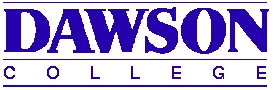
Java Application for Gmail

Version 1.9.3

August 25, 2020



Revisions

| Version | Primary Author(s) | Description of Version | Date Completed |
| --- | --- | --- | --- |
| 1.0 | Ken Fogel | This is the first version of the specification.  It is expected that this document will be revised as work on the project progresses and new features are discovered that would be required but were not in this first draft. | 05/09/23 |
| 1.1 | Ken Fogel | Default port numbers are added.  Additional feature of Reply To added. | 05/09/26 |
| 1.2 | Ken Fogel | Changed from Software Requirement to Software Design specification | 09/10/4 |
| 1.3 | Ken Fogel | Updated for 2012 | 12/10/15 |
| 1.4 | Ken Fogel | Update for 2014  JavaFX replaces Swing  Jodd Email (<http://jodd.org/doc/email.html>) replaces Java Mail  Defined message table and updated other table descriptions | 14/08/11 |
| 1.5 | Ken Fogel | Use Gmail as the SMTP/IMAP server | 14/08/19 |
| 1.6 | Ken Fogel | Updated email message structure to match Jodd classes | 14/09/10 |
| 1.7 | Ken Fogel | Corrected default port numbers for GMail | 14/09/15 |
| 1.8 | Ken Fogel | Update for 2015  Removed Contacts management | 15/08/24 |
| 1.8.1 | Julien Comtois | Correction of spelling, punctuation, and grammar errors | 15/08/27 |
| 1.9 | Ken Fogel | Jodd classes are now the basis of the SQL tables | 16/08/19 |
| 1.9.1 | Ken Fogel | Added Database name to list of configuration | 16/11/05 |
| 1.9.2 | Ken Fogel | Changed the number of required Gmail accounts to three from two.  Updated URLs for new Jodd locations  Added security down grading for GMail | 18/08/13 |
| 1.9.3 | Ken Fogel | Updated software versions.  Added more details to how Folders need to be managed.  Defined Help as using the WebView control and all help written in HTML. | 20/08/25 |

Contents.3

[1 Introduction 1](#_Toc49248655)

[1.1 System Overview 1](#_Toc49248656)

[1.2 Design Map 1](#_Toc49248657)

[1.3 Definitions and Acronyms 1](#_Toc49248658)

[1.4 Gmail Requirements 2](#_Toc49248659)

[2 Design Considerations 3](#_Toc49248660)

[2.1 Assumptions 3](#_Toc49248661)

[2.2 Constraints 3](#_Toc49248662)

[2.3 System Environment 3](#_Toc49248663)

[3 Architecture 4](#_Toc49248664)

[3.1 User Interface Controller 4](#_Toc49248665)

[3.1.1 Folder Tree Module 4](#_Toc49248666)

[3.1.2 Mail List Module 4](#_Toc49248667)

[3.1.3 Message Display Module 4](#_Toc49248668)

[3.1.4 Configuration Composition Module 4](#_Toc49248669)

[3.1.5 Message Composition Module 4](#_Toc49248670)

[3.1.6 Attachment Manager Module 4](#_Toc49248671)

[3.2 Database Controller 4](#_Toc49248672)

[3.2.1 Mail Storage Module 5](#_Toc49248673)

[3.2.2 Folder Storage Module 5](#_Toc49248674)

[3.3 Mail Controller 5](#_Toc49248675)

[3.3.1 SMTP Module 5](#_Toc49248676)

[3.3.2 IMAP Module 5](#_Toc49248677)

[3.4 Configuration Controller 5](#_Toc49248678)

[3.5 Help Module 5](#_Toc49248679)

[4 High Level Design 6](#_Toc49248680)

[4.1 User Interface Controller 6](#_Toc49248681)

[4.1.1 Folder Tree Module 6](#_Toc49248682)

[4.1.2 Mail List Module 6](#_Toc49248683)

[4.1.3 Message Display Module 6](#_Toc49248684)

[4.1.4 Configuration Composition Module 6](#_Toc49248685)

[4.1.5 Message Composition Module 7](#_Toc49248686)

[4.1.6 Attachment Manager Module 7](#_Toc49248687)

