TCP Servers

ComS 587X
Fall, 2002

Servers

- Server provides services, typically at a predefined port
- Server typically runs for a long time (i.e., a daemon)
  - Constantly accepting new connections and processing client requests
  - Usually started by the operating system at boot time

ServerSocket

- java.net.ServerSocket creates Sockets out of incoming connections
  - Bind to specific (usually well-known) port
  - May accept more than one connection
- Constructors include:
  - ServerSocket(int port)
    - Listen queue set to 50
  - ServerSocket(int port, int listenQueueSize)
    - Explicitly set listen queue size
  - ServerSocket(int port, int listenQueueSize, InetAddress addr)
    - Explicitly bind a local address

ServerSocket Methods

- Socket accept()
  - Waits for connection, then returns Socket
- void close()
- InetAddress getInetAddress()
- int getLocalPort()
- int getSoTimeout()
- void implSocket(Socket socket)
- static void setSocketFactory(SocketImplFactory factory)
- void setSoTimeout(int duration)

Accepting Client Connections

- After construction, a serverSocket is ready to accept connections from clients

```
try {
  String s;
  ServerSocket serverSock = new ServerSocket(8080);
  Socket newSock = serverSock.accept();
  PrintStream pstream = new PrintStream(
   newSock.getOutputStream() );
  pstream.println("Hello, Client!");
  newSock.close();
  serverSock.close();
} catch (Exception e) {
  System.err.println("Error: " + e);
}
```

Daytime Server

```
public class Day timeServer {
 public static final int SERVICE_PORT = 13;
 public static void main(String args[]) {
  try {
   ServerSocket server = new ServerSocket(SERVICE_PORT);
   System.out.println("Daytime service started");
   for ( ; ; ) {
    Socket nextClient = server.accept();
    System.out.println("Received request from" +
     nextClient.getlnetAddress()+ ":" + nextClient.getport());
    PrintStream pout = new PrintStream(nextClient.getOutputStream());
    pout.print( new java.util.Date() );
    pout.flush();
    pout.close();
    nextClient.close();
   }
  }
 catch (BindException be) {
  System.err.println("Could not bind " + SERVICE_PORT + ": " + be);
 } catch (IOException ioe) { System.err.println("IO error: " + ioe);}
}
```

Socket Exceptions

- SocketException
  - Generic socket error
- BindException
  - Could not bind to the given local port
    - Port already in use
    - User not allowed access to the port
- ConnectException
  - Could not connect to the remote port
    - Remote port is not accepting connections
    - Remote port's listen queue is full

Socket Exceptions (2)

- NoRouteToHostException
  - Could not reach the remote host
    - Link or router failure
    - No such subnet at the remote network
    - Firewall blocks connections
- InterrupedIOException
  - Read operation is interrupted by a timeout

Summary

- TCP ServerSockets
- Simple server
- Daytime server
- Socket Exceptions