Module 16

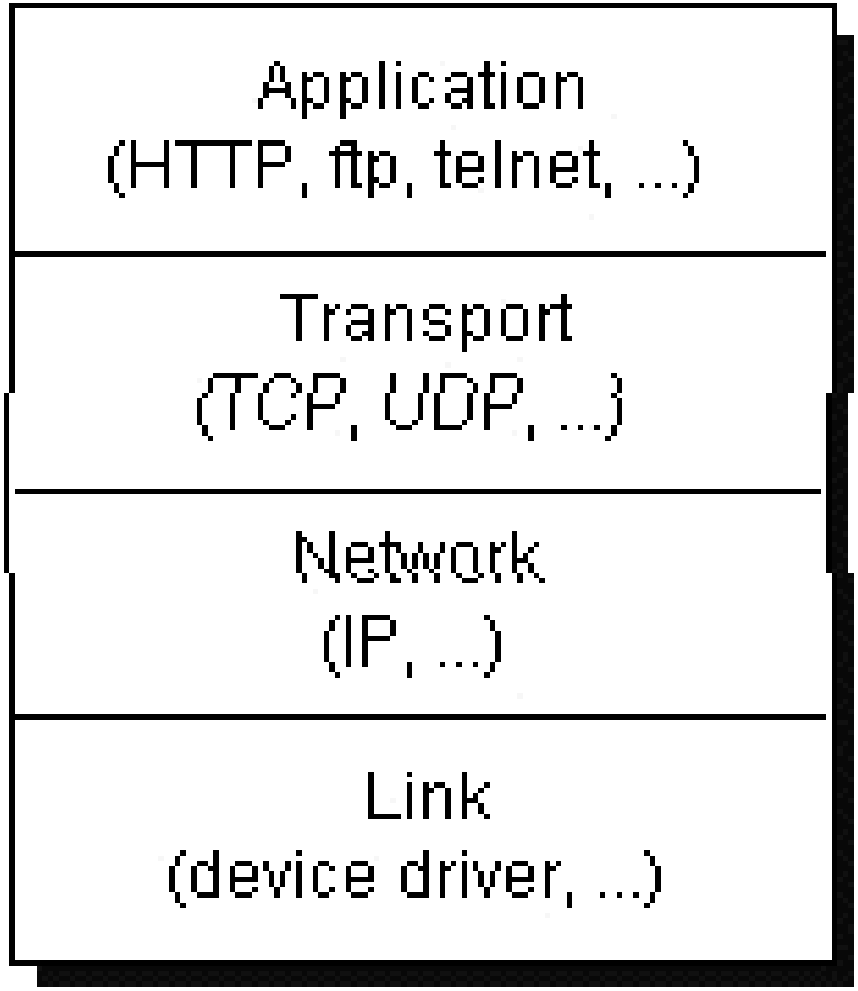
Networking

Objectives

- Understand basic concepts of computer network: Protocols, TCP, UDP, sockets, address and ports
- Classes in java.net package
- Develop code to set up the network connection
- Use classes in java.net package for implementation of TCP or UDP clients and servers
- Use threads in the server to support multi-clients

Networking Basics

- Computers running on the Internet communicate to each other using either the Transmission Control Protocol (TCP) or the User Datagram Protocol (UDP)



TCP & UDP

- *TCP* (*Transmission Control Protocol*) is a connection-based protocol that provides a reliable flow of data between two computers.
- *UDP* (*User Datagram Protocol*) is a protocol that sends independent packets of data, called datagrams, from one computer to another with no guarantees about arrival. UDP is not connection-based like TCP.
- Java programs - at the application layer
  - can use the classes in the java.net package

java.net

- Low Level API
  - Addresses, which are networking identifiers, like IP addresses.
  - Sockets, which are basic bidirectional data communication mechanisms.
  - Interfaces, which describe network interfaces.
- High Level API
  - URIs, which represent Universal Resource Identifiers.
  - URLs, which represent Universal Resource Locators.
  - Connections, which represents connections to the resource pointed to by URLs.

