MyContacts

Design

Definizione delle Classi

Le classi presenti all'interno del nostro progetto sono:

- Contatto
- Rubrica
- RubricaPreferiti
- GestoreOperazioniRubrica
- MyContacts
- MainViewController
- LeftViewController
- RightView1Controller
- RightView2Controller

Oltre alle classi sono presenti anche dei file FXML per la modifica del layout dell'applicazione:

- MainView
- LeftView
- RightView1
- RightView2

Descrizione funzionale delle classi

`Contatto`

La classe `Contatto` rappresenta un'entità all'interno di un sistema di gestione contatti. Ogni oggetto di questa classe conserva informazioni relative a un contatto, inclusi dati anagrafici (nome, cognome) e informazioni di comunicazione (numeri telefonici e indirizzi email). La classe prevede anche un attributo booleano per identificare se un contatto è stato contrassegnato come "preferito".

Attributi

- `nome (String)`: Rappresenta il nome del contatto.
- `cognome (String)`: Rappresenta il cognome del contatto.
- `numeri (List<Integer>)`: Una lista di numeri telefonici, che consente di memorizzare fino a 3 numeri associati al contatto.
- `email (List<String>)`: Una lista di indirizzi email, che permette di memorizzare fino a 3 email associate al contatto.
- `preferito (boolean)`: Indica se il contatto è stato aggiunto ai "preferiti" dell'utente (true se preferito, false altrimenti).

Metodi

- **Metodi di Accesso (Getter e Setter)**
  - `String getNome()`: Ottiene il nome del contatto.
  - `String getCognome()`: Ottiene il cognome del contatto.
  - `List<Integer> getNumeri()`: Ottiene la lista dei numeri di telefono.
  - `List<String> getEmail()`: Ottiene la lista degli indirizzi email.
  - `boolean isPreferito()`: Ottiene lo stato "preferito" del contatto.

- `void setNome(String nome)`: Imposta il nome del contatto.
- `void setCognome(String cognome)`: Imposta il cognome del contatto.
- `void setNumeri(List<Integer> numeri)`: Imposta la lista dei numeri di telefono.
- `void setEmail(List<String> email)`: Imposta la lista degli indirizzi email.
- `void setPreferito(boolean preferito)`: Imposta lo stato "preferito" del contatto.

- **Metodi di Confronto e Ordinamento**
  - `int compareTo(Contatto o)`: Implementa il confronto per ordinare i contatti (prima per cognome e poi per nome).
  - `boolean equals(Object obj)`: Verifica se due contatti sono uguali. Due contatti sono considerati uguali se hanno lo stesso nome, cognome, numeri e email.

- **Metodi di Gestione del Contatto**
  - `void modificaContatto(Contatto c)`: Modifica e informazioni di un contatto con i dati del contatto passato come parametro.
  - `void switchPreferiti()`: Inverte lo stato "preferito" di un contatto.
  - `boolean contattoValido()`: Verifica se un contatto è valido in base ai dati.

MyContacts

La classe `MyContacts` è progettata per avviare l'interfaccia grafica dell'applicazione. Si occupa di caricare il file FXML per l'interfaccia utente,

visualizzarlo in una finestra e gestire il cambiamento dinamico del contenuto della scena.

Attributi

- `scena (Scene)`: Rappresenta la scena attuale dell'applicazione.

Metodi

- **Metodi di Avvio e Gestione della Scena:**
  - `void start(Stage stage)`: Istanzia e visualizza la schermata principale. Il controllore non viene istanziato manualmente, ma viene caricato tramite il loader FXML.
  - `Parent loadFXML(String fxml)`: Carica un file FXML, che definisce l'interfaccia utente della finestra principale. Utilizza `FXMLLoader` per leggere il file FXML nella directory delle risorse.
  - `void main(String[] args)`: chiama la funzione di sistema `launch()`, che avvia l'applicazione.

Rubrica

La classe `Rubrica` gestisce l'elenco principale dei contatti, con operazioni fondamentali per la manipolazione di questi ultimi, come l'aggiunta, la rimozione e la verifica dello stato della rubrica.

Attributi

- `elenco (Set<Contatto>)`: Rappresenta l'insieme dei contatti principali, utilizzando un `TreeSet` per garantire l'ordinamento alfabetico(per cognome).

