Version 1.3

DS420 -- Elliott
Sample code for InetServer and InetClient

These running programs are provided as a guide. The .class files from these programs are also available in one of the DS 420 directories, so that you can use, e.g., the running client to test your own server, or the running server to test your client.

I recommend that you first make sure that Java is running on your machine by loading, and running, the InetAddresser program. Assuming that you are able to get java, TCP/IP, and your firewall sorted out, you should then load, and run, the existing, byte-compiled, InetServer and InetClient.

Then, if you are an advanced programmer you may simply write your own client and server code. Otherwise TYPE IN THE PROGRAMS and get them running. This is an excellent way to learn. If you are clever you can probably figure away to scan the text from the .pdf file, but this will not help you to learn the material, and I strongly do not recommend it.

Use your own variable names, and alter the program logic as you see fit.

Tip: Start small and get something running first, then gradually add functionality, and complexity. But, always keep your program running.

These programs are explicitly without many comments. This is intentional, because I want YOU to figure out exactly what is going on, and **write your own comments**, which is an important part of your grade.

Good luck. Have fun.

Standalone InetAddresser Progam:

```
/** file is: InetAddresser.java Version 1.3
```

```
    Elliott, after Hughes, Shoffner, Winslow

    This will not run unless TCP/IP is loaded on your machine.
----------------------------------------------------------------------
*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

public class InetAddresser {
  public static void main (String args[]) {
    printLocalAddress ();

    BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));

    try {
      String name;
      do {
        System.out.print("Enter a hostname or an IP address, (quit) to
end: ");
        System.out.flush ();
        name = in.readLine ();
      if (name.indexOf("quit") < 0)
        printRemoteAddress (name);
      } while (name.indexOf("quit") < 0);
      System.out.println ("exit");
    } catch (IOException x) {x.printStackTrace ();}
  }

  static void printLocalAddress () {
    try {
      System.out.print("Clark Elliott's INET addresser program\n");
      InetAddress me = InetAddress.getLocalHost ();
      System.out.println("My local name is:       " + me.getHostName
());
      System.out.println("My local IP address is: " +
toText(me.getAddress ()));
    } catch (UnknownHostException x) {
      System.out.println ("I appear to be unknown to myself.
Firewall?:");
      x.printStackTrace ();
    }
  }


  static void printRemoteAddress (String name) {
    try {
      System.out.println ("Looking up " + name + "...");
      InetAddress machine = InetAddress.getByName (name);
      System.out.println ("Host name : " + machine.getHostName ());
      System.out.println ("Host IP : " + toText (machine.getAddress
()));
    } catch (UnknownHostException ex) {
      System.out.println ("Failed to lookup " + name);
    }
  }
  static String toText (byte ip[]) {  /* Make portable for 128 bit
format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }
}
```

InetServer Progam:

```
/** File is: InetServer.java, Version 1.3

    A multithreaded server for InetClient
    Elliott, after Hughes, Shoffner, Winslow

    This will not run unless TCP/IP is loaded on your machine.

    Note that there is a minor design flaw, which is interesting from a
pedagocial standpoint. Uncomment the Thread.sleep line in the main
server
listening loop to have the flaw illustrated.
----------------------------------------------------------------------
*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

class Worker extends Thread {    // Class definition
  Socket sock;                   // Class member, socket, local to
Worker.
  Worker (Socket s) {sock = s;}  // Constructor, assign arg s to local
sock

  public void run(){
    // Get I/O streams from the socket:
    PrintStream out = null;
    BufferedReader in = null;
    try {
      out = new PrintStream(sock.getOutputStream());
      in = new BufferedReader
      (new InputStreamReader(sock.getInputStream()));
      // Note that it is remotely possible that this branch will not
execute:
      if (InetServer.controlSwitch != true){
      System.out.println
        ("Listener is now shutting down as per client request.");
      out.println("Server is now shutting down. Goodbye!");
      }
      else try {
      String name;
      name = in.readLine ();
      if (name.indexOf("shutdown") > -1){
        InetServer.controlSwitch = false;
        System.out.println("Worker has captured a shutdown request.");
        out.println("Shutdown request has been noted by worker.");
        out.println("Please send final shutdown request to listener.");
      }
      else{
        System.out.println("Looking up " + name);
        printRemoteAddress(name, out);
      }
      } catch (IOException x) {
      System.out.println("Server read error");
      x.printStackTrace ();
      }

      sock.close(); // close this connection, but not the server;
    } catch (IOException ioe) {System.out.println(ioe);}
  }

  static void printRemoteAddress (String name, PrintStream out) {
    try {
      out.println("Looking up " + name + "...");
      InetAddress machine = InetAddress.getByName (name);
      out.println("Host name : " + machine.getHostName ());
      out.println("Host IP : " + toText (machine.getAddress ()));
    } catch(UnknownHostException ex) {
      out.println ("Failed in atempt to look up " + name);
    }
  }

  static String toText (byte ip[]) {  /* Make portable for 128 bit
format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }
}

public class InetServer {

  public static boolean controlSwitch = true;

  public static void main(String a[]) throws IOException {
    int q_len = 6; /* Number of requests for OpSys to queue */
    int port = 1565;
    Socket sock;

    ServerSocket servsock = new ServerSocket(port, q_len);
    System.out.println
      ("Clark Elliott's Inet server starting up, listening at port
1565.\n");
    while (controlSwitch) {
      // wait for the next client connection:
      sock = servsock.accept();
      new Worker (sock).start(); // Uncomment to see shutdown bug:
      // try{Thread.sleep(10000);} catch(InterruptedException ex) {}
    }
  }
}
```

