

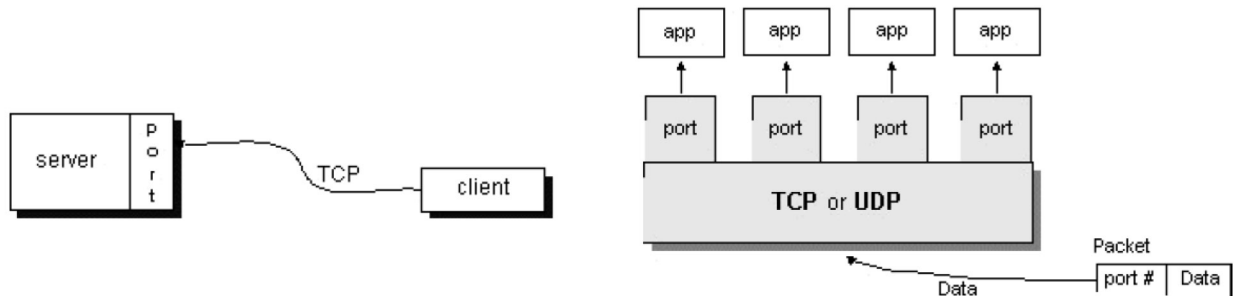
Network I Lab 05

Sockets and 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

Decides what to do with the data

Application Layer

Application Layer

Ensure that packets are received in the order and no data is lost or corrupted.

Transport Layer
(TCP,UDP)

Transport Layer
(TCP,UDP)

Defines arrangement of data and addressing scheme.

Internet Layer
(IP)

Internet Layer
(IP)

Java never sees this layer

Physical Layer

TCP/IP client/server



Server

```
public static int port = 8189;
ServerSocket s = new ServerSocket(port);
Socket incoming = s.accept();
incoming.setSoTimeout(10000);
BufferedReader reader = new BufferedReader(new
InputStreamReader(    incoming.getInputStream()));
PrintStream out = new PrintStream(incoming.getOutputStream());
out.println(">>Server sent: Hello. Enter BYE to exit,..");
System.out.println(">> Host is now connected,..");
while (true)
{
    String str = reader.readLine();
    out.println(str);
    System.out.println(">> Host sent: "+str);
    if (str.equals("BYE"))    break;
}
System.out.println(">> Host is now dis-connected,..");
incoming.close();
s.close();
```

Client	
--------	--

	<pre>Socket echoSocket = new Socket("127.0.0.1",8189); BufferedReader stdIn = new BufferedReader(new InputStreamReader(System.in));PrintWriter out = new PrintWriter(echoSocket.getOutputStream(), true); String userInput; BufferedReader in = new BufferedReader(new InputStreamReader(echoSocket.getInputStream())); System.out.println (in.readLine()); While(true) { userInput = stdIn.readLine(); out.println(userInput); if (userInput.equals("BYE")) break; } out.close(); in.close(); stdIn.close(); echoSocket.close();</pre>
--	--