CS 254 - Computer Networks
Lab 4: Java Email Client

Due: Thursday, Feb 13 (class-time)

Purpose:

There are two parts to this lab. In the first part you will use `telnet` to open an SMTP connection and send an email message. And in the second part you will write a Java program that uses *sockets* to do the same thing.

Background:

Ordinarily one uses a program (client) to make a connection to and interact with a server, e.g. a web browser or an ftp client. However, if one knows the protocol, that is, the commands recognized, etc., then one can connect to a port using *telnet* and give commands to the server and receive responses from the server that listens on that port.

For example, *mail service* is provided at port 25 using *SMTP* – Simple Mail Transfer Protocol. SMTP recognizes commands such as `MAIL FROM:`, `RCPT TO:`, `DATA`, etc. Here is a script of a telnet session in which I connected to port 25 of `kant.compsci.sbu.edu` and provided the SMTP commands necessary to send mail to myself (what I typed appears in bold-face):

```
telnet kant.compsci.sbu.edu 25
220 kant.compsci.sbu.edu Microsoft ESMTP MAIL Service, Version: 6.0.3790.3959 ready
at Tue, 9 Feb 2010 15:06:33 -0500
HELO cs-lab06.compsci.sbu.edu
250 kant.compsci.sbu.edu Hello [149.76.128.107]
MAIL FROM: ska@compsci.sbu.edu
250 2.1.0 ska@compsci.sbu.edu....Sender OK
RCPT TO: andrianoff@sbu.edu
250 2.1.5 andrianoff@sbu.edu
DATA
354 Start mail input; end with <CRLF>.<CRLF>
This is a test message.
It was sent to you by telnetting to kant on port 25.
- ska
.
250 2.6.0 <KANTib0tsVJoYf54zDl00000004@kant.compsci.sbu.edu> Queued mail for
delivery
QUIT
221 2.0.0 kant.compsci.sbu.edu Service closing transmission channel

Connection to host lost.
```

Instructions:

1. Login to one of the PC's in Windows and open a DOS window. Telnet to port 25 of `kant.compsci.sbu.edu` and send mail to your bonaventure.edu mail account.

   Notes: The `HELO` command should identify the hostname of the lab computer you are working on. In the `MAIL FROM:` command provide your login id followed by `@compsci.sbu.edu`

   Further note: **Type carefully!!** Telnet is not forgiving of backspaces.

   Print a copy of your telnet session (by copying and pasting it into a Word or WordPad document) and turn it in with your lab. Also print a copy of the mail message you received and turn it in with your lab.

2. Write a Java program that sends an e-mail message.

   Open a socket to port 25 on kant.compsci.sbu.edu (Refer to the programs you wrote in Lab 2: Investigating Application Layer Protocols with telnet and An Introduction to Socket Programming with Java.) Your Java program will interact with the mail service using SMTP. Below is a summary of the important SMTP commands:

   `HELO host` Identifies the host from which the mail is originating

   `MAIL FROM: name` Name may be anything – it is not checked for validity, however, please use your CS e-mail address

   `RCPT TO: name` E-mail address of recipient

   `DATA` Allows entry of message, terminate message with '.' as only character on a line

   `QUIT` Disconnect

Here is an outline of the structure of the program:

- open a socket
- create a BufferedReader from the socket's input stream (you did this in Lab 2)
- create a PrintWriter from the socket's output stream (this follows the pattern of creating a BufferedReader, using an OutputStream and OutputStreamWriter instead of an InputStream and InputStreamReader). Add a "true" as second parameter for the PrintWriter constructor, this will "flush" the buffer following each println().
- alternate reading from the socket with writing to the socket (mimic the interaction from the telnet session)
- close the socket

Recommendations:

1) "Hardwire" the arguments to the socket constructor.

2) After creating the socket, display the IP address and port number of each end of the connection. (Review Lab 2 where you have done this before.)

3) To send a message to the mail server use the println() method of PrintWriter. You will find it helpful to display the messages you send to the mail server in the console as well.

4) Display the interaction in the console. In particular, display the messages received from the mail server.

(Hint: use

```
if ((line = br.readLine()) != null)
      System.out.println(line);
```

)

5) “Hardwire” the name of the local host in the HELO message. “Hardwire” your e-mail address in the MAIL FROM: message.
6) Use a separate println() for each line of the message to be sent and “hardwire” what is sent.
7) Once you have the program running correctly, modify the program to ask the user for the e-mail address of the recipient and the text of the e-mail message (a one-line message is sufficient since it is easy to do). The program should handle the other details. In particular, investigate how the program can determine the host name of the machine it is running on.
8) Document your program.

Once your program works correctly print a copy of an execution of the program (i.e., print the console) and print a copy of the e-mail that is received. Hand these in along with a copy of your Java program.

Extra Credit: Modify your program to allow the user to enter a multi-line e-mail message from the console. Turn in a copy of the execution, a copy of the email sent, and a copy of the program.

Extra Credit: Modify your program to allow the user to specify a subject line for the email. Turn in a copy of the execution, a copy of the email sent, and a copy of the program.

Help Policy:

Help Policy in Effect for This Assignment: Group Project with Limited Collaboration

In particular, you may discuss the assignment and concepts related to the assignment with the following persons, in addition to an instructor in this course: any member of your group; any St. Bonaventure Computer Science instructor; and any student enrolled in CS 254.

You may use the following materials produced by other students: materials produced by members of your group.