Networking Code

CSCI 201L

Jeffrey Miller, Ph.D.
jeffrey.miller@usc.edu

HTTP://WWW-SCF.USC.EDU/~CSCI201
Outline

- Server Networking
- Client Networking
- Program
Server Software

- A server application is only able to serve requests that are addressed to the computer on which the server application is running
  - A server program cannot listen to a port on another physical server
- So, the only data the server application needs is the port on which to listen
  - The ServerSocket constructor only takes a port as a parameter
- Multiple networked applications can be running on the same computer as long as they are all using different ports
Multi-Threading with Networking

- Multi-threading is usually necessary with networking since there are two things that often are done at the same time:
  - The ability to send data
  - The ability to receive data
- If sending and receiving are not performed in series, multi-threading will be needed:
  - Some applications may only need one program to send data then wait for a response – that would not require multi-threading
Server Networking Example (no multi-threading)

```java
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.ServerSocket;
import java.net.Socket;

public class NetworkingServer {

    public NetworkingServer() {
        try {
            System.out.println("Starting Server");
            ServerSocket ss = new ServerSocket(6789);
            Socket s = ss.accept();
            BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
            PrintWriter pw = new PrintWriter(s.getOutputStream());

            String line = br.readLine();
            System.out.println("Line Received: " + line);
            String str = "CSCI 201";
            System.out.println("Sending: " + str);
            pw.println(str);
            pw.flush();
            pw.close();
            br.close();
            s.close();
            ss.close();
        } catch (IOException ioe) {
            System.out.println("IOE: " + ioe.getMessage());
        }
    }

    public static void main(String [] args) {
        new NetworkingServer();
    }
}
```
Flushed

- Operating systems try to optimize networking similar to how they optimize file I/O
- A buffer exists where data is written into before sending along the socket
- The buffer will not be transmitted over the network until it fills up *unless* we explicitly **flush** the data
- Never forget to flush!
Outline

- Server Networking
- Client Networking
- Program
Client Software

- A client application is able to connect to any server application to which it has access, whether running on the same computer as the client or a different computer.
- The client application needs both the IP address of the server and the port on which the server application with which it wants to communicate is listening.
  - Remember that multiple server applications can be running on the same computer as long as they are listening on different ports.
- A **Socket** is the combination of the IP address and port number needed by the client.
import java.io.BufferedReader;
import java.io.IOException;
import java.io.InputStreamReader;
import java.io.PrintWriter;
import java.net.Socket;

public class NetworkingClient {

    public NetworkingClient() {
        try {
            System.out.println("Starting Client");
            Socket s = new Socket("localhost", 6789);
            BufferedReader br = new BufferedReader(new InputStreamReader(s.getInputStream()));
            PrintWriter pw = new PrintWriter(s.getOutputStream());

            String str = "Line being sent";
            System.out.println("Sending: " + str);
            pw.println(str);
            pw.flush();
            String line = br.readLine();
            System.out.println("Line Received: " + line);
            pw.close();
            br.close();
            s.close();
        } catch (IOException ioe) {
            System.out.println("IOE: " + ioe.getMessage());
        }
    }

    public static void main(String [] args) {
        new NetworkingClient();
    }
}
Outline

▪ Server Networking
▪ Client Networking
▪ Program
Program

- Write a multi-threaded chat program that allows multiple clients to communicate with each other in an asynchronous manner. The clients should communicate with a server program.

C:>java ChatClient localhost 6789
hello
Them: how are you?
fine, and you?
Them: good, thanks

C:>java ChatClient localhost 6789
Them: hello
how are you?
Them: fine, and you?
good, thanks