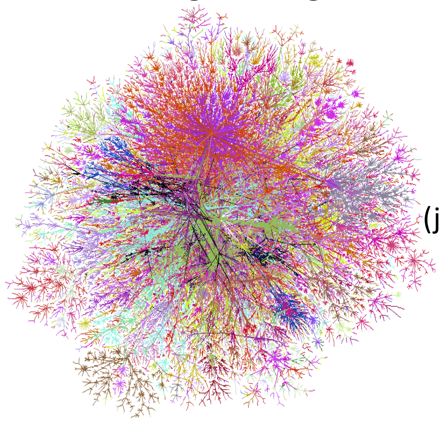
cs2220: Engineering Software



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

Excuse

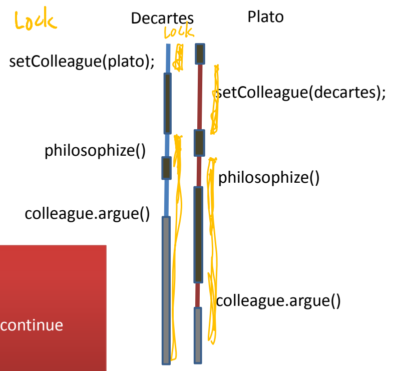
Excuse

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);
  }
}
```

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) {
      colleague.argue();
    }
  }

  public synchronized void argue()
  {
    System.err.println(name + " argues: No! " +
quote);
  }
}
```



Deadlock
Decartes thread:
holds Decartes lock
needs Plato lock to continue
Plato thread:
holds Plato lock
needs Decartes lock to continue

Attempted Fix: Locking Discipline

sync . .

synchronize

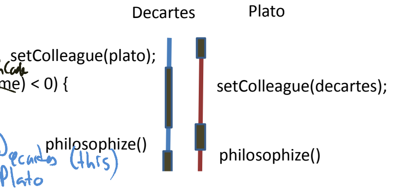
```
public void philosophize () {
  Object lock1, lock2;
  if (colleague != null) {
    if (name.compareTo (colleague.name) < 0) {
      lock1 = this;
      lock2 = colleague;
    } else {
      lock1 = colleague;
      lock2 = this;
    }
    synchronized (lock1) {
      synchronized (lock2) {
    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

~~colleague.name~~ hashCode

lock1 = Decartes (this)
lock2 = Plato

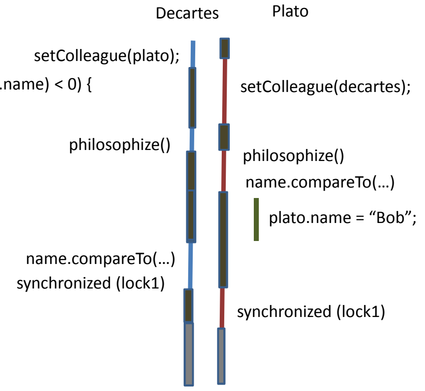
lock1 = Decarts
lock2 = Plato (this)

D.setColleague(Aristotle);

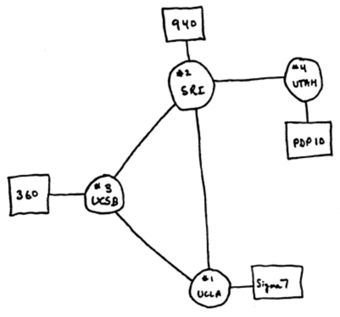


5. (Tricky, extra credit if you can answer this) Our new Philosopher class now has a race condition that could lead to a deadlock or run-time exception (but it would never be apparent from our current main class). Explain what the race condition is, and construct code that reveals it. Feel free to insert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

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

  synchronized (lock1) {
    synchronized (lock2) {
      colleague.argue ();
    }
  }
}
```