- `elencoPreferiti (RubricaPreferiti)` : Contiene un'istanza della classe `RubricaPreferiti`, che gestisce separatamente i contatti preferiti.

Metodi

- **Costruttore**
  - `Rubrica()`: Inizializza l'attributo `elenco` ed `elencoPreferiti` come `TreeSet`.

- **Metodi di Accesso**
  - `Set<Contatto> getElenco()`: Restituisce l'elenco di tutti i contatti presenti nella rubrica.
  - `RubricaPreferiti getElencoPreferiti()`: Restituisce l'oggetto responsabile della gestione dei contatti preferiti.

- **Metodi di Gestione dei Contatti**
  - `void addContatto(Contatto c)`: Aggiunge un contatto alla rubrica e genera un'eccezione se il contatto è `null`.
  - `void removeContatto(Contatto c)`: Rimuove un contatto dalla rubrica e genera un'eccezione se il contatto è `null`.
  - `void resetRubrica()`: Svuota completamente la rubrica, rimuovendo tutti i contatti.
  - `boolean isRubricaVuota()`: Verifica se la rubrica è vuota e restituisce un valore booleano.

- **Gestione dei Contatti Preferiti**
  - `void aggiungiAPreferiti(Contatto c)`: Aggiunge un contatto alla lista dei preferiti e genera un'eccezione se il contatto non è presente nella rubrica.
  - `void rimuoviDaPreferiti(Contatto c)`: Rimuove un contatto dalla lista dei preferiti e genera un'eccezione se il contatto non è contrassegnato come preferito.

RubricaPreferiti

La classe `RubricaPreferiti` si occupa di gestire i contatti contrassegnati come preferiti. Questi contatti vengono trattati separatamente rispetto agli altri per facilitare un accesso rapido.

Attributi

- `elencoPreferiti (Set<Contatto>)`: Contiene l'insieme dei contatti preferiti.

Metodi

- **Costruttore**
  - `RubricaPreferiti()`: Inizializza l'attributo `elencoPreferiti` come un nuovo `TreeSet`.

- **Metodi di Gestione dei Contatti Preferiti**
  - `void addContattoPreferito(Contatto c)`: Aggiunge un contatto all'elenco dei preferiti e lancia un'eccezione se il contatto è `null`.
  - `void removeContattoPreferito(Contatto c)`: Rimuove un contatto dall'elenco dei preferiti e lancia un'eccezione se il contatto è `null`.
  - `void resetRubricaPreferiti()`: Svuota completamente l'elenco dei contatti preferiti.

- **Metodi di Accesso**

- `Set<Contatto> getElencoPreferiti()`: Restituisce l'elenco dei contatti preferiti come un `Set`.

GestoreOperazioniRubrica

La classe `GestoreOperazioniRubrica` si occupa di interagire con la rubrica utilizzando metodi come la ricerca, l'aggiunta, la rimozione dei contatti, il reset della rubrica e la gestione dei file di salvataggio e lettura. La rubrica è gestita tramite l'oggetto `rubrica`, che incapsula l'elenco dei contatti e quello dei contatti preferiti.

Attributi

- `rubrica (Rubrica)` : Oggetto che rappresenta la rubrica e contiene i contatti e i preferiti.

Metodi

- **Metodi di Gestione dei Contatti**
  - `Set<Contatto> ricercaContatti(String text)`: Ricerca contatti nella rubrica il cui nome o cognome iniziano con il testo fornito. Restituisce un insieme di contatti che corrispondono al criterio di ricerca.
  - `boolean aggiungiContatto(Contatto c)`: Aggiunge un nuovo contatto alla rubrica. Restituisce `true` se il contatto è stato aggiunto con successo, `false` altrimenti.
  - `boolean rimuoviContatto(Contatto c)`: Rimuove un contatto dalla rubrica. Restituisce `true` se il contatto è stato rimosso con successo, `false` altrimenti.

- **Metodi di Gestione della Rubrica**

- `void resetTotale()`: Resetta completamente la rubrica, eliminando tutti i contatti e i preferiti.

- **Metodi di Gestione dei File**
  - `void salvaRubrica(String nomefile)`: Salva lo stato corrente della rubrica in un file. Accetta il nome del file come parametro e lancia un'eccezione `IOException` in caso di errore durante il salvataggio.
  - `static void leggiRubrica(String nomefile)`: Metodo statico per importare dei contatti da un file. Lancia un'eccezione `IOException` in caso di errore durante la lettura del file.

