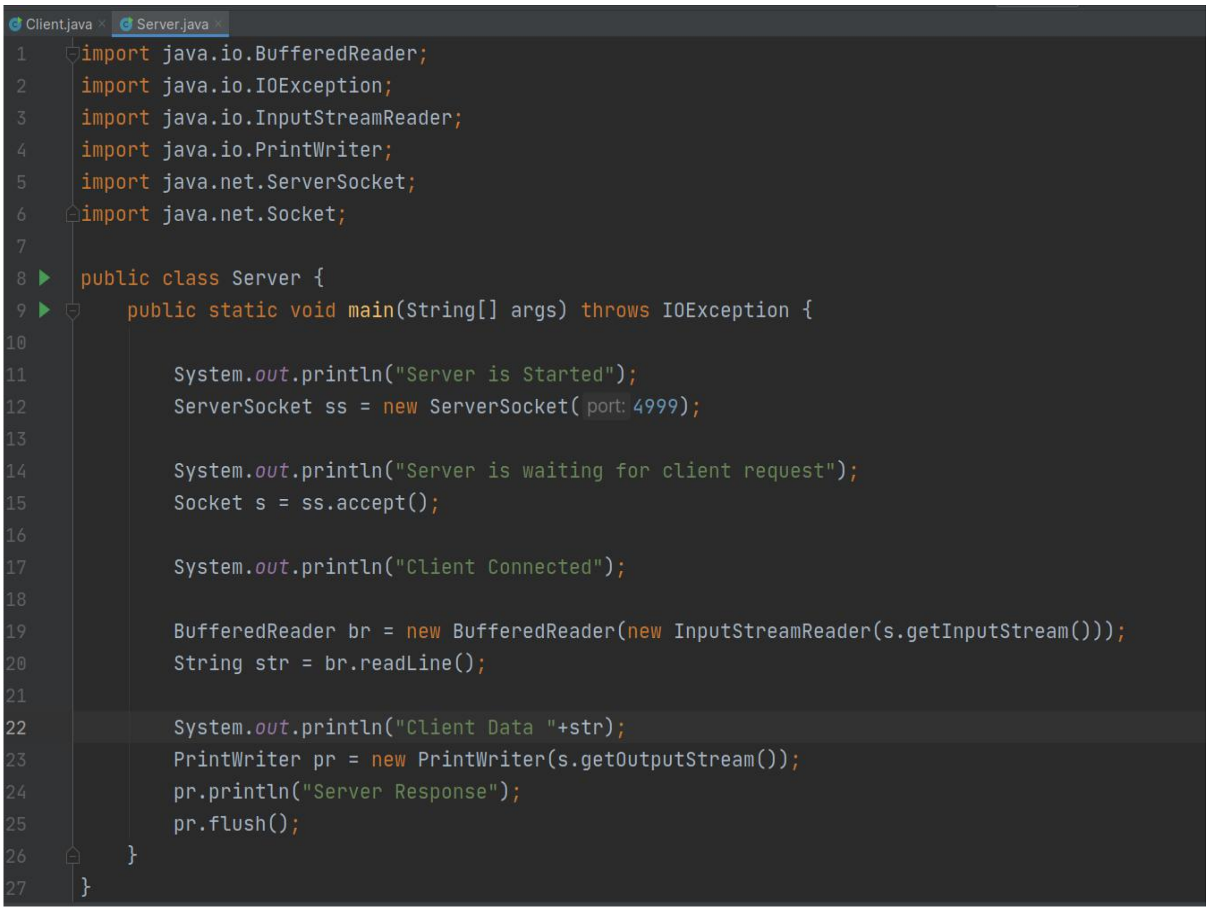


Client.java Server.java

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;

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

        Socket socket  = new Socket( host: "localhost", port: 4999);

        PrintWriter pw = new PrintWriter(socket.getOutputStream());
        pw.println("Ishanka Dilshan");
        pw.flush();

        BufferedReader br = new BufferedReader(new InputStreamReader(socket.getInputStream()));
        String str = br.readLine();
        System.out.println("Server Data "+ str);
    }
}
```

