Network Programming – Client/Server 1

- The key package is java.net which provides support for client/server development.
- The classes are in three categories:

1. Web Classes: enable access to Web docs via their URLs. These classes are URL and URLConnection
2. Raw Classes: Socket, ServerSocket, DatagramSocket, InetAddress. These provide stream-based interfaces to hosts on the Internet. They enable you to develop protocols, connect to existing files and develop client/server applications.
3. Extension Classes: ContentHandler, URLStreamHandler. They can be used to extend the Web classes to cater for new document and content formats. Use if you need to handle a new graphics standard and write classes to interface to the existing Web classes.

Network Information

- Java allows developers to create networks connections in several ways. This allows Applets to be much more flexible than HTML for example.
- Whilst simple Applets are self-contained, more complex applications will rely on using external servers.
- In addition, subject to security restrictions, Java code can connect to any arbitrary server. The Java libraries don't limit the user to using only HTTP. Any IP can be used such as FTP.
- Java is therefore often used as a gateway to databases, proxy servers, network game handlers and mail servers.
- Java has libraries to help with these applications. An example is the ODBC library to help with database connectivity.
- We do not have time to look at these classes, but if you do get involved in commercial programming you will need to!

Port Numbers

- For a client to access a server code it needs to know the port number you are using.
- There is a universal set of port numbers. Those less than 1024 are usually already allocated.
- The danger of selecting a random port number is that if you move the code to another server the port could be already allocated.
- It is normal to pass the port number into a constructor.

Socket Classes

- Sockets are used to establish a stream-based i/p and o/p between server and client computers.
- A socket is a logical concept; it is a connection to a host on the Internet and consists of an IP address and a Port number. A socket object represents the Java version of a TCP connection.
- A ServerSocket object is constructed on a server and enables the server to listen for clients. It establishes a connection when it receives an incoming connection message from a client.
- ServerSocket has three constructors which specify the port to bind to, the queue length for incoming connections, and the IP address to bind to as:

```
ServerSocket(int) throws IOException
ServerSocket(int, int) throws …
ServerSocket(int, int, InetAddress) throws …
```

Outline Server Connection

```
try {
   // set up a server on this port num
   sock = new ServerSocket(port);
   System.out.println("Started on port" + port);
   conn = sock.accept();
   /* returns with a socket from a client which is
      trying to connect to the server. */
}
catch (Exception e){
   System.out.println("Error:" + e);
   System.exit(1);
   /* A 0 parameter closes the program.
      Non-zero allows error codes to be passed. */
}
```

Client Connection

- Socket objects are created by the client. The most popular constructor is:

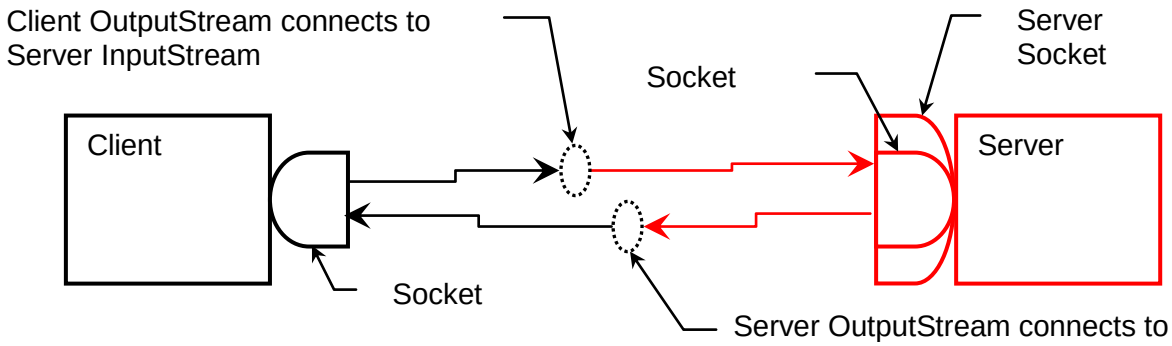
```
Socket (String x , int y)
// x: address of the server e.g. MyServer.cov.ac.uk
// y: port number
```

- Communication between clients and a server is achieved by I/O streams. A socket is associated with an i/p stream and an o/p stream. The methods to obtain these streams are:

```
InputStream ips = sock.getInputStream();
OuputStream ops = sock.getOuputStream();
```

These streams can then be used to send and receive data.

A Possible Schematic



PrintWriter inserts data on OutputStream

BufferedReader extracts data from InputStream

There is no particular significance to the graphical symbols shown here. Students diagram just needs to clearly distinguish the different elements.

A Name Server

(See Simple Client/Server code)

- This is a common device. Let's examine a system to associate the names of an employee with their email address.
- This is also a good opportunity to demonstrate the use of a Hashtable.

Problems with this initial Server

- The server will only handle one client connection at a time.
- If there is more than one, they are queued and the next is processed once the first has closed.
- To overcome this we need to make use of threads.
- Each process can then be handled by a separate thread so that we can seemingly handle multiple clients simultaneously.