`MainViewController`

La classe `MainViewController` funge da controller principale. Gestisce il layout principale composto da un pannello sinistro ed un pannello destro, caricando dinamicamente i contenuti della parte destra sulla base delle azioni dell'utente.

Attributi

- `leftPane` (`AnchorPane`): È il pannello sinistro dell'interfaccia utente.
- `rightPane` (`StackPane`): È il pannello destro dinamico dell'interfaccia utente.

Metodi

- **Metodi di Inizializzazione**
  - `initialize(URL url, ResourceBundle rb)` : Carica il contenuto del `leftPane` da un file FXML chiamato `leftView`. Collega il controller

della vista sinistra all'istanza del controller principale, abilitando la comunicazione tra i due.

- **Metodi di Gestione degli Eventi**
  - `loadView(String fxmlFileName)`: Carica dinamicamente le diverse view nel pannello rightPane.
  - `loadView1()`: Carica la vista `rightView1` all'interno di `rightPane`. Invoca internamente `loadView`.
  - `loadView2()`: Carica la vista `rightView2` all'interno di `rightPane`. Invoca internamente `loadView`.

`LeftViewController`

La classe `LeftViewController` è il controller per il pannello sinistro dell'applicazione. Questa classe gestisce la logica relativa agli elementi dell'interfaccia utente situati nel pannello sinistro, che include un menu, un campo di ricerca, un toggle per i preferiti, e una tabella dei contatti.

Attributi

- **Elementi FXML**
  - `addButton` (`MenuItem`): Pulsante per aggiungere un nuovo contatto.
  - `importButton` (`MenuItem`): Pulsante per importare una rubrica.
  - `exportButton` (`MenuItem`): Pulsante per esportare una rubrica.
  - `searchField` (`TextField`): Campo di testo per effettuare ricerche tra i contatti.
  - `prefToggle` (`ToggleButton`): Pulsante a scorrimento per mostrare/nascondere i contatti preferiti.
  - `contattiTable` (`TableView<Contatto>`): Tabella che visualizza i contatti.
  - `nomeColumn` e `cognomeColumn` (`TableColumn<Contatto, String>`): Colonne della tabella per il nome e il cognome.

- **Dati e Riferimenti**
  - `contatti` (`ObservableSet<Contatto>`): Contenitore dei contatti gestiti dalla tabella. È un set osservabile per aggiornare dinamicamente la vista.
  - `mainViewController` (`MainViewController`): Riferimento al controller principale per la comunicazione tra il pannello sinistro e il resto dell'applicazione.

Metodi

- **Metodi di Inizializzazione**
  - `initialize(URL url, ResourceBundle rb)`: Metodo chiamato automaticamente durante l'inizializzazione.
  - `setMainViewController(MainViewController mainViewController)`: Imposta il riferimento al controller principale per abilitare la comunicazione e la gestione centralizzata della logica applicativa.

- **Metodi di Gestione degli Eventi**
  - `aggiungiContatto(ActionEvent event)`: Viene invocato quando l'utente preme il pulsante nella barra di menù "`Aggiungi contatto`". Chiama il metodo `loadView2()` del controller principale permettendo all'utente di inserire dati per la creazione di un nuovo contatto.
  - `importaRubrica(ActionEvent event)`: Viene invocato quando l'utente preme il pulsante nella barra di menù "`Importa`". Importa in rubrica i contatti presenti in un file scelto dall'utente.
  - `esportaRubrica(ActionEvent event)`: Viene invocato quando l'utente preme il pulsante nella barra di menù "`Esporta`". Esporta i contatti della rubrica in un file scelto dall'utente.
  - `ricercaContatto(KeyEvent event)`: implementa la ricerca dei contatti basata sull'input dell'utente nel campo di ricerca.
  - `mostraPreferiti(ActionEvent event)`: Viene invocata quando l'utente preme il ToggleButton dei preferiti. Attiva o disattiva la visualizzazione dei contatti preferiti.

RightView1Controller

La classe `RightView1Controller` è il controller per il pannello destro dell'applicazione. Questa vista gestisce la visualizzazione e l'interazione con i dettagli di un contatto selezionato, includendo funzioni per modificarlo, eliminarlo e contrassegnarlo come preferito.