java.net.NetworkInterface

- public String getName()
- public String getDisplayName()
- public List<InterfaceAddress> getInterfaceAddresses()
- public Enumeration<InetAddress> getInetAddresses()
- public static Enumeration<NetworkInterface> getNetworkInterfaces()
- public static NetworkInterface getByName(String name)
- public static NetworkInterface getByInetAddress(InetAddress addr)
- public byte[] getHardwareAddress()
- **ListNIFs.java**

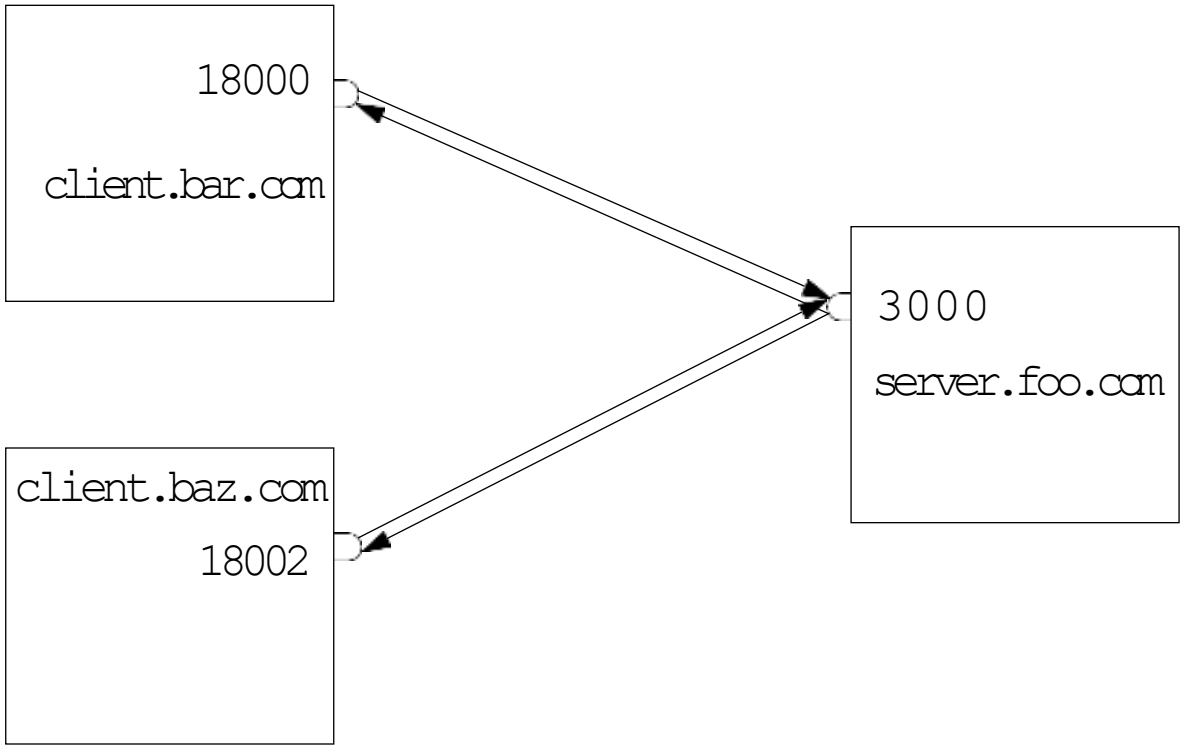
java.net.InetAddress

- public String getHostName()
- public String getHostAddress()
- public static InetAddress getLocalHost()
- public static InetAddress getByName(String host)
- public static InetAddress getByAddress(byte[] addr)
- **testIPAddr.java**

