Networking in Java

Michael Li
http://www.cs.nott.ac.uk/~jwl/G52APR

Outlines

• Networking basics
  – Network architecture
  – IP address and port
  – Server-client model
  – TCP and UDP protocol
• Java Socket
• A simple client program to send email

Internet hardware structure

• Network cards
• Hubs
• Routers
• Gateways

Networking Diagram

Networking in Java

• Java provides lots of networking support
• High-level interfaces
  – Fetch a file via HTTP
  – Execute a method on a remote object
• Low-level interfaces
• We'll start from the bottom and move up

IP Address

• In Internet Protocol (IP), each machine is given an address
• In IPv4, each address is made up of 4 bytes, e.g. 128.243.80.167
• Certain numbers have special values
• IPv6 is coming online – which expands the address range
DNS

- Domain Name Service (DNS): associating a textual name with an IP address
- www.google.com \(\leftrightarrow\) 209.85.169.104
- www.cs.nott.ac.uk \(\leftrightarrow\) 128.243.80.167
- A DNS server converts a DNS name to an IP address.

Command Prompt

- Two very useful commands:
  - `ipconfig/all`
  - `ping www.google.com`
- They are MS-DOS commands, can be executed in command prompt.

ipconfig command

ping command

Ports

- A port is a 16-bit number, 0-65535
- Some protocols work on specific ports
- Can’t create a server on a port below 1024
- Unless you are root (or equivalent)
- Usually the system assign the client’s port

IP/Port examples

- HTTP servers listen on port 80
- SMTP is port 25, POP3 is 110 and IMAP4 143
- www.bbc.co.uk has address 212.58.244.66
- So to connect to the BBC website you need to access port 80 on 212.58.244.66
InetAddress class
- In java.net package
- Can handle both IPv4 and IPv6 addresses
- No constructor, uses static methods to make objects
- Either from the numeric value or via DNS

InetAddress class
- InetAddress.getByAddress(byte[] addr)
  //Create an InetAddress representing the address in addr
- InetAddress.getByName(String host)
  //Determines the IP address of a host, given the host’s name
- InetAddress.getAllByName(String host)
  //Return an array of InetAddresses valid for //host

InetAddress class
- byte[] getAddress()
  //returns the raw IP address
- String getHostName()
  //returns the host name for this IP address
- String getCanonicalHostName()
  //returns the FQDN for this IP address
- String getHostAddress()
  //returns a textual presentation of the IP

Internet Protocol
- Internet is built on top of IP – the Internet protocol
- Designed to allow internetworking between different network types
- Mapped on top of other network technologies, e.g. Ethernet, ATM etc.

IP Networking
- Communication is generally point to point
- One machine (the client) connects to another machine (the server)
- Data is broken up into packets and sent between the two machines
- Packets can get lost, duplicated or delivered out of order

Server-Client model
A Server responses requests from the client(s).
TCP vs UDP

- **TCP**
  - Transmission Control Protocol
  - Stateful
  - Accurate delivery
  - But can incur delays
  - Effectively becomes stream based

- **UDP**
  - User datagram protocol
  - Stateless
  - Packet (or datagram) based
  - Not guarantee on delivery
  - Good for real-time applications (e.g. streaming media)

**Socket**

- A socket is one end-point of a two-way communication link between two programs running on the network.
- Socket classes are used to represent the connection between a client program and a server program.
- The java.net package provides two classes-- **Socket** and **ServerSocket** --that implement the client side of the connection and the server side of the connection, respectively.

**TCP Server-Client model**

- One machine (the server) sets up a socket to listen for incoming connections
- Another machine (the client) attempts to connect one of its sockets to the server's socket
- The server accepts this connection
- And data can be sent and received

**Socket connections**

- All sockets have an address and a port number
- To connect to a socket you need to know its address and port
- Data transfer is duplex (both machines can send data at the same time)
Sockets in Java

• IP connection made using sockets
• Java encapsulates these in objects
• Create a Socket object
• connect() to a specific IP and port
• Transfer data
• Then close() when finished

Socket methods

• Socket()
  //New socket
• Socket(InetAddress addr, int port) throws IOException
  //Create a new socket and connect to specified address and port
• void close() throws IOException
  //Closes this socket
• void connect(SocketAddress endpoint) throws IOException
  //Connect socket to endpoint, SocketAddress is a convenience class
• void close() throws IOException
  //Closes this connection

Also convenience methods that allow you create directly from a String
Go check the Java document for detail.

Socket methods example

```
Socket clientSocket = new Socket();
clientSocket.connect("192.168.0.101",1344);
```

```
Socket clientSocket = new Socket("192.168.0.101",1344);
```

```
Socket clientSocket = new Socket("localhost",1566);
```

Transferring Data

• You already know how to do this …
• Uses the standard Java I/O classes, InputStream and OutputStream
• Methods provided in Socket object to access the I/O objects
• Can then wrap in BufferedReader and PrintWriters etc.

Socket methods

• InputStream getInputStream() throws IOException
  //Gets an InputStream for this Socket
• OutputStream getOutputStream() throws IOException
  //Gets an OutputStream for this Socket
• Closing the streams will close the socket

A TCP client example

• As a simple example of socket programming we can implement a program that sends email to a remote site
  – This is taken from Core Java, vol 2, chapter 3
• Email servers use port number 25 which is the SMTP port
  – Simple mail transport protocol
• A client socket to the mail server can be opened on port 25
  Socket s=new Socket("engmail.bham.ac.uk",25);

• SMTP uses the following protocol
  HELO sending host
  MAIL FROM: sender email address
  RCPT TO: recipient email address
  DATA
  mail message
  ...
  QUIT

import java.io.*;
import java.net.*;
public class EmailSender{
  public static void main() throws Exception{
    Socket s=new Socket("128.243.15.141",25);
    PrintWriter out=new PrintWriter(s.getOutputStream());
    String hostname= InetAddress.getLocalHost().getHostName();
    out.println("HELO " + hostname);
    out.println("MAIL FROM: " + "jiawei.li@cs.nott.ac.uk");
    out.println("RCPT TO: " + "jiawei.li@cs.nott.ac.uk");
    out.println("DATA");
    out.println("This is a test.");
    out.println("QUIT");
  }
}

Summary

• Networking basics
  – Network architecture
  – IP address and DNS
  – port
  – InetAddress class
  – Server-client model
• Socket
• A simple client program to send email