Attributi

- **Elementi FXML**
  - `modifyButton` (`MenuItem`): Pulsante per modificare il contatto attualmente visualizzato.
  - `deleteButton` (`MenuItem`): Pulsante per eliminare il contatto attualmente visualizzato.
  - `nomeLabel`, `cognomeLabel` (`Label`): Etichette che mostrano il nome e il cognome del contatto.
  - `numero1Label`, `numero2Label`, `numero3Label` (`Label`): Etichette per i numeri di telefono del contatto.
  - `email1Label`, `email2Label`, `email3Label` (`Label`): Etichette per gli indirizzi email del contatto.
  - `preferitoCheck` (`CheckBox`): Checkbox per indicare se il contatto è contrassegnato come preferito.

- **Dati e Riferimenti**
  - `mainViewController` (`MainViewController`): Riferimento al controller principale per abilitare la comunicazione tra le diverse viste.

Metodi

- **Metodi di Inizializzazione**

- `initialize(URL url, ResourceBundle rb)`:Metodo chiamato automaticamente durante l'inizializzazione della vista.
- `setMainViewController(MainViewController mainViewController)`: Imposta il riferimento al controller principale per consentire la gestione centralizzata delle azioni e delle transizioni di vista.

- **Metodi di Gestione degli Eventi**
  - `modificaContatto(ActionEvent event)`:Viene invocato quando l'utente preme il pulsante nella barra di menù "`Modifica`". Chiama il metodo `loadView2()` del controller principale permettendo all'utente di inserire dati per la modifica del contatto.
  - `eliminaContatto(ActionEvent event)`:Viene invocato quando l'utente preme il pulsante nella barra di menù "`Elimina`". Rimuove dalla rubrica il contatto selezionato.
  - `switchPreferito(ActionEvent event)`:permette contrassegnare o rimuovere il contatto dai preferiti, in base allo stato di una `CheckBox`.

`RightView2Controller`

La classe `RightView2Controller` è il controller per il pannello destro dell'applicazione. Questa vista gestisce l'inserimento o la modifica di un contatto, fornendo campi di input per i dettagli del contatto e opzioni per confermare o annullare le modifiche.

Attributi

- **Elementi FXML**
  - `stopButton` (`MenuItem`): Pulsante per annullare l'operazione corrente.
  - `confirmButton` (`MenuItem`): Pulsante per confermare e salvare i dettagli del contatto.

- `nomeField`, `cognomeField`: Campi per l'inserimento del nome e del cognome del contatto.
- `numero1Field`, `numero2Field`, `numero3Field`: Campi per l'inserimento di uno, due o tre numeri di telefono.
- `email1Field`, `email2Field`, `email3Field`: Campi per l'inserimento di uno, due o tre indirizzi email.
- `preferitoCheck` (`CheckBox`): Casella di controllo per contrassegnare il contatto come preferito.

- **Dati e Riferimenti**
  - `mainViewController` (`MainViewController`): Riferimento al controller principale per abilitare la comunicazione e la gestione delle transizioni tra le diverse viste.

