Networking

9-26-2011

Opening Discussion

- What did we talk about last class?
- IcP solutions

Networking

- These days, a computer loses a lot of its value if it isn't networked.
- We need to learn how to allow our programs to talk to other computers.
- This can happen in a lot of different ways from just reading information off the machine to having a “dialog” to exchange information.
- Most things we want are in the java.net package.

Sockets

- Computers communicate over sockets. They come in two main flavors.
  - TCP – This is the default. Does handshaking to determine if messages get through. Reliable, but slower.
  - UDP – Throw packets out and hope the other side gets them. Fast, but code has to deal with possible dropped packets.
- One machine acts as a server and waits on a port. Other machines, clients, can connect to that port.

Sockets and Streams

- Sockets in Java communicate through streams. So any code you wrote for file streams can be converted to networking with little to no effort.
- Let's write a simple telnet based chat room first.
- After that we can add either chat or sending drawings to our main program.

Remote Method Invocation (RMI)

- Standard socketing approach gets challenging when there are a lot of different method types. Java has RMI to help deal with this.
- Steps in RMI (for Scala)
  - Make a completely abstract trait that implements java.rmi.Remote with the methods you want to call remotely. @throws(classOf[RemoteException])
  - Implement in class that extends java.rmi.server.UnicastRemoteObject.
  - Naming.bind/rebind and Naming.lookup
  - rmiregistry

Minute Essay

- What questions do you have about networking?