Networking With JavaTechnology

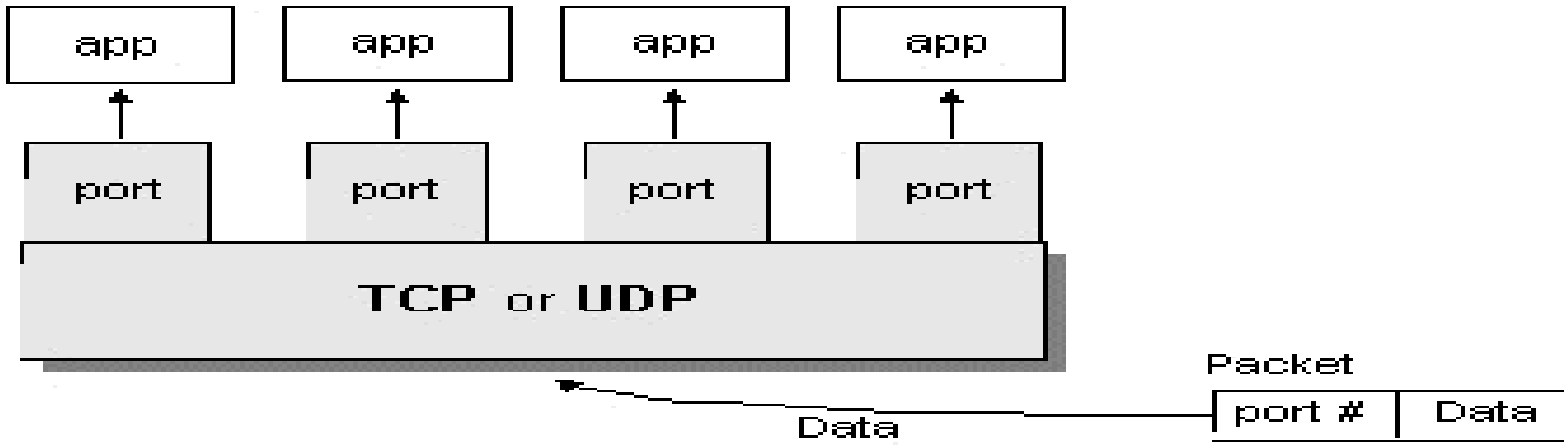
- To address the connection, include the following:
  - The address or name of remote machine
  - A port number to identify the purpose at the server
- Port numbers range from 0–65535.

Networking



Ports

- computer is identified by its 32-bit IP address
- Ports are identified by a 16-bit number, which TCP and UDP use to deliver the data to the right application
  - The TCP and UDP protocols use ports to map incoming data to a particular process running on a computer



Sockets

- Sockets are means to establish a communication link between machines over the network.
- Sockets hold two streams: an input stream and an output stream.
- Each end of the socket has a pair of streams.
- java.net package provides 4 kinds of Sockets:
  - *Socket* is a TCP client API, connect to a remote host.
  - *ServerSocket* is a TCP server API, accept connections from client sockets.
  - *DatagramSocket* is a UDP endpoint API, used to send and receive datagram packets.
  - *MulticastSocket* is a subclass of DatagramSocket, dealing with multicast groups