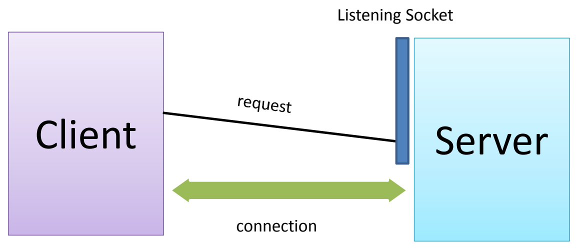


Networking



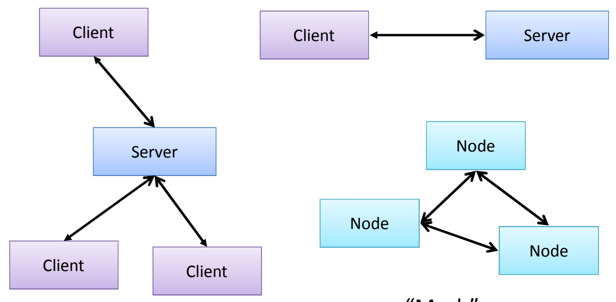
Internet 1969

Client-Server Networking



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

Network Topologies

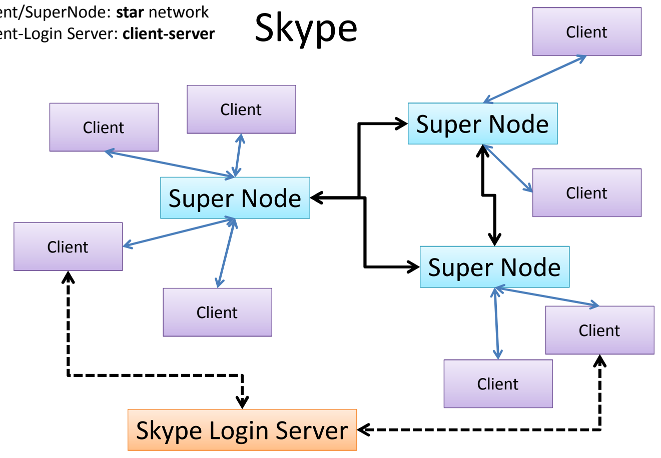


"Star"

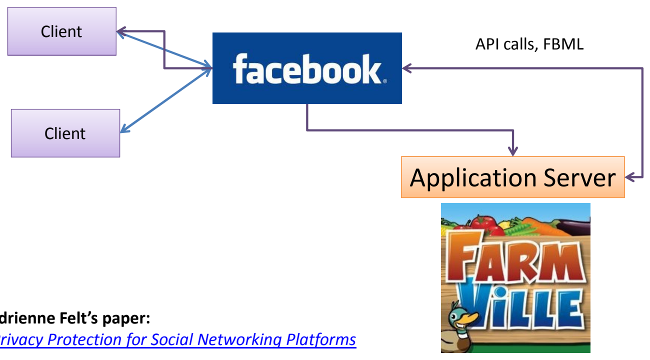
"Mesh"

Skype

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



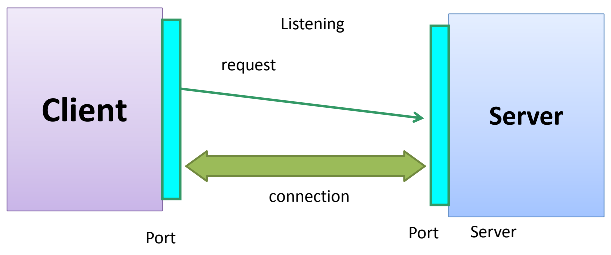
Facebook



Adrienne Felt's paper:
Privacy Protection for Social Networking Platforms

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

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

Ports 1024-49151: registered

Any application can use these

Ports 49152-65535: dynamic/private

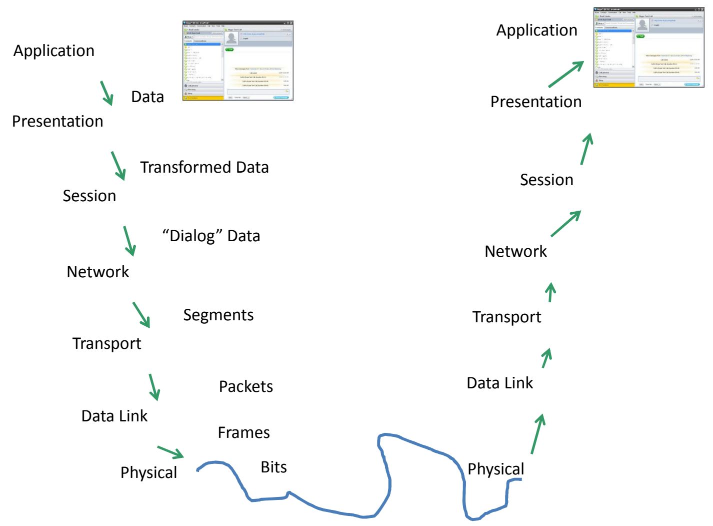
GUIs and Networks

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?

