Class 23:
Network Programming
(just enough to make you dangerous)

Fall 2010
UVa
David Evans

Bill Cheswick’s map of the Internet (1999)

Plan for Today
- PS5 (finally!)
- Networking

Philosopher Deadlock

public class Philosopher {
    private Philosopher colleague;
    ...
    public synchronized void setColleague(Philosopher p) {
        colleague = p;
    }
    ...
    public synchronized void philosophize() {
        System.err.println(name + " says " + quote);
        if (colleague != null) { // Need a colleague to start an argument.
            colleague.argue();
        }
    }
    ...
    public synchronized void argue() // REQUIRES: this.colleague != null
    {
        System.err.println(name + " argues: No! " + quote);
    }
}

Attempted Fix: Locking Discipline

public void philosophize() {
    Object lock1, lock2;
    if (colleague != null) {
        if (name.compareTo(colleague.name) < 0) {
            lock1 = this;
            lock2 = colleague;
        } else {
            lock1 = colleague;
            lock2 = this;
        }
    } else {
        lock1 = lock2 = null;
    }
    synchronized (lock1) {
        synchronized (lock2) {
            colleague.argue();
        }
    }
}

5. (Tricky, extra credit if you can answer this) Our new Philosopher class now has a race condition that could lead to a deadlock or run-time exception (but it would never be apparent from our current main class). Explain what the race condition is, and construct code that reveals it. Feel free to insert sleep pauses as necessary to make it easier to reveal.
Networking

Internet 1969

Client-Server Networking

Same host can be both a client and a server (peer-to-peer networking)

Network Topologies

“Star”

“Mesh”

Skype

SuperNodes: form a mesh overlay network
Client/SuperNode: star network
Client-Login Server: client-server

Skype Login Server

Facebook

API calls, FBML

Application Server

How can you run skype, browsers, YouTube, etc. all on one host?

Adrienne Felt’s paper:
Privacy Protection for Social Networking Platforms
**Ports**

One host can be involved in several simultaneous network connections: use ports to direct traffic to the right process.

**Port Numbers**

- **Ports 0-1023**: assigned by Internet Assigned Numbers Authority
  - Privileged programs (administrator/root)
  - 25: smtp, 80: http, 110: pop3, 205: appletalk
  - [http://www.iana.org/assignments/port-numbers](http://www.iana.org/assignments/port-numbers)
- **Ports 1024-49151**: registered
  - Any application can use these
- **Ports 49152-65535**: dynamic/private

**GUIs and Networks**

**GUIs**: great problem for subtyping and inheritance
- OOP was invented to program GUIs (and build simulations)

**Network programming**: great problem for data abstraction

Why are GUIs great for OOP and networks great for data abstraction?

**OSI Abstraction Layers**

<table>
<thead>
<tr>
<th>Layer</th>
<th>Abstraction Level</th>
<th>Example Protocols</th>
</tr>
</thead>
<tbody>
<tr>
<td>Application</td>
<td>semantic objects</td>
<td>HTTP, SMTP, Skype, ...</td>
</tr>
<tr>
<td>Presentation</td>
<td>data representation, encryption, machine-independent data</td>
<td>SSL, TLS, MIME</td>
</tr>
<tr>
<td>Session</td>
<td>host-host communication</td>
<td>sockets</td>
</tr>
<tr>
<td>Transport</td>
<td>end-to-end connections, flow control</td>
<td>TCP, UDP</td>
</tr>
<tr>
<td>Network</td>
<td>routing, logical addressing</td>
<td>IP</td>
</tr>
<tr>
<td>Data Link</td>
<td>physical addressing</td>
<td>Ethernet, 802.11</td>
</tr>
<tr>
<td>Physical</td>
<td>binary transmission</td>
<td>X25, RS-232, POTS</td>
</tr>
</tbody>
</table>

What do we gain/lose by viewing network in these abstraction layers?

**Java Sockets**

```
java.net.Socket
java.net.ServerSocket
```
java.net.ServerSocket

```java
import java.net.ServerSocket;
import java.io.IOException;

public class Network {
    static public void main(String args[]) {
        ServerSocket ss1, ss2;
        try {
            ss1 = new ServerSocket(8000);
        } catch (IOException ioe) {
            System.err.println("Exception 1: " + ioe);
        }
        try {
            ss2 = new ServerSocket(8000);
        } catch (IOException ioe) {
            System.err.println("Exception 2: " + ioe);
        }
    }
}
```

**Exception 2: java.net.SocketException:**
Unrecognized Windows Sockets error: 0: JVM_Bind

Accepting Connections

```java
public Socket accept() throws IOException
    Listens for a connection to be made to this socket
    and accepts it. The method blocks until a connection
    is made. A new Socket s is created and, if there
    is a security manager, the security manager's
    checkAccept method is called with
    s.getInetAddress().getHostAddress() and s.getPort()
    as its arguments to ensure the operation is allowed.
    This could result in a SecurityException.
```