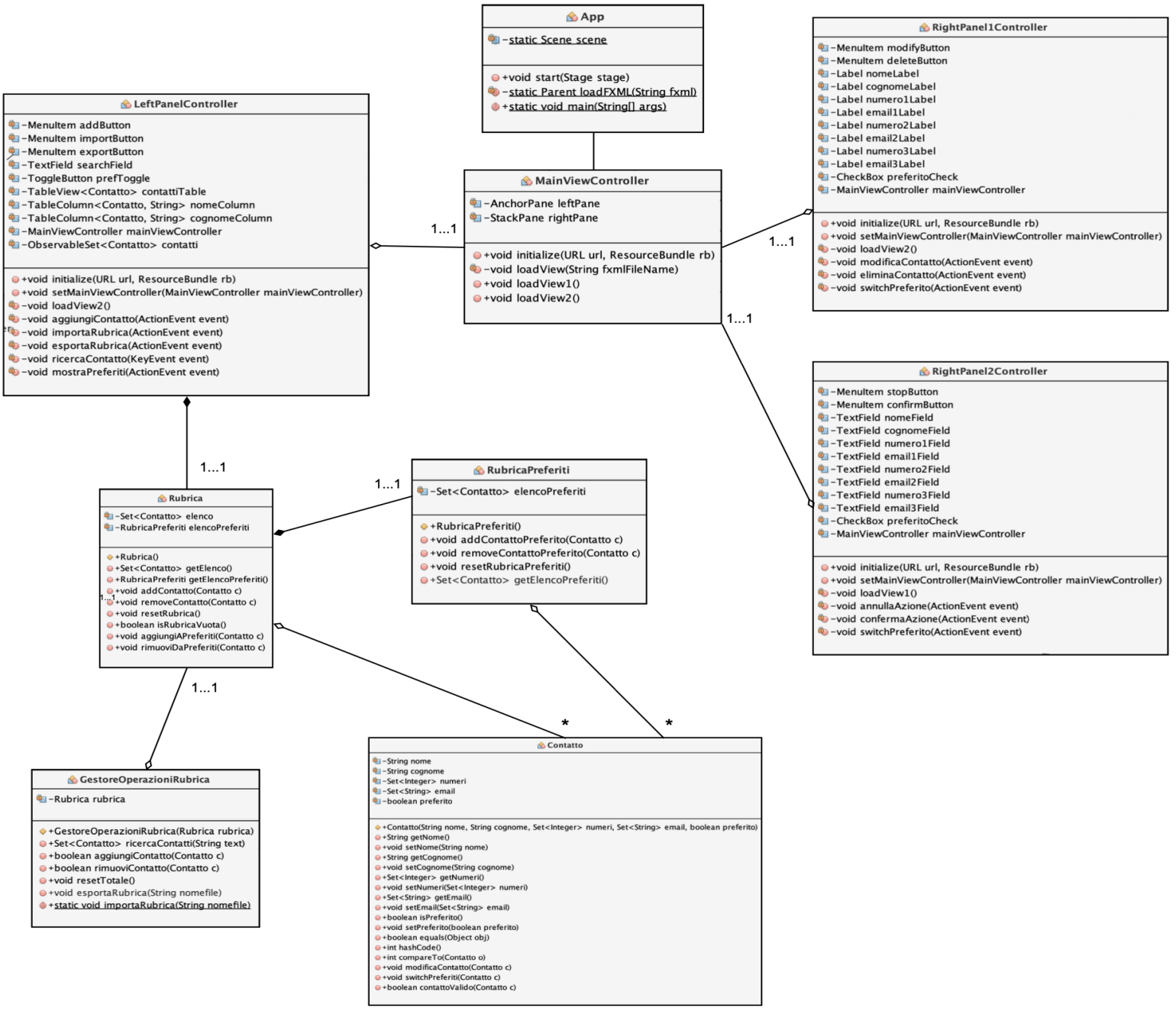
Metodi

- **Metodi di Inizializzazione**
  - `initialize(URL url, ResourceBundle rb)`: Metodo chiamato automaticamente durante l'inizializzazione della vista.
  - `setMainViewController(MainViewController mainViewController)`: Imposta il riferimento al controller principale per consentire la gestione centralizzata delle azioni e delle transizioni di vista.

- **Metodi di Gestione degli Eventi**
  - `annullaAzione(ActionEvent event)`: Invocato quando l'utente preme il pulsante "Annulla". Chiama il metodo `loadView1()` del controller principale per tornare alla visualizzazione precedente senza salvare le modifiche o l'aggiunta di un contatto.
  - `confermaAzione(ActionEvent event)`: Invocato quando l'utente preme il pulsante "Conferma". Salva le modifiche al contatto corrente o l'aggiunta di un contatto e ritorna alla visualizzazione principale chiamando il metodo `loadView1()` del controller principale.