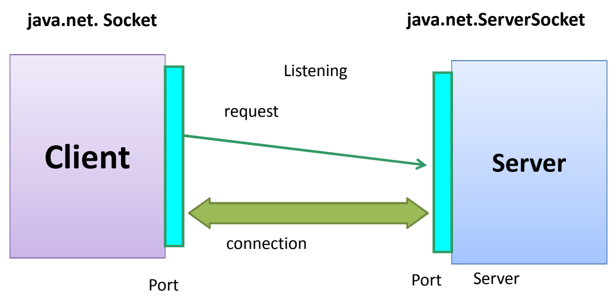


OSI Abstraction Layers

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

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

Java Sockets

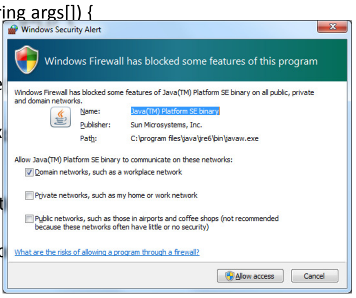


java.net.ServerSocket

public ServerSocket(int port) throws IOException
 EFFECTS: Initializes this to a new server socket on port. If the socket cannot be created, throws IOException.

```
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);
   }
  }
}
```

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

public class Network {
  static public void main(String args[]) {
    ServerSocket ss1, ss2;
    try {
      ss1 = new ServerSocke
    } catch (IOException ioe)
      System.err.println ("Ex
    }
    try {
     ss2 = new ServerSocket
    } catch (IOException ioe)
     System.err.println ("Exc
   }
  }
}
```


Windows Security Alert

Windows Firewall has blocked some features of this program

Windows Firewall has blocked some features of Java(TM) Platform SE binary on all public, private and domain networks.

Name: Java(TM) Platform SE binary
Publisher: Sun Microsystems, Inc.
Path: C:\program files\java\jre6\bin\javaw.exe

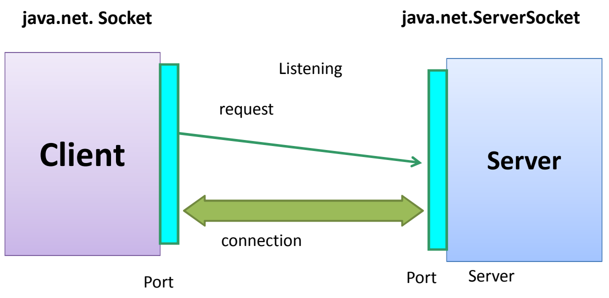
Allow Java(TM) Platform SE binary to communicate on these networks:

☑ Domain networks, such as a workplace network
☐ Private networks, such as my home or work network
☐ Public networks, such as those in airports and coffee shops (not recommended because these networks often have little or no security)

What are the risks of allowing a program through a firewall?

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

Java Sockets



Accepting Connections

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

```
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();
      ...
    } 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.

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

Creating a Socket

```
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

```
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);
    } catc
      Sys
    }
  }
}
```

```
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

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.

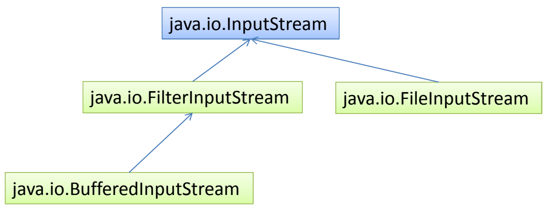
Input Streams

public abstract class InputStream
public abstract int **read**() throws IOException

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.

other methods: close, available, skip, etc.

Lots of InputStreams



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()));

PrintWriter out =
  new PrintWriter
    (connect.getOutputStream(), true);
```

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);
   }
} catch (IOException ioe) {
   System.err.println("IO Exception: " + ioe);
}
System.out.println ("Closing connection...");
try {
  connect.close();
} 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/

```
public class WebServer {
  static public void main(String args[]) {
   ServerSocket listener;
   try {
     listener = new ServerSocket(80);
   } catch (IOException ioe) {
     System.err.println("Cannot open server socket: " + ioe);  return;
   }
  while (true) {
    try {
      System.out.println("Server: waiting for connection...");
      Socket connect = listener.accept();   create new thread
      PrintWriter out = new PrintWriter(connect.getOutputStream(), true);
      BufferedReader in = new BufferedReader(new InputStreamReader(connect.getInputStream()));
      String inString;
      while ((inString = in.readLine()) != null) {
         // HTTP request: GET <file> <protocol>
         ... // Process request by printing to out
      }
      connect.close();
    } catch (IOException ioe) {
      // log error
    }
...
}
```

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

Apache Tomcat

JSP (JavaServer Pages)

Other Languages

django

Python

symfony

PHP

RAILS

Ruby

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.