JAVA Network API

To be discussed

1 -  java.net ................................................................. 1
2 -  Connection-Oriented vs. Connectionless Communication ................................. 1
3 -  Connectionless: ......................................................................................... 1
4 -  Networking Protocols ................................................................................. 2
5 -  Sockets ......................................................................................................... 2
6 -  Multicast Addressing..................................................................................... 2
7 -  Internet Addressing ...................................................................................... 3
8 -  InetAddress Class .......................................................................................... 3
9 -  Connection-Oriented Network Communication ................................................. 4
10 - Connectionless Communication .................................................................... 5
11 - Connecting to an SMTP Mail Server .............................................................. 5

1 -  java.net

Set of classes that support network programming using communication protocols of Internet

Internet Protocol (IP)

Transport Control Protocol (TCP)

User Datagram Protocol (UDP)

Communication – Internet Protocol (IP) packets

2 -  Connection-Oriented vs. Connectionless Communication

Connection-Oriented: Communication link btw source port/IP address and a destination port/IP address (Telephone conversation)

Stream-based communication btw client and server
TCP – FTP, SMTP, HTTP
Connection continues until the link is broken
Error detection and error correction
Reliability of the communication

3 -  Connectionless:

It does not establish a link for the duration of the connection
Postal mail
UDP – Time Protocol
On the datagram, destination port and IP address are written.
4 - Networking Protocols

Data communication language.

IP - Internet Protocol: Datagram protocol, transmitted packets of information are not guaranteed to be delivered. No stream of related packets! Connectionless ~ Postal


UDP - User Datagram Protocol: Connectionless protocol

<table>
<thead>
<tr>
<th>Protocol</th>
<th>Port</th>
<th>Encoding</th>
<th>Expl.</th>
</tr>
</thead>
<tbody>
<tr>
<td>Echo</td>
<td>7</td>
<td>Tcp/udp</td>
<td></td>
</tr>
<tr>
<td>Daytime</td>
<td>13</td>
<td>Tcp/udp</td>
<td>ASCII representation of current time on server</td>
</tr>
<tr>
<td>ftp</td>
<td>20</td>
<td>Tcp</td>
<td>Transfer files</td>
</tr>
<tr>
<td>telnet</td>
<td>23</td>
<td>Tcp</td>
<td>Interactive remote command-line communication</td>
</tr>
<tr>
<td>Smtp</td>
<td>25</td>
<td>tcp</td>
<td>“Simple mail transfer protocol”</td>
</tr>
<tr>
<td>Pop3</td>
<td>110</td>
<td>Tcp</td>
<td>Transfer of accumulated email from the host to clients</td>
</tr>
<tr>
<td>http</td>
<td>80</td>
<td>Tcp</td>
<td>Hypertext transfer protocol</td>
</tr>
</tbody>
</table>

5 - Sockets

Socket: The ends of communication link between processes.

Java communication over a network: Streams model.

Socket: Two streams - Input stream and Output Stream

Sending data: A process sends data to the other via writing to the output stream associated with the socket.

Receiving data: A process receives data from the other via reading from the input stream associated with the socket.

6 - Multicast Addressing
Unicast – Packets are sent from a source host to a destination in a point-to-point fashion.

Send IP packets to multiple destinations - Multicast

Host group – Using a single destination IP address.

Setting up a connection

Telephone analogy:

One machine must be running a program that is waiting for a connection.
The other machine must reach the first.
The machine must be online - Connected to a live TCP/IP network.

7 - Internet Addressing

Full domain name lookups via the Domain Name System DNS - global distributed telephone book.
Addressing the connection

Address or the name of the remote machine + a port number + identification of the purpose.

Port numbers: 16-bit numbers and are in the 0-65535 range.

Port numbers below 1024 are reserved for predefined services - telnet, SMTP mail, ftp.
The port numbers by the local and remote machine must agree.

8 - InetAddress Class

Java: 32-Bit IP address through InetAddress class - Convert a textual Internet address of the form host.Subdomain.domain into an object representing that address.

No constructor!

Rather, static class methods getByName() and getAllByName() to create instances of InetAddress.

To get the Internet address of the localhost, InetAddress.getLocalHost()
TCP/IP socket connections are implemented in java.net package.

Server: Assigns a port number.

Client requests a connection: Server opens the socket connection with the accept() method.

Client: Establishes a connection with host on port.
10 - Connectionless Communication

UDP (User Datagram Protocol) - Connectionless protocol.

Two Java classes support UDP - DatagramSocket and DatagramPacket.

DatagramPacket has two constructors - One for receiving data and one for sending data.

DatagramPacket(byte[] recvBuf, int readLength) - Used to set up a byte array to receive a UDP packet. The byte array is empty when passed to the constructor and the int variable is set to the number of bytes to read.

DatagramPacket(byte[] sendBuf, int sendLenght, InetAddress iaddr, int iport) - Used to set up a UDP packet for transmission.

DatagramSocket - Used to read and write UDP packets. There are three constructors that allow you to specify which port and Internet address to bind.

DatagramSocket() - Bind to available port on local host.
DatagramSocket(int port)
DatagramSocket(int port, InetAddress iaddr)

11 - Connecting to an SMTP Mail Server

Email to a machine - Port number 25

Once admitted, use SMTP communication protocol to talk with the machine.

The SMTP server sends an initial identification string.

You reply by telling which machine you are sending from

If ok, the server acknowledges.

You reply by giving it your email address: From:sender@host.subdomain.domain

If ok, server acknowledges.

You reply by giving it the To: receiver@remotehost.subdomain.domain

If ok, server acknowledges.

You send the entire email msg, line by line and end the msg with a single line containing .

If ok, server acknowledges.
You sign off by sending QUIT.