[4.2 Database Controller 7](#_Toc49248688)

[4.2.1 Mail Storage Module 7](#_Toc49248689)

[4.2.2 Folder Storage Module 7](#_Toc49248690)

[4.3 Mail Controller 7](#_Toc49248691)

[4.3.1 SMTP Module 7](#_Toc49248692)

[4.3.2 IMAP Module 8](#_Toc49248693)

[4.4 Configuration Controller 8](#_Toc49248694)

[4.5 Help Controller 8](#_Toc49248695)

[5 Data Structures 9](#_Toc49248696)

[5.1 Configuration 9](#_Toc49248697)

[5.2 Folder Table 9](#_Toc49248698)

[5.3 Email Messages 9](#_Toc49248699)

[5.3.1 Jodd Email Class 10](#_Toc49248700)

[5.3.2 Jodd EmailAttachment Class 10](#_Toc49248701)

[5.3.3 Jodd EmailAddress Class 10](#_Toc49248702)

# Introduction

The purpose of this document is to outline the design specification for an email client program for Google’s Gmail. This program will be developed by students in the Dawson College Computer Science Technology course 420-517-DW Software Development Project - Java III.

## System Overview

The software design in this document is for a program called the Java Application for Gmail or JAG for short. Its purpose is to carry out the tasks associated with an email program. It will allow its user to create, send, receive and store email messages by communicating with a standard email server over the Internet. The messages created will be in HTML or plain text format. Messages received may be in either HTML or text format.

## Design Map

This JAG design is based on a subset of the features of an existing email client applications such as Microsoft Outlook. This document will describe the modules of the program and their functionality. It will outline the necessary data structures.

## Definitions and Acronyms

E-mail is sent and received using a variety of different protocols of which this program will use:

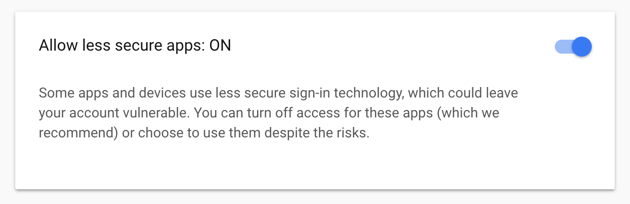
* **SMTP** - Simple Mail Transfer Protocol for sending
* **POP** – Original Post Office Protocol for receiving, seldom used now
* **IMAP** - Internet Message Access Protocol for receiving emails, current standard
* **TLS** – Transport Layer Security encryption technology used on the internet. Previously known as SSL, Secure Socket Layer, an obsolete protocol.
* **MIME** – Multi-Purpose Internet Mail Extension. All email is transmitted as plain text. Mime is responsible for the conversion of binary information such as images and encrypted data into plain text.

Some important definitions include:

* **address** - electronic mail address
* **attachment** - a file, either text or binary, that is sent as part of the message
* **bcc, blind carbon copy** - a copy of the message sent to other recipients whose names are not shown to the recipients
* **cc, carbon copy** - a copy of the message sent to other recipients whose names you can see
* **forward** – create a new message containing the body of an existing message
* **mail** - refers to electronic mail
* **mail folder** - a folder that holds specific mail messages.  eg. Inbox, Sent, Trash
* **mailing list** - a list of recipients who all receive the message
* **port** - a single internet connection can support 64K channels of communication numbered from 1 to 64K. Gmail uses 993 for IMAP and 465 for SMTP
* **recipient** - the person or persons who receives the mail.
* **reply** - a response to a received message sent to the sender
* **reply all** – a response to a received message that is sent to everyone in the To and CC fields but not the BCC field
* **sender** - the person who sent the mail
* **snail mail** - standard post office mail (the kind with a stamp)

## Gmail Requirements

You will need to create three Gmail accounts. One will be used for sending, one will be used for receiving emails and one will be used for testing cc, bcc and multiple recipients. These accounts are only required for the development of the system. When setting up these accounts you need to turn on *Allow less secure apps* in the security settings for each account.



# Design Considerations

## Assumptions

The program will be written using the standard features of the Java language. Only third party libraries named in this document, JavaFX may be used. **JUnit** testing must be employed.