InetClient Progam:

```
/** file is: InetClient.java Version 1.3

    Elliott, after Hughes, Shoffner, Winslow
    Use with InetServer.java, the multithreaded listener.

    This will not run unless TCP/IP is loaded on your machine.
----------------------------------------------------------------------
*/

import java.io.*;  // Get the Input Output libraries
import java.net.*; // Get the Java networking libraries

public class InetClient{
  public static void main (String args[]) {
    String serverName;
    if (args.length < 1) serverName = "localhost";
    else serverName = args[0];

    System.out.println("Clark Elliott's Inet Client.\n");
    /* printLocalAddress (); */
    System.out.println("Using server: " + serverName + ", Port: 1565");
    BufferedReader in = new BufferedReader(new
InputStreamReader(System.in));
    try {
      String name;
      do {
      System.out.print
        ("Enter a hostname or an IP address, (quit) to end: ");
      System.out.flush ();
      name = in.readLine ();
      if (name.indexOf("quit") < 0)
        getRemoteAddress(name, serverName);
      } while (name.indexOf("quit") < 0);
      System.out.println ("Cancelled by user request.");
    } catch (IOException x) {x.printStackTrace ();}
  }

  static void printLocalAddress () {
    try {
      InetAddress me = InetAddress.getLocalHost ();
      System.out.println("My local name is:       " +
                   me.getHostName());
      System.out.println("My local IP address is: " +
                   toText(me.getAddress()));
    } catch (UnknownHostException x) {
      System.out.println ("I appear to be unknown to myself.
Firewall?:");
      x.printStackTrace ();
    }
  }

  static String toText (byte ip[]) {  /* Make portable for 128 bit
format */
    StringBuffer result = new StringBuffer ();
    for (int i = 0; i < ip.length; ++ i) {
      if (i > 0) result.append (".");
      result.append (0xff & ip[i]);
    }
    return result.toString ();
  }

  static void getRemoteAddress (String name, String serverName){
    Socket sock;
    BufferedReader fromServer;
    PrintStream toServer;
    String textFromServer;

    try{
      /* Open our connection to server port, choose your own port
number.. */
      sock = new Socket(serverName, 1565);

      // Create filter I/O streams for the socket:
      fromServer =
      new  BufferedReader(new
InputStreamReader(sock.getInputStream()));
      toServer   = new PrintStream(sock.getOutputStream());

      // Send machine name or IP address to server:
      toServer.println(name); toServer.flush();

      // Read two or three lines of response from the server,
      // and block while synchronously waiting:
      for (int i = 1; i <=3; i++){
      textFromServer = fromServer.readLine();
      if (textFromServer != null) System.out.println(textFromServer);
      }

      sock.close();
    } catch (IOException x) {
      System.out.println ("Socket error.");
      x.printStackTrace ();
    }
  }
}
```

Optional code: Thanks Pat Walsh

```
switch (args.length) {
 case 1:
   port=args[0];
   break;
 case 2:
   port=args[0];
   server=args[1];
 default:
   System.out.println("Usage: %java InetClient <port> <server>\n");
```