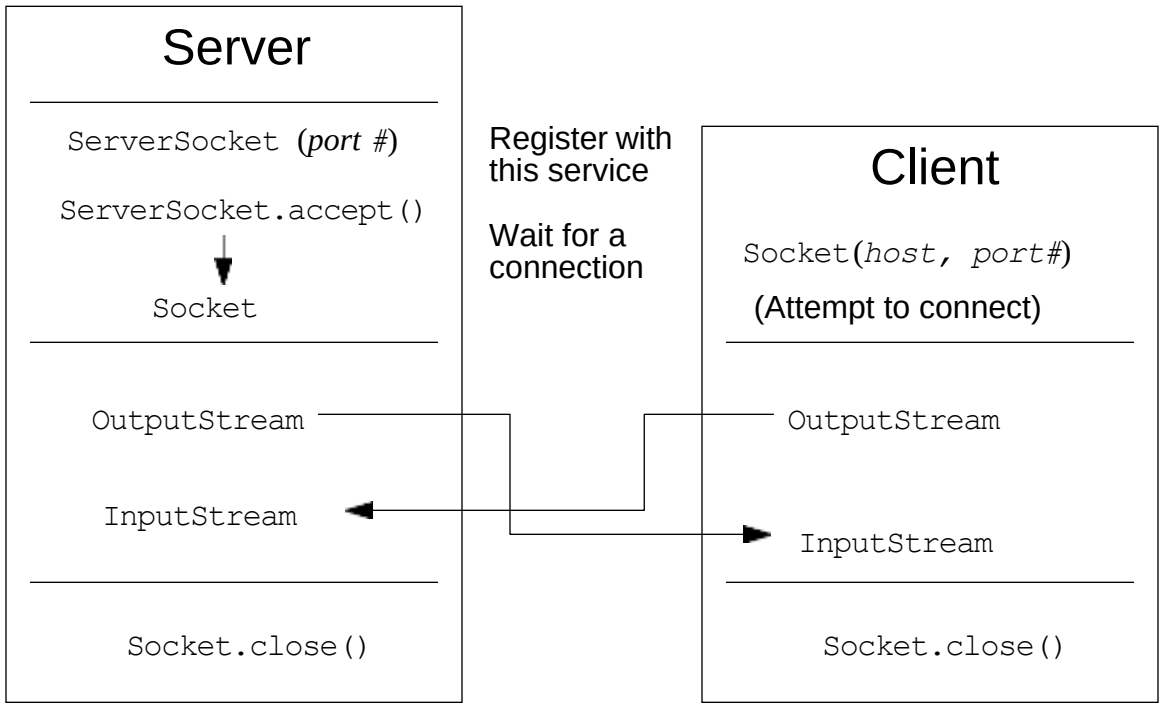
Networking Classes in the JDK

- java.net
- TCP
  - The URL, URLConnection, Socket, and ServerSocket classes
- UDP
  - The DatagramPacket, DatagramSocket, and MulticastSocket classes

Socket

- A server runs on a specific computer and has a socket that is bound to a specific port number.
  - waits, listening to the socket for a client to make a connection request
- A client
  - Open a socket.
  - Open an input stream and output stream to the socket.
  - Read from and write to the stream according to the server's protocol.
  - Close the streams.
  - Close the socket
- TCPServer.java TCPClient.java

Java Networking Model



Supporting Multiple Clients

```
while (true) {
        accept a connection ;
        create a thread to deal with the client ;
}
```

- MultiTCPServer.java TCPClient.java

Datagram

- A *datagram* is an independent, self-contained message sent over the network whose arrival, arrival time, and content are not guaranteed.
  - DatagramSocket, DatagramPacket, MulticastSocket
  - An application can send and receive DatagramPackets through a DatagramSocket.
  - DatagramPackets can be broadcast to multiple recipients all listening to a MulticastSocket
  - **UDPServer.java UDPClient.java**

Broadcasting to Multiple Recipients*

- **MulticastServer**
  - InetAddress group = InetAddress.g tByName("230.0.0.1 "); //使用InetAddress类创建多播组地址
  - MulticastSocket s = new MulticastSocket(2020);//用MulticastSocket类创建一个多播套接字
  - DatagramPacket sPackage = new DatagramPacket(buf, length, group, port);//数据报文包
  - s.send(sPackage);//发送多播包
  - **MulticastServer.java**

Broadcasting to Multiple Recipients*

- **MulticastClient**
  - InetAddress group = InetAddress.getByName(230.0.0.1);
  - MulticastSocket s = new MulticastSocket(2020);
  - s.joinGroup(group);//加入多播组
  - DatagramPacket recv = new DatagramPacket(buf, buf.length);
  - s.receive(recv);//接收
  - **MulticastClient.java**

Summary

- Networking Basic Concepts
- java.net - NetworkInterface, InetAddress
- TCP & UDP
- Addresses, Ports, Sockets
- TCP Classes - Socket, ServerSocket classes
- UDP Classes - DatagramPacket, DatagramSocket
- Supporting Multiple TCP Clients
- Broadcasting to Multiple UDP Recipients*