## Constraints

The primary constraint is that the system must be completed by the end of week ten. To this end you must look for simplicity rather than complexity. Features not explicitly in this document and not mentioned in class should be avoided unless they are recognized as critical to the operation of the program. Such changes must be submitted to your instructor for approval and then added to this document.

## System Environment

The programming language will be Java version 14 and the GUI will use JavaFX. The Gluon Scene Builder, a visual editor, will be used to create much of the GUI.

The JAG will depend on an external DBMS for its message store. The DBMS will be MySQL. Additional information on MySQL and downloads of the program can be found at http://www.mysql.com.

The JDBC driver will be Connector/J. Additional information on this library can be found at http://dev.mysql.com/downloads/connector/j/.

The JAG is not dependent on any specific operating system. It should be able to run on any computer that supports Java.

The library for accessing an email server will be Jodd Email (https://jodd.org/email/).

# Architecture

The architecture of this program is based on a system of modules each of which provides a specific aspect of the system. Each of these modules will be constructed in such a way as to minimize the coupling with other modules. This will allow each module to be tested as they are written without depending on other modules.

## User Interface Controller

The program will present a number of user interfaces for the different tasks a user will carry out. This controller will display the appropriate interfaces as required.

### Folder Tree Module

The folder tree will display a tree of the folders in a JavaFX TreeView..

### Mail List Module

The mail list will present in a JavaFX TableView the messages associated with the folder selected in the folder tree.

### Message Display Module

The message display will present, in an HTMLEditor control set to display only, the body of a message selected in the mail list.

### Configuration Composition Module

The configuration composition will present a form in which the details of the program’s configuration can be entered or edited.

### Message Composition Module

The message composition will present a JavaFX HTMLEditor in which an email can be composed.

### Attachment Manager Module

Attachments that are added to a new message or received in a message are processed by this module.

## Database Controller

All interactions with the database will be handled by this controller. Other modules will not access the database. Instead, they will send messages to the manager who will in turn determine which sub module will handle the request.

### Mail Storage Module

The mail storage will be responsible for all message related tasks that require access to the DBMS.

### Folder Storage Module

The folder storage is responsible for maintaining the list of folders names.

## Mail Controller

All interactions with the mail server will be handled by this controller.

### SMTP Module

This module will carry out SMTP conversations with the mail server for sending emails.

### IMAP Module

This module will carry out IMAP conversations with the mail server for receiving emails.

## Configuration Controller

This controller will maintain the configuration data necessary for the operation of the system. It will provide the appropriate information to other modules that make requests to it.

## Help Module

This module is responsible for displaying help screens. Every possible action a user can make must be described here using the JavaFX WebView control.

# High Level Design

## User Interface Controller

This controller will be responsible for all tasks and events associated with presenting the user interfaces. When the program begins it will construct all of its sub modules. It will manage the main menu of the application.

### Folder Tree Module

This module will create a panel that uses a Tree that shows the list of folders as retrieved from the folder table of the DBMS. Users will be able to create new folders or delete existing folders. When this occurs the panel must refresh its display.

This module will interact with the Database Controller. It will send messages to the Mail List module to inform it of changes in the choice of current folder.

### Mail List Module

This module will create a panel that uses a TableView to display a list of emails associated with the selected folder in the Folder Tree Module. Each row in the table will represent a single email. Users will be able to select single rows in the table. This selection will determine which message will be displayed in the Message Display Module. An email selected in the table can be replied to, forwarded, or deleted. Single rows can be dragged to the Tree in the Folder Tree Module to associate the message with a different folder.

This module will receive messages from the Folder Tree Module telling it which message to display based on the user’s choice in the tree. It will determine which leaf in the tree is chosen during a drag and drop operation so that the folder assigned to the email can be changed. It will call upon the Database Controller to get the data to display and to update records. It will call upon the Message Composition Module when a user forwards or replies to a message.

### Message Display Module

This module will display the body of the message selected in the Mail List Module. It will use an HTMLEditor control that can display both plain text and HTML messages. This is a read only display. This module will call upon the Database Manager to retrieve the message it will display. Whenever the user selects a message in the Mail List Module a message will be received from that module.

