Benutzerdokumentation

In diesem Abschnitt der Benutzerdokumentation werden die Funktionalitäten des Mail-Clients und deren Benutzung näher erläutert.
Startet man das Programm lädt zuerst die Startseite, auf welcher zentralisiert die beiden Textfelder „E-Mail“ und „Passwort“ zu sehen sind, in denen man seine Login-Daten eingeben kann. Darunter befinden sich weitere Optionen. Als erstes gibt es die Möglichkeit zwischen den Beiden E-Mail Protokollen iMap und Pop3 zu wählen. Mit IMAP (Internet Message Access Protocol) holt ein E-Mail-Programm nur die aktuell benötigten Informationen vom Server. Die Dateien bleiben am Server bestehen. Pop3 (Post Office Protocol) hingegen holt von einem EMailProgramm die E-Mails vollständig vom Server. Die Dateien sind danach nicht mehr am Server, sondern nur noch in Ihrem E-Mail-Programm gespeichert. Beim nicht auswählen dieser Option wird automatisch iMap verwendet. Des weiteren kann man bei dem Label „Remember me“ einen Haken setzen um beim nächsten Login-Vorgang die Login-Daten automatisch ausfüllen zu lassen. Beim bestätigen des „Login“ Buttons meldet sich das Programm mit den eingegebenen Login-Daten beim jeweiligen E-Mail Anbieter an. Durch das Betätigen des Buttons „Forgot Login Details“ wird man auf die Website: „https://support.google.com/accounts/answer/7682439?hl=de“ weitergeleitet auf der man dann sein Gmail-Konto wiederherstellen kann. Am oberen Rand gibt es noch die Möglichkeit den sogenannten Party-Modus auszuwählen. Dieser färbt den Hintergrund abwechselnd in verschiedene Farben und lässt sich auch nach der Anmeldung noch einstellen, sodass der Nutzer auch beim verwenden seines Mail-Clients auf fröhliche Gedanken kommt **(Nutzer mit Epilepsie sollten diesen Modus nicht aktivieren!).**

Nach der Anmeldung lädt der Home Bildschirm die neusten Mails in den Posteingang. Links unten in der Ecke wird dem Nutzer angezeigt mit welcher E-Mail-Adresse er angemeldet ist und es gibt direkt da drunter die Möglichkeit sich wieder abzumelden. Die Mails werden im Posteingang farblich akzentuiert mit Absender, Betreff und Datum angezeigt. Die Mails werden zeitlich sortiert in den Posteingang geladen, sodass ganz oben immer die neuste Mail zu finden ist. Wählt man eine Mail mit einem Klick aus, so wird diese komplett geladen. Nun kann man die Nachricht lesen und diese in den Ordner Archiv, „archivieren“ oder Papierkorb, „löschen“, verschieben. Während der Programmausführung aktualisiert sich der Mail-Client regelmäßig von alleine, sodass Mails, die während der Benutzung ankommen, auch direkt in den Posteingang geladen werden.
Die verschiedenen Postfächer sind links in einer Spalte untereinander angeordnet. Neben jedem Postfach steht wie viele Mails sich in dem jeweiligen Postfach befinden. Betätigt man den Button „Nachricht verfassen“ wechselt die Oberflächenansicht und es gibt nun die Möglichkeit eine E-Mail zu schreiben. In den oberen Beiden Textfeldern gibt man Empfänger und Betreff an, im großen freien Teil, die Nachricht. In der unteren Leiste kann man nun auswählen ob die Mail direkt versendet „Nachricht senden“, oder in den Entwürfe Ordner „Nachricht speichern“ verschoben werden soll. Der Ordner Postausgang listet die Mails auf, die gerade verschickt werden und noch laden. Im Ordner Archiv werden die archivierten Mails angezeigt. Durch das Archiv kann man den normalen Posteingang quasi ein wenig aufräumen. Im Ordner Entwürfe befinden sich bereits verfasste Mails, die aber noch nicht abgeschickt worden sind. Im Ordner Papierkorb befinden sich die aus dem Posteingang einem Anderem Ordner gelöschten Mails. In den Spam Ordner werden automatisch Spam Mails reingeladen. Klickt man auf “Alle E-Mails” werden alle Mails aus allen Ordnern ausgelistet.

