Distributed Network Programming

Prativa Nyaupane

Review

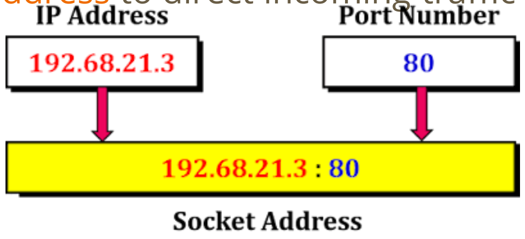
- Ports

Outline

- TCP
- UDP
- IP Address
- Ports
- Socket Programming using TCP and UDP
- Working with URLs and URL Connection Class
- Email Handling using Java Mail API
- Architecture of RMI
- Creating and Executing RMI applications
- Architecture of CORBA
- RMI vs CORBA
- IDL and Simple CORBA Program

Socket

- A socket is one endpoint of a two-way communication link between two programs running on the network.
    - **Port as an Endpoint**: Represents a specific service or application within a machine.
    - **Socket as an Endpoint**: Represents the full address (IP + port) used to uniquely identify and communicate with a specific service on a network.
- It is a programming mechanism that allows a computer to send and receive data over the network.
- To ensure data reaches the intended application on the right computer, sockets are assigned port numbers. The transport layer(consisting of TCP/UDP) uses the combination of port number and IP address to direct incoming traffic to the appropriate program on the specific machine.



```
Socket socket = new Socket(“127.0.0.1”, 5000)
```

Introduction to Socket Programming

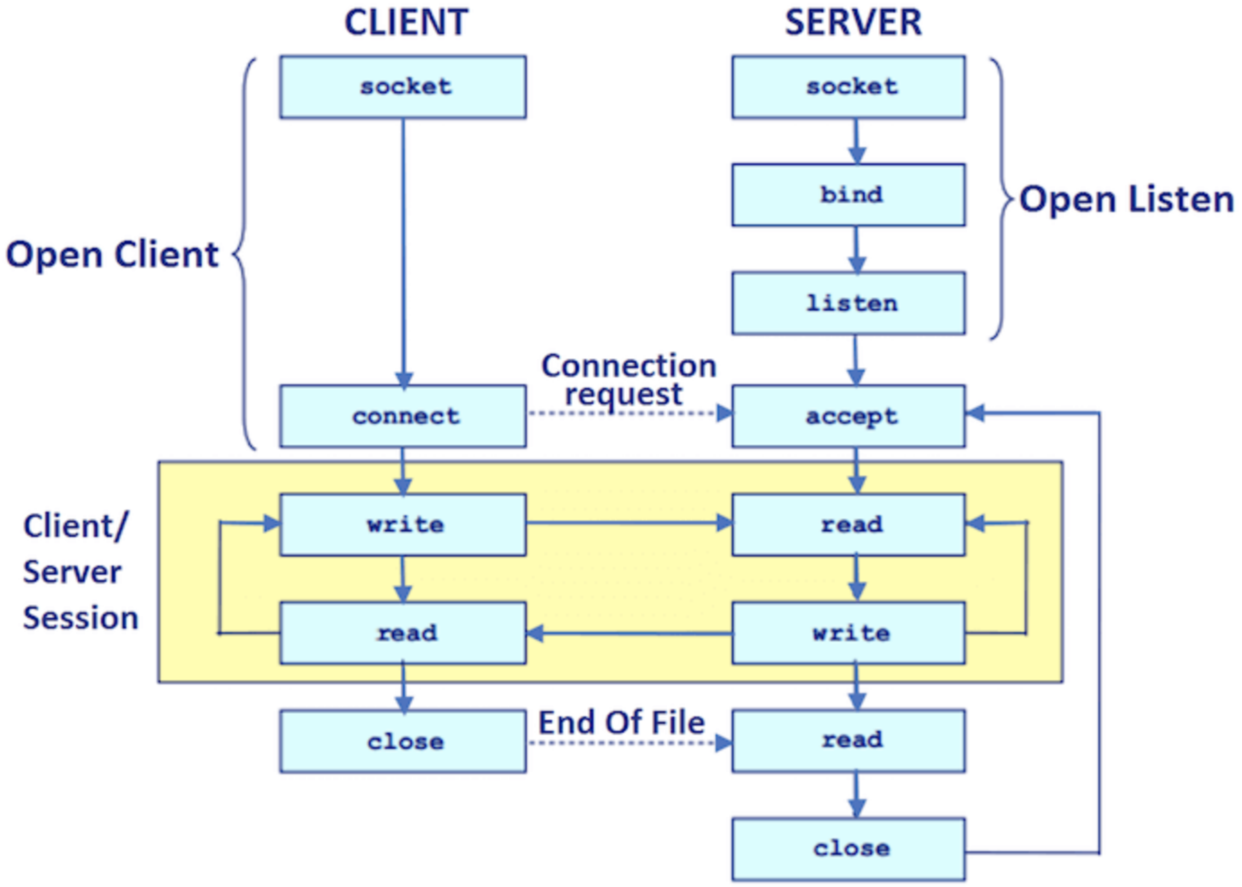
- Socket programming is a building block in network programming that allows communication between two endpoints (typically a client and a server) over a network.
- In Java, the Socket class is used to establish a network connection between a client and a server.
    - Two categories of Sockets:
        - A server socket - It awaits request from a client
        - A client socket - It establishes communication between the client and server.

Socket Programming (contd...)

- The client-side socket is responsible for initiating the connection, while the server-side socket listens for incoming client connections.
- Socket programming supports various communication protocols, such as TCP (Transmission Control Protocol) and UDP (User Datagram Protocol), depending on the requirements of the application.
    - TCP: `Socket socket = new Socket(serverAddress, port); // Connects to the server`
    - UDP: `DatagramSocket socket = new DatagramSocket(); // No connection establishment`
- TCP sockets provide reliable, ordered, and error-checked communication, making them suitable for applications that require guaranteed data delivery.
- UDP sockets provide a connectionless and unreliable communication mechanism, which is useful for real-time applications or situations where speed is prioritized over reliability.

Socket Programming (contd...)

- Socket programming allows developers to build network applications such as chat systems, file transfer protocols, remote control applications, and more.
- Socket programming allows for bidirectional communication between the client and the server. Clients can send requests to the server, and the server can respond with the requested data or perform actions based on the client's request.
- The client-side socket initiates the connection by specifying the server's IP address and port number to establish a communication channel.
- The server-side socket listens on a specific port for incoming client connections. Once a client connects, the server accepts the connection and can communicate with the client.


CLIENT
SERVER
socket
socket
bind
Open Listen
listen
Open Client
Connection request
connect
accept
Client/ Server Session
write
read
read
write
close
End Of File
read
close

SOCKET API