### Configuration Composition Module

This module will present a form consisting of text fields that represent the necessary information for configuration. This module will interact with the Configuration Controller. When the program is run the first time this module will be called upon before the program can do anything else. This is a custom form that you design.

### Message Composition Module

This module will present a form consisting of text fields for the purpose of entering all the necessary addressing information. For the To, CC, and BCC fields the user will be permitted to enter more than one email address and information for these fields entered directly by the user. There will be a subject field and the message body field.

If the message is based on a request for a Reply To, Reply All or Forward then the body of the original message will appear in the new message. The user will be able to cancel the message so as not to send it. The action of sending the message will result in the message being assigned to the outbox folder.

An HTMLEditor will be used and based on the choice of the user it will create either a plain text or HTML text message. Attachments can be added to the message.

### Attachment Manager Module

This module will use file dialogs to either retrieve a file from the file system to be attached to a message or save an attachment from a received email. Attachments are initially stored in the database so saving it to disk requires retrieving the file from the database.

## Database Controller

This controller will be responsible for all communication with the RDBMS. Other modules will send messages that will be examined to determine what type of database access is required.

### Mail Storage Module

This module will either retrieve or update mail records. The messages will be delivered in a collection of beans. Even if only a single message is involved it will still be stored in a collection. This module will interact with the Mail List Module, Message Display Module, and Message Composition Module.

### Folder Storage Module

This module will either retrieve or update records in the folder records. The Folder Tree Module will call upon this module to display its tree or make changes to it. Messages from the Table display are dragged to a folder name in the Tree to change its folder name. New folder names are added to the tree and then messages are assigned to it.

## Mail Controller

This controller will use the Jodd Email API to interact with the mails server. Depending on the operation it will call upon either the SMTP or IMAP module to carry out the requested tasks.

### SMTP Module

This module will carry out the necessary tasks to retrieve email. It will be able to interact with Google Gmail’s server that requires authentication and encryption. When it is successful it will send a message to Database Manager to have the folder field of the message sent changed to the ‘sent’ folder. Any errors that occur must be brought to the attention of the user.

### IMAP Module

This module will carry out the necessary tasks to send mail. It will be able to interact with Google Gmail’s server. When a message is received the module will interact with the Database Manager to store the message with the folder set to ‘inbox’. Any errors that occur must be brought to the attention of the user.

## Configuration Controller

This controller is responsible for the text file containing the initialization information for the program. The information will be stored using the Properties library. It will interact with the Configuration Composition Module, the Database Controller, and the Mail Controller.

## Help Controller

This module will use help pages written in HTML and displayed with the JavaFX WebView control.

# Data Structures

This section identifies data structures used in the program. All tables must have a primary key.

## Configuration

This is the set of properties for the configuration of the program. It will be stored in a text file and accessed with the Properties API of Java.

* The user’s name
* The user’s email address which is also the user name for Gmail
* The user’s Gmail password
* The URL of the IMAP server
* The URL of the SMTP server
* The IMAP port number (default 993)
* The SMTP port number (default 465)
* The URL of the MySQL database
* The database name
* The port of the MySQL database (default 3306)
* The user name for the MySQL database
* The password for the MySQL database

## Folder Table

A folder is simply a name that categorizes an email. The default folder is INBOX and all emails are assigned this folder name when received. A user must be able to change this name to one that already exists in the Folder Table or add a new one. Messages can then be displayed based on their folder name by selecting the folder from the tree control.

* Primary Key
* Folder Name

## Email Messages

You will map each of the necessary Jodd email data classes to an SQL table. You can find the Jodd documentation at https://jodd.org/email/. Here are the specific Jodd classes that must be mapped.

### Jodd Email Class

This is the primary class that contains all the data of an email. You can see its API at https://oblac.github.io/jodd-site/javadoc/. You will need to map every data field of the Email class to an SQL table.

### Jodd EmailAttachment Class

This is the class that contains attachments. There are two types of attachments. One is an attachment that appears as a file you can download. The second is an embedded attachment such as when an image is included in the body of an html message. It is found as a field in the Email class. Attachments will be stored in the database as a BLOB type.

### Jodd EmailAddress Class

This class is used for email addresses. The address consists of one string for the user name and a separate string for the email address. All addresses must be validated using the RFC2822AddressParser class in Jodd.