

Module 15

Networking

Objectives

- Develop code to set up the network connection
- Understand the TCP/IP Protocol
- Use `ServerSocket` and `Socket` classes for implementation of TCP/IP clients and servers

Relevance

How can a communication link between a client machine and a server on the network be established?

Networking

This section describes networking concepts.

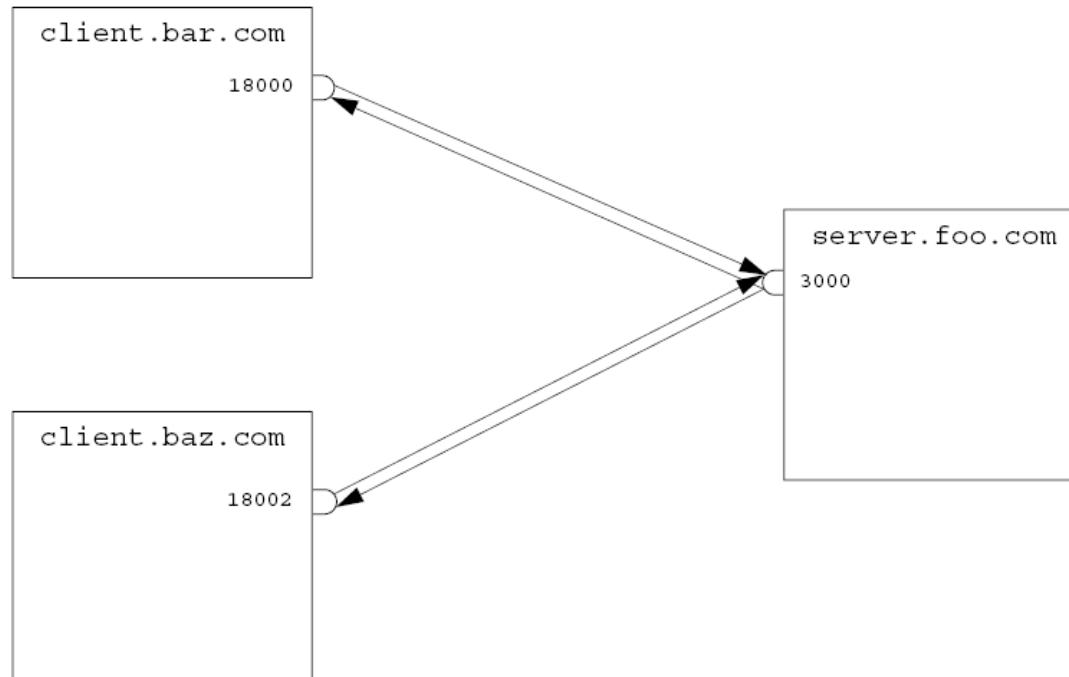
Sockets

- Sockets hold two streams: an input stream and an output stream.
- Each end of the socket has a pair of streams.

Setting Up the Connection

Set up of a network connection is similar to a telephone system: One end must *dial* the other end, which must be *listening*.

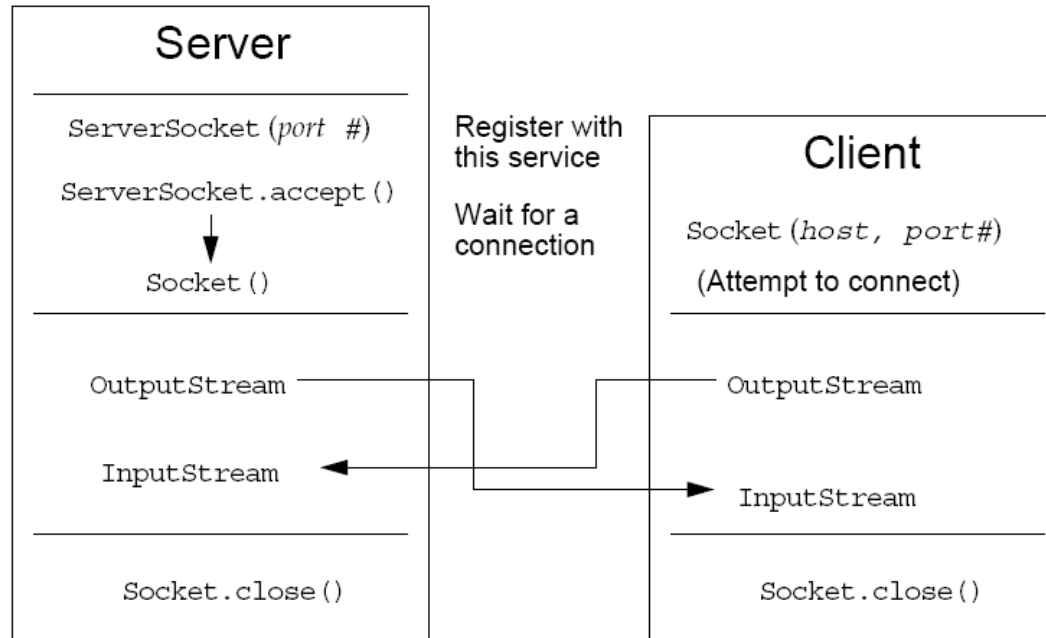
Networking



Networking With Java Technology

- To address the connection, include the following:
 - The address or name of remote machine
 - A port number to identify the purpose at the server
- Port numbers range from 0–65535.

Java Networking Model



Minimal TCP/IP Server

```
1  import java.net.*;
2  import java.io.*;
3
4  public class SimpleServer {
5      public static void main(String args[]) {
6          ServerSocket s = null;
7
8          // Register your service on port 5432
9          try {
10             s = new ServerSocket(5432);
11         } catch (IOException e) {
12             e.printStackTrace();
13         }
```


Minimal TCP/IP Server

```
14
15 // Run the listen/accept loop forever
16 while (true) {
17     try {
18         // Wait here and listen for a connection
19         Socket s1 = s.accept();
20
21         // Get output stream associated with the socket
22         OutputStream slout = s1.getOutputStream();
23         BufferedWriter bw = new BufferedWriter(
24             new OutputStreamWriter(slout));
25
26         // Send your string!
27         bw.write("Hello Net World!\n");
```

Minimal TCP/IP Server

```
28
29     // Close the connection, but not the server socket
30     bw.close();
31     s1.close();
32
33     } catch (IOException e) {
34         e.printStackTrace();
35     } // END of try-catch
36
37     } // END of while(true)
38
39     } // END of main method
40
41     } // END of SimpleServer program
```

Minimal TCP/IP Client

```
1  import java.net.*;
2  import java.io.*;
3
4  public class SimpleClient {
5
6      public static void main(String args[]) {
7
8          try {
9              // Open your connection to a server, at port 5432
10             // localhost used here
11             Socket s1 = new Socket("127.0.0.1", 5432);
12
13             // Get an input stream from the socket
14             InputStream is = s1.getInputStream();
15             // Decorate it with a "data" input stream
16             DataInputStream dis = new DataInputStream(is);
```

Minimal TCP/IP Client

```
17
18     // Read the input and print it to the screen
19     System.out.println(dis.readUTF());
20
21     // When done, just close the stream and connection
22     dis.close();
23     s1.close();
24
25     } catch (ConnectException connExc) {
26         System.err.println("Could not connect.");
27
28     } catch (IOException e) {
29         // ignore
30     } // END of try-catch
31
32     } // END of main method
33
34     } // END of SimpleClient program
```