AHRAM CANADIAN UNIVERSITY

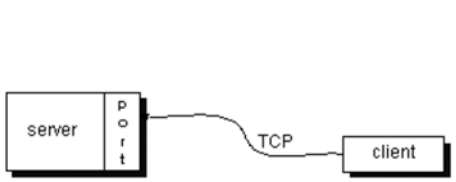
Session 04.02
Network Programming

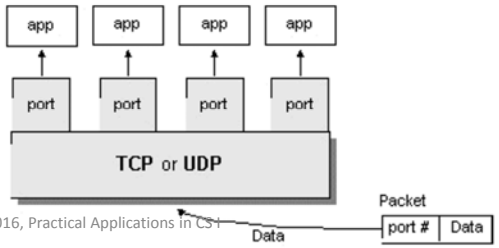
Dr. Ahmed M. ElShafee

Concepts of socket programming

Ports

- Ports are a virtual channels that enables applications to share the single network channel to exchange data.
- Port numbers are represented by 16-bit numbers. (0 to 65,535) The port numbers ranging from 0 - 1023 reserved for use by well known application
- services such as HTTP and FTP and other system services.





Sockets

You can reach required service via its network and port IDs. what then?

a) If you are a client

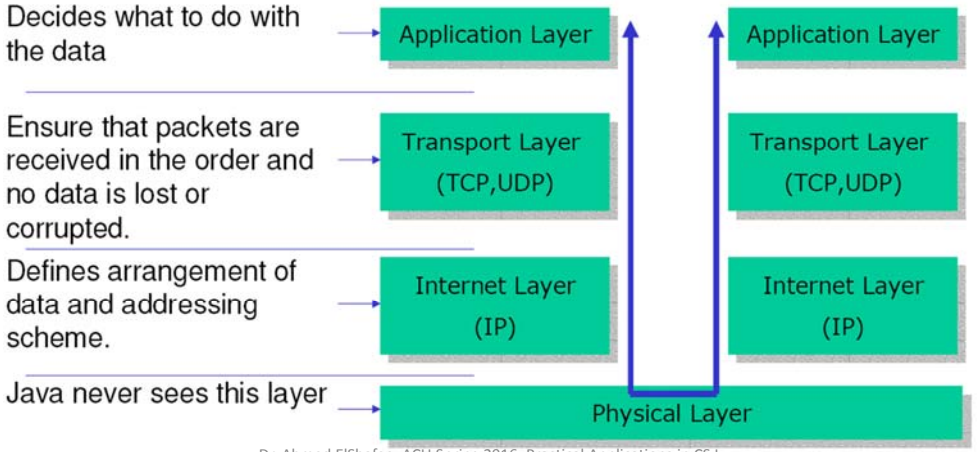
- you need an API that will allow you to send messages to that service and read replies from it

b) If you are a server

- you need to be able to create a port and listen at it.
- you need to be able to read the message comes in and reply to it.

The **Socket** and **ServerSocket** are the Java client and server classes to do this.

Network Layers

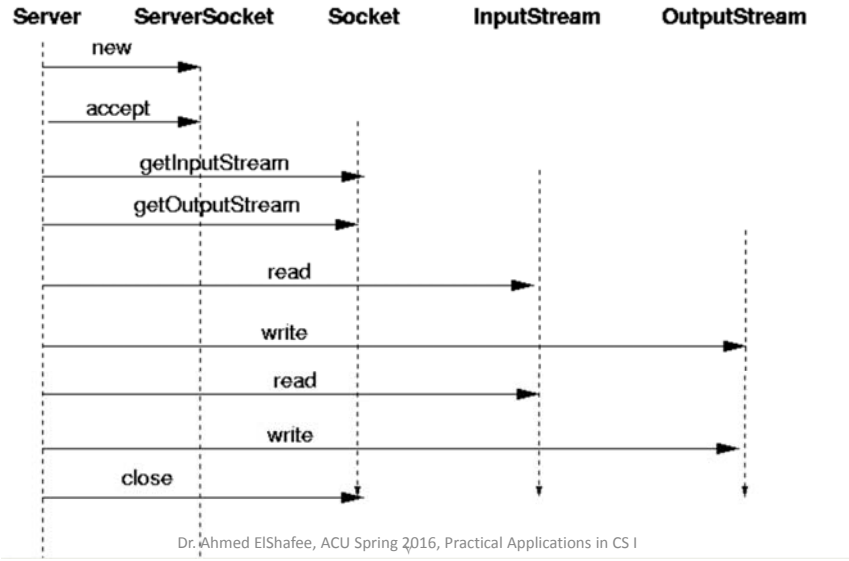


TCP and UDP

Java only supports TCP (Transmission Control Protocol), UDP (User Datagram Protocol) and application layer protocols built on top of these.

TCP/IP client/server

TCP Server



Simple Server example

function	code
Create server socket	s = new ServerSocket(int PORT);
Accepts incoming connection	Socket incoming = = s.accept();
Create reader object form accepted connection object as a data source	BufferedReader reader = new BufferedReader(new InputStreamReader(incoming.getInputStream()));
Create print stream object to write data to socket as a data source	PrintStream out = new PrintStream(incoming.getOutputStream());
Read line from socket	String str = reader.readLine();
Write line to socket	out.println(str);

Simple client socket

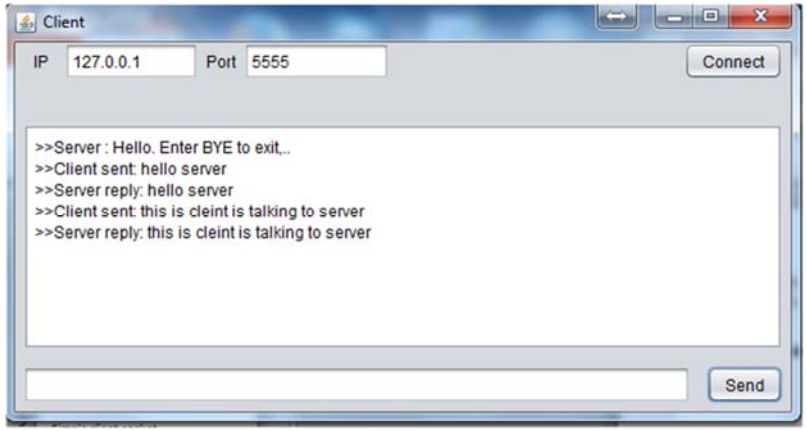
function	code
Create remote socket	echoSocket = new Socket(IP, port);
Create print writer object to write chars to remote socket	PrintWriter out = new PrintWriter(echoSocket.getOutputStream(), true);
Create buffered reader object to read chars from local socket	BufferedReader in = new BufferedReader(new InputStreamReader(echoSocket.getInputStream()));
Write line to remote socket	out.println(”line”);
Read line from local socket	String str=in.readLine()

Server

Check lab manual

Client

Check lab manual



Thanks,..

See you next week (ISA),…