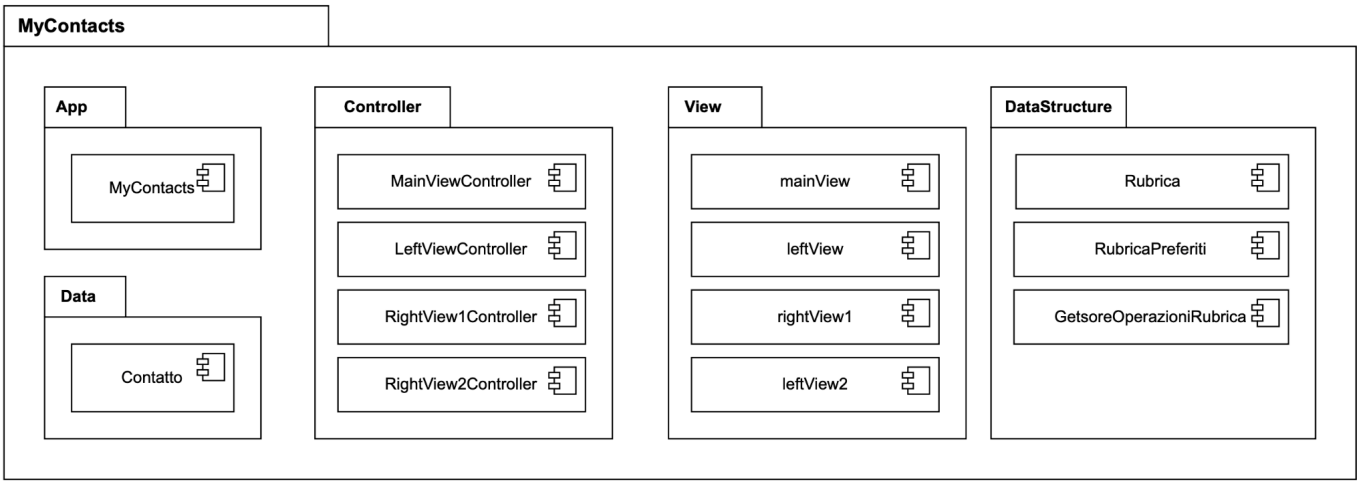
- `switchPreferito(ActionEvent event)`: Permette di contrassegnare o rimuovere il contatto come preferito, in base allo stato della checkbox.

Diagrammi delle Classi



Package Diagram

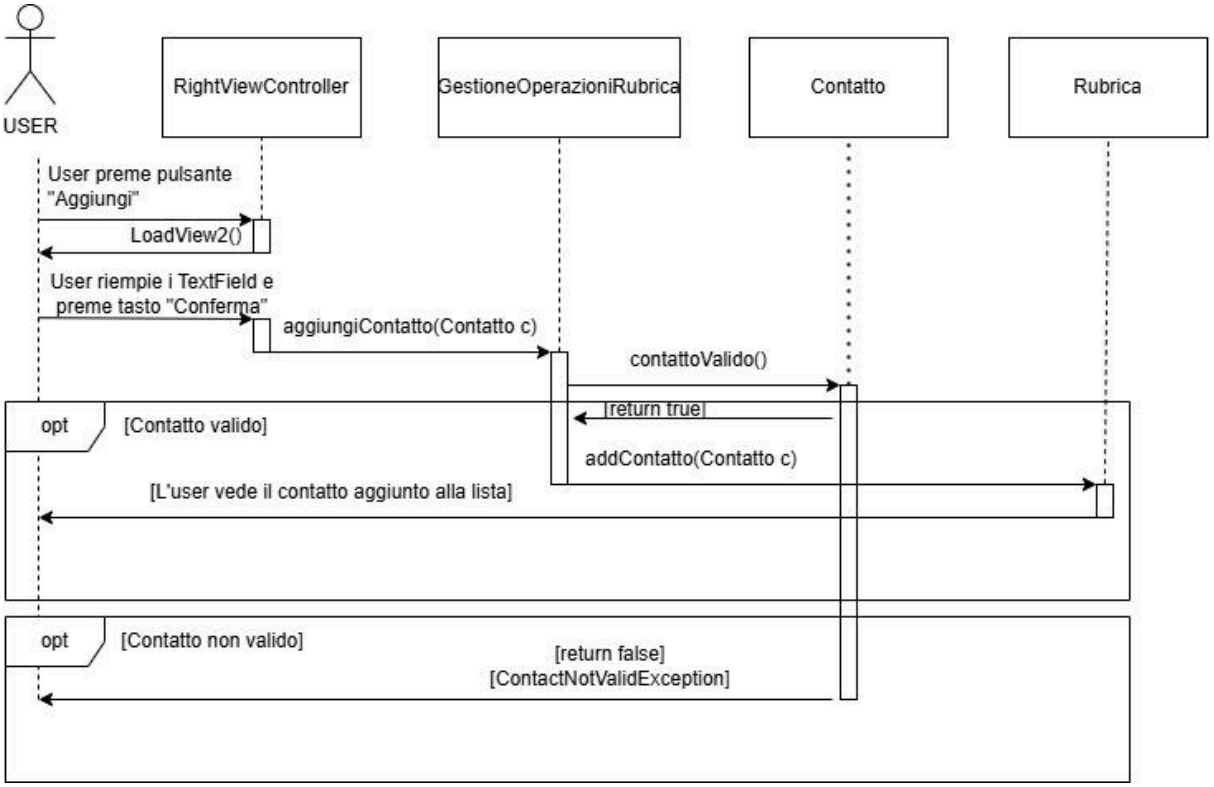
Il seguente Package Diagram descrive come abbiamo diviso a livello logico e strutturale i Package del nostro progetto, questo anche per avere un ulteriore organizzazione all'interno di esso.



