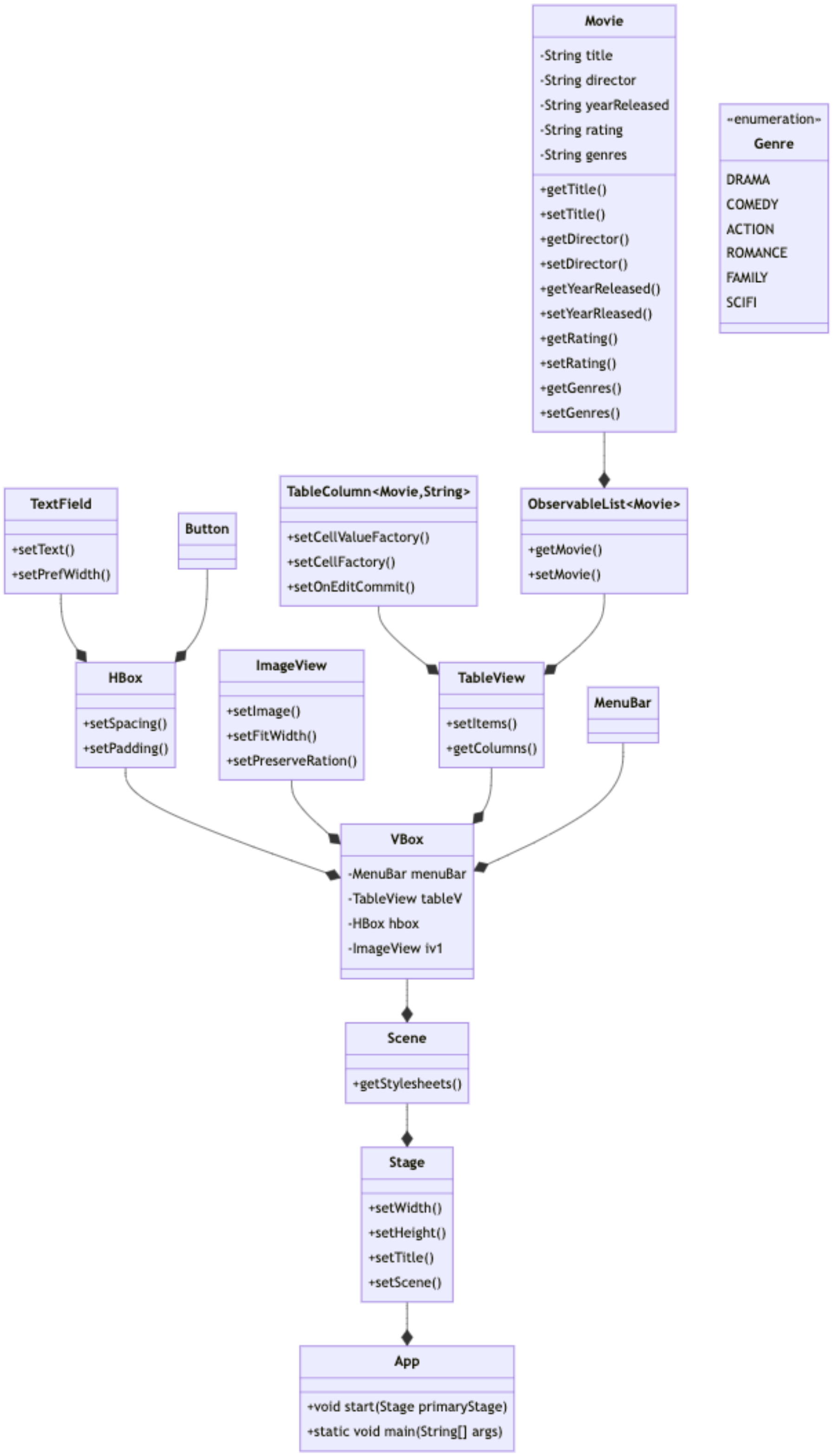
MovieDB Implementation Manual

UML Diagram



MOVIE

The Movie class is the base class of this app. It has attributes as below:

-String title
-String director
-String yearReleased
-String rating
-String genres

In order to listen to actions, I have formatted my String to a SimpleStringProperty, This class provides a full implementation of a Property wrapping a String value.

