1. Serialization
2. Network

Link: <https://www.slideshare.net/tusharkute/network-programming-in-java>

Link: <https://slideplayer.com/slide/5150902/>

Client <-> Network <-> Server

**Socket**

Sockets provide an interface for programming networks at the transport layer-> Network communication using Socket as I/O.

Socket is endpoint for communication between two machines.

Socket-based communication can communicate on program in Java or Non-Java.

Socket uses TCP to communicate over the network

**Constructor:**

Socket(String remoteHost, int remotePort)

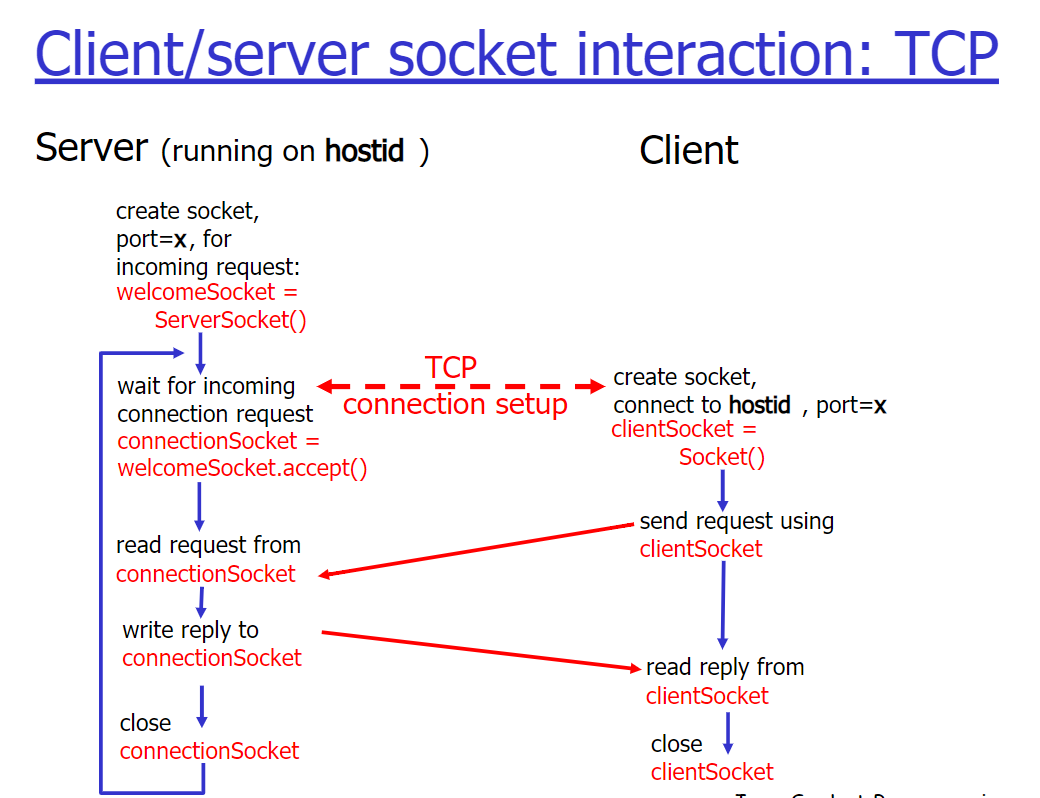
Socket(InetAddress ip, int remotePort)

**TCP-Transmission Control Protocol**

TCP provides a reliable flow of data between 2 computers (point-to-point).

Ex: HTTP,FTP, Telnet require a reliable communication channel.

The *URL,URLConnection, Socket, ServerSocket* classes all use Transmission Control Protocol (TCP) to communicate over the network