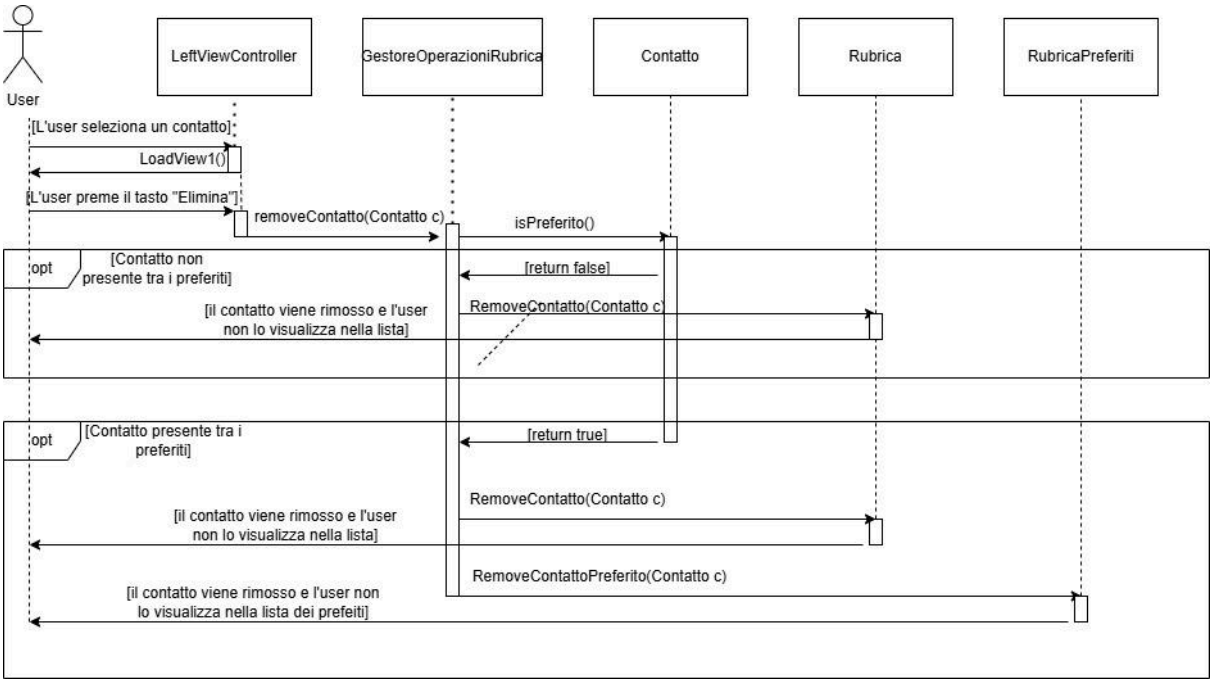
Il diagramma rappresenta l'architettura del sistema organizzata in pacchetti secondo una chiara separazione tra i dati (data, datastructure), la logica applicativa (controller) e la visualizzazione (view).