JavaFX's properties hold actual values and provide change support, invalidation support, and binding capabilities. So it allows us to listen to state changes from JavaFX UI controls.

```
public class Movie implements Serializable {
    private final SimpleStringProperty title;
    private final SimpleStringProperty director;
    private final SimpleStringProperty yearReleased;
    private final SimpleStringProperty rating;
    private final SimpleStringProperty genres;
    //constructor
    Movie(String title, String director, String yearReleased,
String rating, String genres) {
        this.title = new SimpleStringProperty(title);
        this.director = new SimpleStringProperty(director);
        this.yearReleased = new
SimpleStringProperty(yearReleased);
        this.rating = new SimpleStringProperty(rating);
        this.genres = new SimpleStringProperty(genres);
    }
}
```

ObservableList<Movie>

I use an ObservableList<Movie> to maintain the whole movie list. It is a list that allows listeners to track changes when they occur.

```
private final ObservableList<Movie> data = 
FXCollections.observableArrayList(
     new Movie("The Matrix", "Lana Wachowski, Lilly 
Wachowski", "1999", "5", "Sci-Fi"),
     new Movie("Crouching Tiger, Hidden Dragon", "Ang Lee", 
"2000", "5", “Action”),
     …
)
```

TableView

The table view is populated with data from an ObservableList<Movie> collection.

+setItems()
+getColumns()
+setEditable()

A TableView is made up of a number of TableColumn instances. Each TableColumn in a table is responsible for displaying (and editing) the contents of that column. As well as being responsible for displaying and editing data for a single column.

For example, if the value of a movie's rating property is set to a new value, that update will be reflected in TableView because it listens for the property change.

```
tableView.setItems(ObservableList<Movie>)

tableV.getColumns().addAll(titleCol, directorCol, 
yearreleasedCol, ratingCol, genresCol);
```

TableColumn<Movie, String>

+setCellValueFactory()-populate individual cells in the column
+setCellFactory()
+setOnEditCommit()

```
titleCol.setCellValueFactory(new PropertyValueFactory<Movie, 
String>("Title"));
titleCol.setOnEditCommit(
   new EventHandler<TableColumn.CellEditEvent<Movie, 
String>>() {
       @Override
       public void handle(TableColumn.CellEditEvent<Movie, 
String> t) {((Movie) t.getTableView().getItems().get(
t.getTablePosition().getRow())).setTitle(t.getNewValue());
       }
   }
);
```

Button

An eventHandler is attached to the add Button
I bind an event handler to the Add button. Here I create a new Movie, and add it to the Observable list.

```
btnAdd.setOnAction(new EventHandler<ActionEvent>() {
   @Override
   public void handle(ActionEvent e) {
       data.add(new Movie(
            addTitle.getText(),
            addDirector.getText(),
            addYearReleased.getText(),
            addRating.getText(),
            addGenres.getText()));
```

Other Javafx classes

App

-Stage primaryStage
-Scene scene
-VBox vb
+void start(Stage primaryStage)
+static void main(String[] args)

Stage

The Stage class is used to construct the main window of the app. Contains one scene.

+setWidth()
+setHeight()
+setTitle()
+setScene()

Scene

Within the scene, the VBox component contains the other layout components including a menu bar, a label, a table with six columns, and a horizontal row at the bottom.

+getStylesheets(), you add the CSS file here.

HBox and VBox

The HBox and VBox layout their child controls in a single horizontal or vertical row respectively.

ImageView

+setImage()
+setFitWidth()
+setPreserveRation()

TextField

Text input component that allows a user to enter a single line of unformatted text.

+setText()
+setPrefWidth()