Installationsdokumentation

Hardware/ Systemanforderung: 2 GB RAM, jede moderne CPU, Internetverbindung
Software: IntelliJ, Windows 8 / mac OS 13 oder neuer, Java Development KIT (JDK)
Standardbibliotheken: Gson, JavaFx
Laufzeitsysteme: Java Development Kit 11 oder neuer

Beschreibung Installationsprozess:
Um Java FX zu nutzen, benötigen Sie ein Java Development Kit zwischen Version sieben und zehn von Oracle. Falls Sie eine andere Version installiert haben, können Sie die JavaFX-Bibliothek auch als externe Bibliothek hinzufügen. Dies ist unter anderem mittels Gradle möglich. Sie sollen dies stets mit Gradle und Java Development Kit 11 machen.

Der Benutzer muss zu Beginn das Programm herunterladen und diese auf eine IDEA seiner Wahl öffnen. Es ist wichtig, dass man ein Gradle-Projekt erstellt. Im nächsten Schritt müssen die Bibliotheken eingebunden werden. Hier muss der Benutzer in seiner IDEA die build.gradle öffnen und die dependencies für Gson hinzufügen. Dabei muss dieser Code hinzugefügt werden “implementation 'com.google.code.gson:gson:2.8.6' ”. Zudem muss in der build.gradle unter den Dependencies die Mail-Api eingefügt werden mit “compile 'javax.mail:mail:1.4.7' ”. Um das Programm lauffähig zu machen, muss das Programm mit run.gradle ausgeführt werden. Dies geschieht in dem man z.B. auf IntelliJ oben links auf “edit configurations” klickt und dann auf das “+” klickt und Gradle auswählt. Schließlich muss der Task “run” heißen und man muss dringend unter “Gradle Project” das Projekt auswählen, indem gerade gearbeitet wird . Bekommt man keine weitere Fehlermeldungen, kann das Programm ausgeführt werden. Dabei muss der Nutzer auf das “Play” klicken. Dann startet das Programm und man kann sich mit seiner Mail anmelden.

Systemdokumentation

Beschreibung der verwendeten Tools, Frameworks und Klassenbibliotheken

Tools:

Scene builder:
Zum grafischen und damit einfacheren Arbeiten mit Designelementen aus JavaFx. Man sieht sofort wie die GUI aussieht, und muss sich nicht mit Zahlen rumhantieren.

Intellij:

Klassenbibliotheken:

Gson:
Für das arbeiten mit Json-Dateien, insbesondere das konvertieren von Json-Dateien zu Java-Klassen.

JavaFx:
Stellt schon wichtige grafische Benutzer Elemente, wie Buttons oder Eingabefelder zur verfügung.

Framework:

Gradle:
Zum einbinden und verwalten von den verschiedensten Libraries, wie Gson oder JavaFx.

Java-Mail-API:
Zum Arbeiten mit schon vorgegebenen notwendigen Methoden für das Schreiben und Empfangen von Emails.

Beschreibung der Softwarestruktur und der einzelnen Bestandteile

Login und Startseite

Die zwei wichtigsten Klassen sind “Login” und “Startseite”, da in diesen Klassen alle Bedienelemente vorhanden, referenziert oder instanziiert werden.
Startet man das Programm so startet man die Klasse “Login”, in der man seine Google-Anmeldedaten eingeben kann. Die grafischen JavaFx Bedienelemente von Login sind hierbei alle in der Klasse selbst und es wird auf keine andere JavaFx-Klasse zugegriffen. Ist der Login erfolgreich schließt sich das Login-Fenster und es öffnet sich das Startseitenfenster.
In diesem befindet sich der Knopf zum abmelden, wird dieser gedrückt schließt sich das Startseitenfenster und es öffnet sich wieder das Login Fenster.

JavaFx

Alle Java-Klassen, die ein Teil der grafischen Bedienoberfläche sind besitzen eine fxml-Datei in denen der JavaFx-spezifische Code steht. Dieser Code wurde zum Großteil durch die Nutzung von Scene-Builder (s.o.) automatisch generiert.

Hier eine Auflistung solcher Klassen:

- EmailKnopf: Ein Button-Objekt, welches die E Mail grafisch darstellt, damit man die Mail mit einem Knopfdruck öffnen kann
- EmailSchreiben: Das grafische Interface, in dem man eine Email verfassen kann
- MailAnzeige: Das grafische Interface, in dem man eine Email lesen kann
- Login: s.o./s.u.
- Startseite: s.o./s.u.