```
java.io.InputStreamReader
public InputStreamReader(@NotNull java.io.InputStream in)
Creates an InputStreamReader that uses the default charset.
Params:              in – An InputStream
External annotations: @org.jetbrains.annotations.NotNull
< 1.8 (2) >
```

Server Side:-

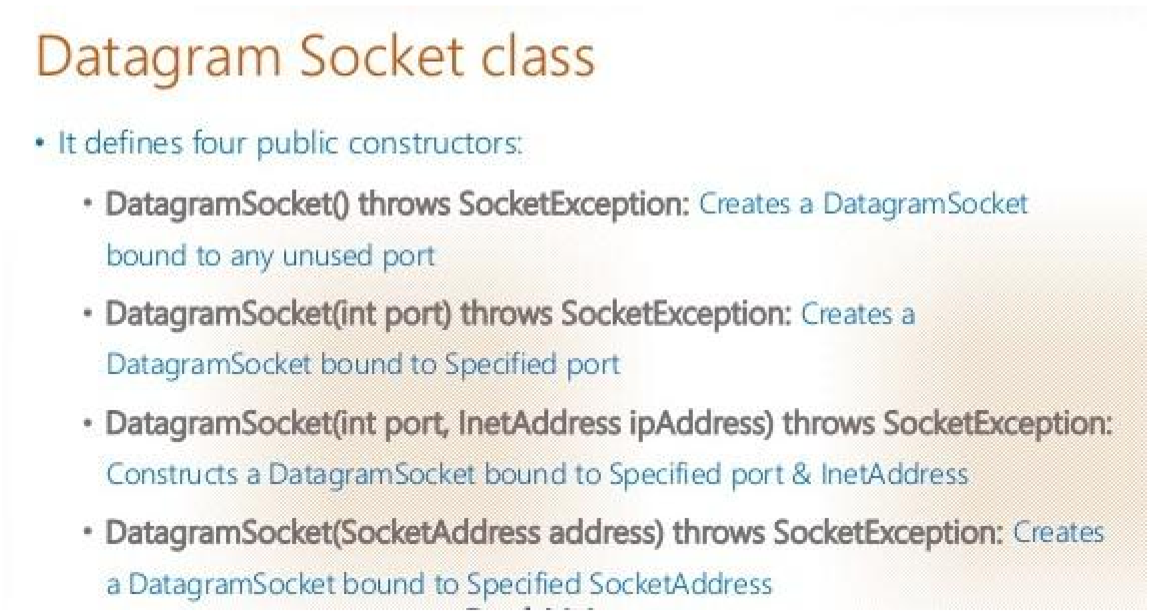


Client.java Server.java

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;

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

        System.out.println("Server is Started");
        ServerSocket ss = new ServerSocket( port: 4999);

        System.out.println("Server is waiting for client request");
        Socket s = ss.accept();

        System.out.println("Client Connected");

        BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
        String str = br.readLine();

        System.out.println("Client Data "+str);
        PrintWriter pr = new PrintWriter(s.getOutputStream());
        pr.println("Server Response");
        pr.flush();
    }
}
```

- **DatagramSocket class**

  Datagram socket is a type of network socket which provides connection-less point for sending and receiving packets. Every packet sent from a datagram socket is individually routed and delivered. It can also be used for sending and receiving brodcast messages. Datagram Socket is the java's mechanism for providing network communication via UDP instead fo TCP.



Datagram Socket class

- It defines four public constructors:
  - **DatagramSocket() throws SocketException:** Creates a DatagramSocket bound to any unused port
  - **DatagramSocket(int port) throws SocketException:** Creates a DatagramSocket bound to Specified port
  - **DatagramSocket(int port, InetAddress ipAddress) throws SocketException:** Constructs a DatagramSocket bound to Specified port & InetAddress
  - **DatagramSocket(SocketAddress address) throws SocketException:** Creates a DatagramSocket bound to Specified SocketAddress