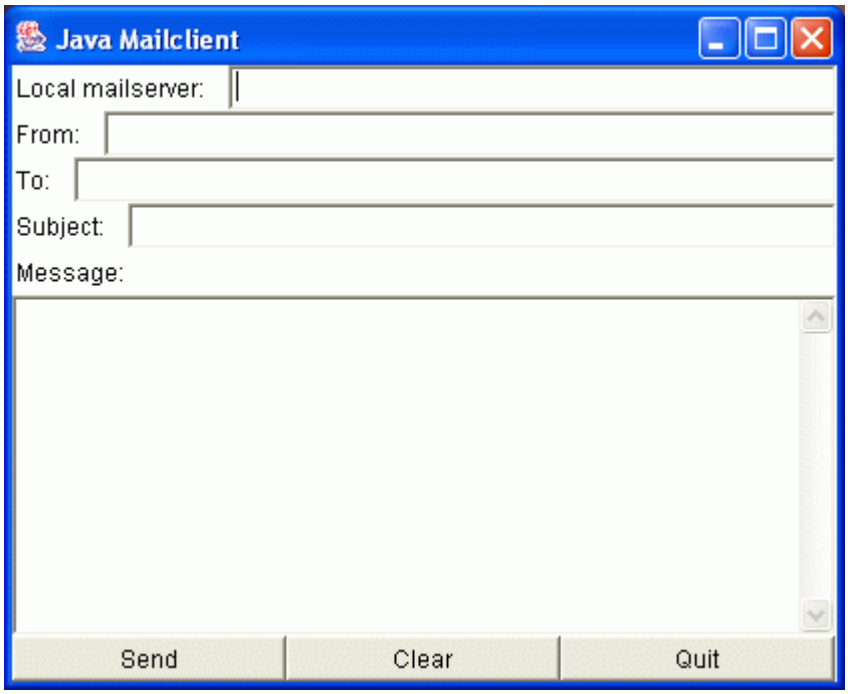
PA1

ASSIGNMENT DESCRIPTION

Your assignment is to create a simple GUI **Mail Transfer Agent (MTA)** that can send an email. Most of the program has already been written—it is up to you to fill in the missing pieces.*

DETAILS



Five java files have been provided for you:
MailClient.java – The user interface
Message.java – Mail message
Envelope.java – SMTP envelope around the Message
SMTPConnection.java – Connection to the SMTP server

The only file that requires substantial revisions is *SMTPConnection.java*. Comments throughout the source code are provided to guide you in making the appropriate changes. The changes involve adding the **Java socket programming** code and the **SMTP protocol** logic.

In completing this assignment, you should begin by familiarizing yourself with the codebase. Your first step should be to find where execution begins (in Java programs, execution always begins in *public static void main()*) and then follow the logic of the code chronologically. The program framework has been laid out neatly for you; understand the structure and work within it. You will find that much of the work is already done for you.

Next hone in on *SMTPConnection.java* and read through the comments. What role does this class play in the overall application? What pieces of functionality are missing?

Your first programming task should be to get the code to compile and run. Once it is up and running, you can start debugging. **Add *System.out.println()* statements containing all read input and written output so the program's execution can be easily followed.**

Once your code compiles, it is time to start adding the SMTP protocol logic. Research SMTP to find out which socket the mail server listens on, the sequence of commands in the send mail dialogue, and the meaning of the reply codes. You might start with the SMTP **Request for Comments** (RFC):

http://www.ietf.org/rfc/rfc0821.txt?number=821

Summary of steps:

1. Understand the flow of execution
2. Understand which pieces of functionality are missing
3. Get the program to compile
4. Make the socket connection to the mail server
5. Send an email to yourself
6. Add the enhancement

Helpful SMTP commands and reply codes:

Command	**Reply Code**
(after making the connection)	220
HELO	250
MAIL FROM	250
RCPT TO	250
DATA	354
QUIT	221

Appropriate GUI field values:

Field	**Value**
Local mailserver:	mail.cedarville.edu
From:	[firstname]@cedarville.edu
To:	[your Cedarville email address]
CC:	[your Cedarville email address]
Subject:	[up to you]
Message:	[up to you]

ENHANCEMENT

Add a "CC:" field to the GUI. Do research on SMTP to figure out the syntax for including an additional email recipient. Note, your application should behave like a real mail sending application. As you know, the CC field is optional, not required, when sending an email. You must code the logic to use and validate this field only when it is present.

GRADING

Working Code	75%
Well Written Code	10%
Working Enhancement	10%
Following Hand-in Instructions	5%

HAND-IN INSTRUCTIONS

Print out the following and hand it in at the beginning of class on the day it is due. Add headers between each partition describing what follows. The order is important.

1. Your *SMTPConnection.java* source code
2. A screen shot of your GUI, with all of the fields filled out except the "CC:" field
3. The console output from clicking the "Send" button on the screen from the above screen shot
4. A screen shot of your GUI, with all of the fields filled out including the "CC:" field
5. The console output from clicking the "Send" button on the screen from the above screen shot

Note, because you included *System.out.println()* statements, your console should contain every line written from the MTA and received from the mail server, with each line starting with "SENT: " or "RCVD: ".

Also, please keep all of your original source code in case it is needed for reference.

[*] This assignment concept, description, and source code is largely unmodified from *Programming Assignment 2: A Mail User Agent in Java* from Kurose and Ross's text, *Computer Networking: A Top-Down Approach*