Sequence Diagrams

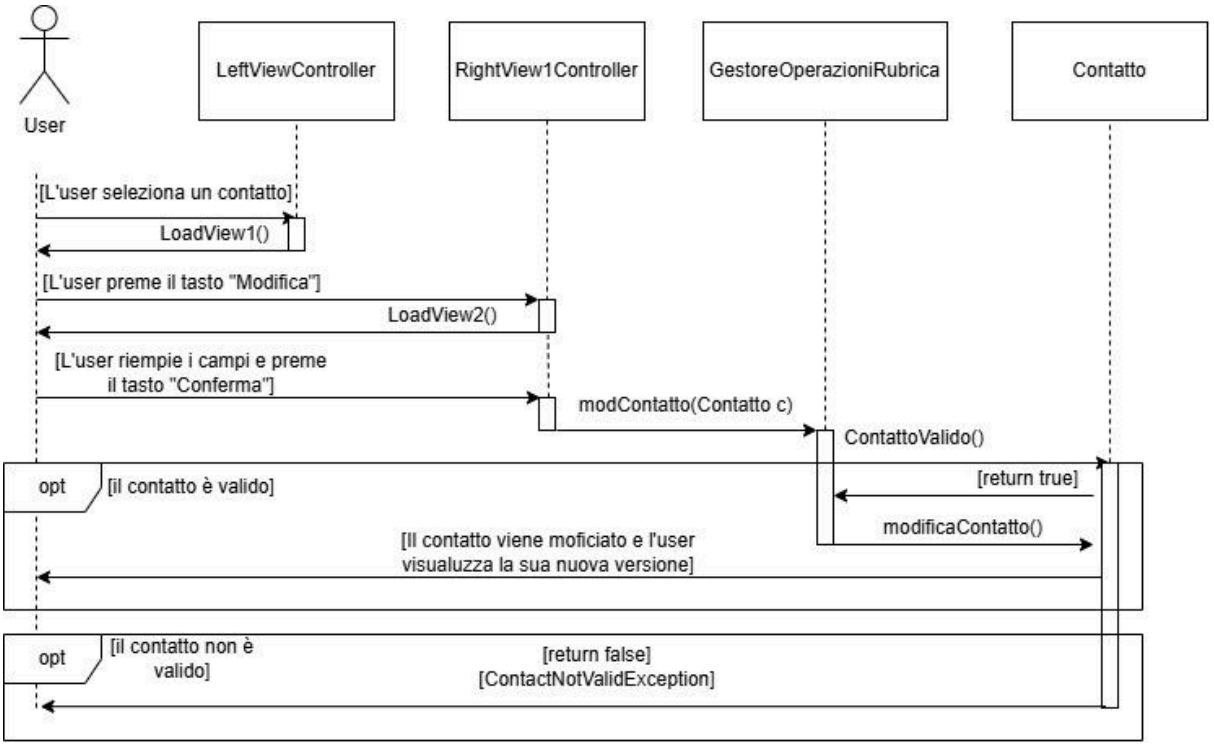
AGGIUNGI CONTATTO



RIMUOVI CONTATTO



MODIFICA CONTATTO



Changelog

Abbiamo apportato modifiche alla struttura della GUI per migliorare l'organizzazione e la chiarezza. È stata aggiunta una nuova scena, che verrà visualizzata quando si preme il pulsante " nome del bottone da premere ". Questa scena mostrerà nel dettaglio le informazioni relative a un singolo contatto.

L'obiettivo della modifica è visualizzare l'interfaccia dedicata alle informazioni di un contatto (già implementata nella versione precedente) solo quando necessario, mantenendo l'interfaccia principale più ordinata e minimale.

La nuova scena include i seguenti elementi:

- **Label "Nome"**: visualizza il nome del contatto.
- **Label "Cognome"**: visualizza il cognome del contatto.
- **Tre label "Numero telefonico"+<numero_del_contatto[1-3]>**: mostrano fino a tre numeri di telefono associati al contatto.
- **Tre label "E-mail" + <email_del_contatto[1-3]>** : mostrano fino a tre indirizzi e-mail associati al contatto.
- La sezione sinistra dell'interfaccia grafica presenta una **MenuBar** per effettuare operazioni su uno specifico contatto, essa è composta da:
    - un tasto **"Modifica"** per modificare i dati del **contatto**.
    - un tasto **"Preferiti"** per aggiungere il contatto alla lista dei preferiti.

Partecipanti

NOME	COGNOME	MATRICOLA
Gabriele	Lupo	0612707641
Michele	Naddeo	0612708622
Andrea	Sangineto	0612707424
Gianmarco	Vitolo	0612708376