Announcements

- Project proposals due Friday
- Signup sheet for planning meetings
- Interested in TAing next semester?
Today’s Plan

SERVERS

- Web Standards and Protocols
  - HTTP
  - HTML

- Server Programming with Squint
  - TCP Ports
A BRIEF EXAMPLE

<html>
<head>
    <TITLE>Tom Murtagh's Home</TITLE>
</head>

<body>
    <H1>Tom Murtagh</H1>
    <P>OK!!! I've given in and made a home page. Well, it's a start...</P>
</body>

</html>
HTML Document Structure

<HTML>

<HEAD>

<TITLE> something short but sweet </TITLE>

</HEAD>

<BODY>

something not so short but sweet

</BODY>

</HTML>
HTML HEAD TAGS

<TITLE> My Wonderful Webpage </TITLE>
HTML BODY TAGS

- `<P>`
- `<BR>`
- `<Hn>` ...
- `</Hn>`
- `<B>` ...
- `<B>`
- `<CENTER>` ...
- `</CENTER>`
ATTRIBUTES

- `<p align=center>`  
- `<font color=red> ...
  </font>`
LINKS

< A HREF="http://www.cs.williams.edu"> ... </a>
ACCESSING OTHER FILES

<IMG SRC="someGreatPicture.gif">
Clients vs. Servers

Role of clients - retrieve data from servers
1. Contact server (... = new NetConnection(“www.cs”, 80); )
2. Send requests ( toServer.out.println(“GET ...”); )
3. Retrieve responses ( response = toServer.in.nextLine(); )
4. Disconnect (toServer.close();)

Role of servers - “serve” data to clients
1. Patiently wait for client connections ( ??? )
2. Accept valid connection ( ??? )
3. Receive requests ( request = fromClient.in.nextLine(); )
4. Send data to clients ( fromClient.out.println( response )
5. Close client connection ( fromClient.close() );
Ports

- Servers “listen” for client connection requests on specific ports
  - Recall that HTTP uses port 80
  - Ports are like phone number extensions
- In Squint, we indicate that a server wants to activate a port by saying:
  
  ```java
  TCPPort connectPort = new TCPPort( 80 );
  ```
Answering a Call

A server can

- wait for a client to create a NetConnection, and
- get access to the NetConnection

by saying:

```java
NetConnection fromClient =
    connectPort.acceptNetConnection();
```

assuming:

```java
TCPPort connectPort = new TCPPort(110);
```
Network Client Events

In our client code, we added a MessageListener to be notified when the server sent us a new message:

```java
toServer.addMessageListener(this);
...

public void dataAvailable() {
}
```
Network Server Events

We can ask Java to execute a method when our server receives a connection request:

```java
connectPort.addConnectionListener( this );

public void connectionEstablished( TCPPort p ) {
    NetConnection fromClient =
        connectionPort.acceptNetConnection();
    ...
}
```
Connecting to Files

Scanner txtfile = new Scanner( new File( "name" ) );

String line = txtfile.nextLine();

while ( txtfile.hasNextLine() ) { ...
try {
    statement that might not work
} catch ( Exception error ) {
    statements to make things better
}