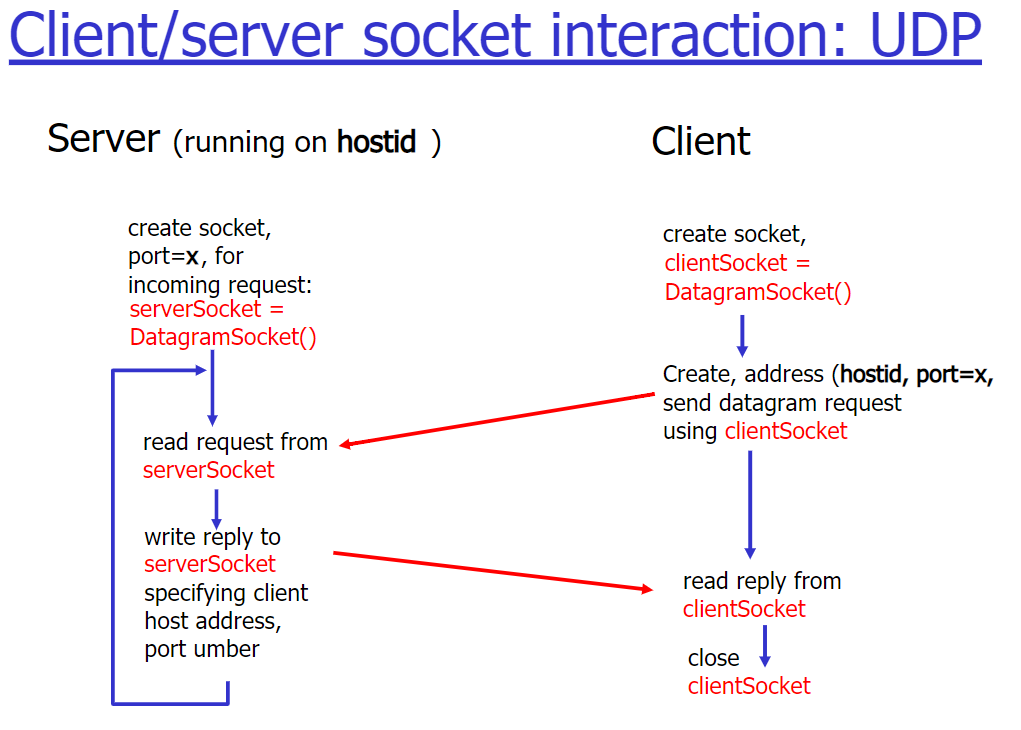


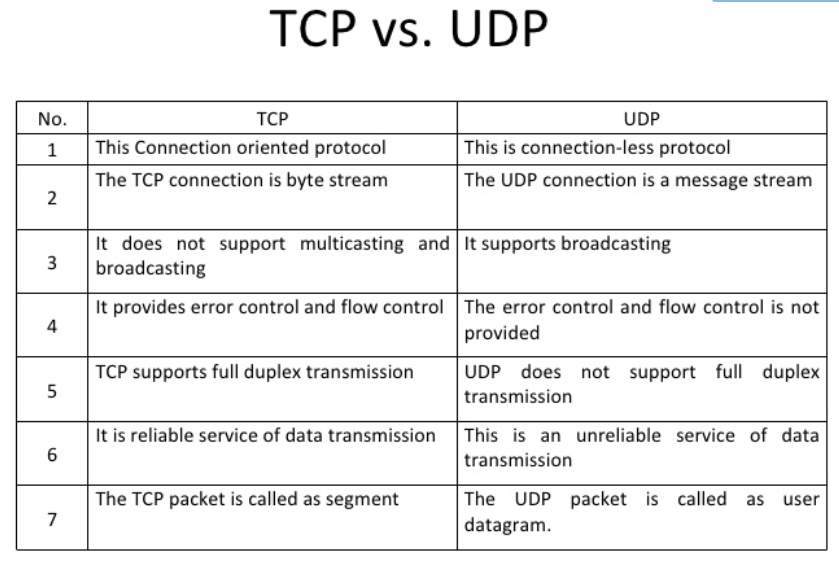
**UDP-User Datagram Protocol**

UDP sends independent packets of data (called datagrams) from one computer to another with no guarantees about arrival -> not reliable, but good in speed and cost

Ex: streaming media, games, Internet telephony…

The *DatagramPacket, DatagramSocket, MulticastSocket* classes are for use with User Datagram Protocol (UDP)





Port # IP

**Port**:

* TCP and UDP use Ports to deliver the data to the right application
* 16 bit integer value (2^16), 0 – 1023 (well-known ports) to 65535
* FTP (20,21); TELNET (23); SMTP (25); POP3 (110); HTTP (80); DSN(53)

**IP**:

* IP as address 32bit
* IP uses to deliver data to the right computer on the network
* Java.net.InetAddress-> both IP address and domain name

1. Thread