Server

```java
import java.net.ServerSocket;
import java.io.IOException;

public class Server {
    static public void main(String args[]) {
        ServerSocket listener;
        try {
            listener = new ServerSocket(8000);
        } catch (IOException ioe) {
            System.err.println("Cannot open server socket: " + ioe);
            return;
        }
        System.out.println("Server: waiting for connection...");
        try {
            Socket sock = listener.accept();
            ... etc ...
        } catch (IOException ioe) {
            System.err.println("Cannot accept: " + ioe);
        }
    }
}
```
Sockets

**Socket**(String host, int port) throws IOException, UnknownHostException

**EFFECTS:** Creates a stream socket and connects it to the specified port number on the named host. If the specified host is null it is the loopback address.

```java
public void close() throws IOException
EFFECTS: Closes this socket.
```

Creating a Socket

```java
import java.net.Socket;
import java.io.*;
import java.net.UnknownHostException;

public class Client {
    public static void main(String[] args) {
        Socket connect;
        try {
            connect = new Socket("www.microsoft.com", 80);
            System.out.println("Connected: " + connect);
        }
        catch (UnknownHostException uhe) {
            System.err.println("Cannot find host: " + uhe);
        }
        catch (IOException ioe) {
            System.err.println("Cannot open connection: " + ioe);
        }
    }
}
```

Port 80 = http (web server)

Connected to Microsoft

```java
import java.net.Socket;
import java.io.*;
import java.net.UnknownHostException;

public class Client {
    public static void main(String[] args) {
        Socket connect;
        try {
            connect = new Socket("www.microsoft.com", 80);
            System.out.println("Connected: " + connect);
        }
        catch (UnknownHostException uhe) {
            System.err.println("Cannot find host: " + uhe);
        }
        catch (IOException ioe) {
            System.err.println("Cannot open connection: " + ioe);
        }
    }
}
```

Connected: Socket[addr=www.microsoft.com/207.46.19.30, port=80, localport=1359]

Connected: Socket[addr=www.microsoft.com/207.46.225.60, port=80, localport=1371]

Communicating

```java
Socket methods:

public OutputStream getOutputStream() throws IOException
EFFECTS: Returns an output stream for this socket.

public InputStream getInputStream() throws IOException
EFFECTS: Returns an input stream for this socket.
```

Input Streams

```java
public abstract class InputStream
public abstract int read() throws IOException
EFFECTS: Returns the next byte in the input stream and advances the stream.
other methods: close, available, skip, etc.
```

Lots of InputStreams

```java
java.io.InputStream
java.io.FilterInputStream
java.io.BufferedInputStream
java.io.FileInputStream
```

Reads the next byte of data from the input stream. The value byte is returned as an int in the range 0 to 255. If no byte is available because the end of the stream has been reached, the value -1 is returned. This method blocks until input data is available, the end of the stream is detected, or an exception is thrown. A subclass must provide an implementation of this method.

Returns: the next byte of data, or -1 if the end of the stream is reached.
Throws: IOException - if an I/O error occurs.
Readers

java.io.InputStreamReader extends java.io.Reader
Higher level abstraction on an InputStream
InputStreamReader(InputStream in)
EFFECTS: Initializes this to an InputStreamReader on stream in.
int read()
EFFECTS: Reads a single character.

Buffering

BufferedReader extends Reader {
BufferedReader(Reader r)
int read ()
String readLine ()
EFFECTS: Returns the next line and advances the reader.

Cookbook...

Socket connect;
...
BufferedReader in =
new BufferedReader (new InputStreamReader (connect.getInputStream()));

A Simple Web Browser

import java.net.Socket;
import java.io.*;
import java.net.UnknownHostException;
public class Client {
public static void main(String[] args) {
Socket connect;
try {
connect = new Socket("www.virginia.edu", 80);
System.out.println("Connected");
} catch (UnknownHostException uhe) {
...
return;
} catch (IOException ioe) {
System.err.println("Cannot open ...");
return;
}
try {
PrintWriter out = new PrintWriter (connect.getOutputStream(), true);
BufferedReader in = new BufferedReader (new InputStreamReader (connect.getInputStream()));
out.println("GET / HTTP/1.0\r\n");
String inString;
while ((inString= in.readLine()) != null) {
System.out.println(inString);
}
connect.close();
} catch (IOException ioe) {
System.err.println("IO Exception: " + ioe);
}
System.out.println("Closing connection...");
try {
} catch (IOException ioe) {
System.err.println("Error closing: " + ioe);
}

Higher Abstraction Levels

java.net.HttpURLConnection
URLConnection (URL url)
Constructs a URL connection to the specified URL.
void connect()
Object getContent(Class [] classes)

http://java.sun.com/docs/books/tutorial/networking/urls/

What would need to be different in a real web server?
Building Web Applications

- Don’t hand code a server like this!
- Lots of powerful frameworks available

- Java
- Other Languages: JSP (JavaServer Pages), Python, Ruby, PHP, Django, Symfony, RAILS

Charge

- Exam 2: out Thursday, due Tuesday, Nov 23

If you have topics you want me to review before the exam, send them by Monday afternoon.