Die Klasse Startseite besitzt mehrere Knöpfe, die meisten davon sind für das Öffnen der jeweiligen EMail-Ordner da. Zu diesen Knöpfen gehören Posteingang, Archiv, Papierkorb, Postausgang und Entwürfe. Durch das Drücken dieser Knöpfe wird durch die Methode “MailsInFensterAuflistenServer” eine Instanz von der Klasse “GetMails” erstellt.
GetMails ist dafür zuständig eine Liste der Mails aus dem ausgewählten Ordner zu übergeben. Entweder kann man sich eine direkte Liste aus dem lokalen Speicher oder eine Liste, die neue Emails vom Server direkt der Liste hinzufügt, übergeben lassen.

Emails verschieben

Wenn man auf eine Email klickt öffnet sich das “MailAnzeige”-Interface, welches zwei Knöpfe erzeugt, die vorher nicht da waren. Ein Knopf bietet die Möglichkeit die Nachrichten zu archivieren, der andere bietet die Möglichkeit die Nachricht in den Papierkorb zu verschieben.
Um Mails zu verschieben, gibt es die Klassen “ArchivMuell” und “Muell”, die beide dafür da sind die Nachricht auf Serverebene und auf lokaler Speicherebene zu verschieben, nur haben beide Klassen andere Zielordner, in die verschoben werden soll. “ArchiMuell” verschiebt die Mail in den Archiv-Ordner und “Muell” verschiebt die Mail in den Papierkorb-Ordner.
Beide Klassen werden in der Klasse “Startseite” aufgerufen.

Json-Dateien

Damit die Emails in einem einfachen Format auf dem lokalen Rechner gespeichert werden können, werden alle Mails, die wir aktuell vom Server geholt haben, in Json-Dateien gespeichert. Wir benutzen das Json-Format zur Speicherung der Mails, damit wir diese mithilfe der Gson-Klasse wieder zurück in Java-Objekte konvertieren können, um auf diesen dynamisch zu arbeiten.

Um die Anmeldedaten des Nutzers zu speichern benutzen wir ebenfalls eine Json-Datei, welche auch auf Wunsch hin, je nachdem, ob bei der Checkbox “Anmeldedaten speichern” ein Häkchen gesetzt wurde oder nicht, beim abmelden aus der Json-Datei gelöscht oder bleibt gespeichert.

Die Json-Dateien sind lokal auf dem Rechner gespeichert.
Meldet man sich mit einer anderen Email-Adresse an werden die Json-Dateien der Email-Ordner und der Anmeldedaten überschrieben.

Emails versenden

Will man eine Email versenden, gibt es dafür in der Startseite den Knopf “Nachricht verfassen”, welcher auf der Anzeigefläche ein dafür vorgesehenes Interface öffnet. Das Interface besitzt drei Textfelder, eines für den Empfänger, eines für den Betreff und eines für die Nachricht selber. Sobald man dabei auf den Knopf “Nachricht Senden” klickt wird, sofern keine Fehler auftreten, wie z.B. dass die Empfänger-Adresse nicht existiert, wird die Nachricht versendet und es erscheint das Label “Nachricht wurde gesendet” in grüner Schrift unten mittig. Die Nachricht wird außerdem unter Postausgang gespeichert.

Netzwerk

Alle Netzwerkoperationen werden über die Java-Mail-API geregelt. Man kann im Login Fenster noch auswählen, ob man zum Empfangen der Mails iMap oder Pop3 als Sicherheitsprotokoll benutzen möchte.

Projektdokumentation

Entwicklungsphasen:

1. **Login: Fenster/fxml**
    - Wir mussten uns mit Scene-Builder vertraut machen
    - Wir mussten uns mit der Funktionsweise von JavaFx vertraut machen
    - Wir haben die erste Szene erstellt und erste Bedienelemente eingebaut wie:
        i. Knöpfe
        ii. Eingabefelder
        iii. Checkboxen
2. **Login: Anmeldung bei Gmail (POP3 und iMap)**
    - Zunächst hatten wir Schwierigkeiten mit den Befehlen der Java-Mail-API eine Verbindung zu den Google-Servern aufzubauen
    - Nach einigen Internetrecherchen haben wir es schließlich geschafft Mails von Gmail zu empfange und in der Konsole zu printen
    - Entstehung der Klasse “GetMails”
3. **Login: Anmeldedaten in ein Json Objekt speichern und Auslesen**
4. **Login: Startseite nach Anmeldung öffnen**
    - Wir hatten Probleme damit das Login-Fenster zu schließen nachdem sich die Startseite geöffnet hat → haben wir gelöst, indem wir die Startseite in der gleichen Stage laden, also ins gleiche Fenster öffnen.
    - Entstehung der Klasse “Startseite”
5. **Startseite: Fenster/fxml, Seiten menü erstellen, und ein Feld indem sich**
    - Entstehung von Knöpfen “Nachricht verfassen”, “Abmelden” (noch ohne Funktion), “Posteingang” (noch nicht funktionstüchtig) und “Alle Mails” (noch nicht funktionstüchtig)
6. **Startseite: Abmeldung (= Anmeldedaten löschen wenn sie nicht gespeichert werden sollen)**
    - Beim Anmelden wird in der AnmeldeDaten.json ein entsprechender boolean mit gespeichert
7. **Startseite: Mails aus Posteingang bei Gmail holen und in Json schreiben**
    - Da die Mails als Message-Objekt übergeben werden hatten wir Probleme die Nachricht lesbar anzuzeigen, da die meisten Nachrichten Multiparts enthielten → Haben wir durch die Methode “getTextFromMultipart” gelöst, die entsprechende Hilfsmethoden aus der Java-Mail-API benutzt
8. **Startseite: Mails aus Posteingang in Fenster auflisten**
    - Zunächst hatten wir Probleme mit der Formatierung der Schrift auf den E- Mail-Knöpfen → Haben wir gelöst indem wir den Knöpfen zusätzliche Labels für die informationen hinzugefügt haben (Absender, Betreff, Inhalt)
9. **Startseite: Laden der Mails effizienter machen**
    - Abbruch des Mail ladens, sobald eine geladene Mail der neusten Mail in der Json entspricht
    - Dann die neue geladenen Mails, und die in der Json gespeicherten Mails auflisten

10. **Startseite: Mailanzeige öffnen wenn ein Element der Mail-Liste ausgewählt wird**
    - Entstehung der Methoden **“**MailsInFensterAuflistenServer” und “mailButtonGenerieren”
    - Wir hatten Schwierigkeiten den Email-Knopf so zu generieren, dass dieser die Daten der entsprechenden Mail der Mail-Anzeige übergibt → Haben wir gelöst indem wir ein FxmlLoader-Objekt mit der MailAnzeige.fxml erstellt haben und anschließend ein neuses Java-MailAnzeige-Objekt erstellt haben, welches wir mit dem FxmlLoader.load initialisiert haben, dannach konnten wir mit MailAnzeige.setValues alle Werte der Mail übergeben
11. **Startseite: Mails schreiben und über Gmail senden**
    - Eine neue FXML welche die Eingabe von Empfänger, Betreff und Inhalt ermöglicht wurde erstellt, und daraus eine Mail erstellt, die abgeschickt wird.
12. **Startseite: Mails aus den Ordnern: Postausgang, Archiv, Entwürfe, Papierkorb, Spam bei Gmail holen, und Auflisten.**
    - Die Methode “GetMails” wurde erstellt, mit der sich die Mails aus einem übergebenen Ordner holen lassen
13. **Startseite: Mails löschen / Archivieren aus dem Posteingang**
    - Löschung von Mail auf Server Seite, und Lokal in der Json
14. **Startseite: Mails als Entwurf speichern, und Entwürfe im Fenster zum Mail Senden aufrufen**
15. **Startseite: Die gespeicherten Mails löschen, wenn sich ein anderer Nutzer anmeldet**
16. **Design Elemente hinzugefügt (Party Mode, Farbverlauf für die Buttons)**
    - Gif als Hintergrunde wurde erstellt und eingebunden
    - Methode zur generation von Farbcodes für die Buttons wurde erstellt.
17. **Feinschliff**

Zuteilung der Aufgaben:

Wir haben uns immer in der gesamten Gruppe getroffen, und über “Code with me” zusammen am Projekt gearbeitet, lediglich das Farbdesign wurde von Johannes alleine gemacht.