OSTBAYERISCHE
TECHNISCHE HOCHSCHULE
REGENSBURG

Name:	Christoph Prenissl
Email:	christoph.prenissl@st.oth-regensburg.de
Student ID:	3174997

June 10, 2021

IMDB Software to fetch data of Hollywood Actors and Actresses

Task 4 - Project Report

Contents

1 Project Description

This project mainly consists of creating a Python client to fetch data of the *IMDB Top 50 Actors and Actresseslist* and also gather their movie data. The client uses an API to fetch all the basic actors/actresses list (4.1), the actor/actress About section (4.2) and all their movies (4.3). For Sections 4.4 - 4.7 Web-scraping is used to get awards data and ratings of the movies. The client is presented in a window based UserInterface where the user can click to gather the wanted information.

The specifics are handled in this document.

2 Tools, Modules and Data-Structure

2.1 Presentation Tools

For presenting the project I also used VS Code. I wrote the reports in $\LaTeX$ with the help of the *LaTex Workshop* extension and *PlantUml* to present core structures and flows of the client. In the presentation of the client I used *Powerpoint*.

2.2 Development Tools

For development of the client I used *Python 3.9.4* in *Visual Studio Code* with the *Python IntelliSense* extension which helped me in code completion and understanding the structure of all the frameworks and modules. For version control I used *Git* and the helpful VS Code extension *GitLens* which helped me to keep track of my changes in the project files.

The UI was designed with *Qt Designer*. It was very convenient to have a graphical UserInterface to drag and drop widgets and have an overal understanding of all elements.

2.3 Modules

When it comes to the modules, *Requests* and *BeautifulSoup 4* were necessary to handle all the web scraping. The http context is created with the module *SSL*. For most data I used a data frame created with *Pandas*.

The Graphical UserInterface was implemented using *PyQt6*. The framework also provides Thread libraries to help with multi-threading.

3 Design

The client has one base module at the root of the project and the children modules *landing* and *detail.*

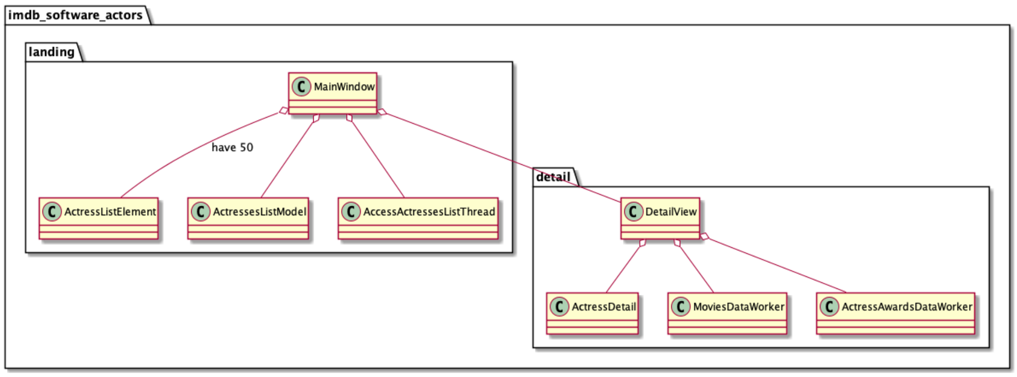


Figure 1: All packages used in the project with the most important classes.

While the base module only contains the entry point in the *main.py* file,

the *landing* module handles the asynchronous fetching of the actresses list (4.1), visualisation and interaction with the list and initializes an DetailView from *detail* when a list element gets accessed. *MainWindow* is a QObject class which handles the UI update and communicates with *AccessActressesListThread* for data provision. For the ListView containing the actresses/actors in MainWindow *ActressesListModel* is used for correctly displaying the actor/actress data. *ActressListElement* is a data wrapper for all the data to present in the list.

The *detail* module helds logic for fetching deeper information with multi-threading on an actor or actress regarding ratings and awards. It also contains the code for UI displaying and updates. The *DetailView* class acts analogously to MainView as an controller for handling UI updates and triggering events on its threads. These threads manage the workers *MoviesDataWorker* and *ActressAwardDataWorker* for retrieving the needed data. *ActressDetail* functions as a wrapper for a data instance.

4 Functionalities

In this section all the necessary specifications for the project will be further discussed.

4.1 List of all available actors and actresses

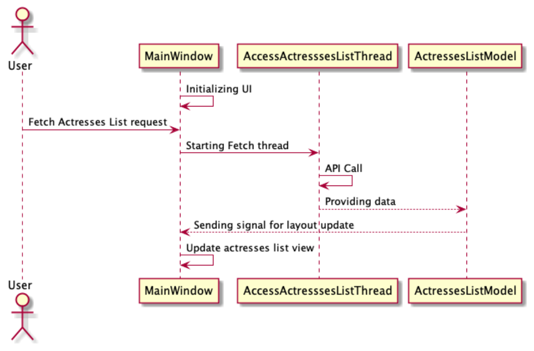

User
MainWindow
AccessActresssesListThread
ActressesListModel
Initializing UI
Fetch Actresses List request
Starting Fetch thread
API Call
Providing data
Sending signal for layout update
Update actresses list view
User
MainWindow
AccessActresssesListThread
ActressesListModel

Figure 2: Main flow of fetching actresses list

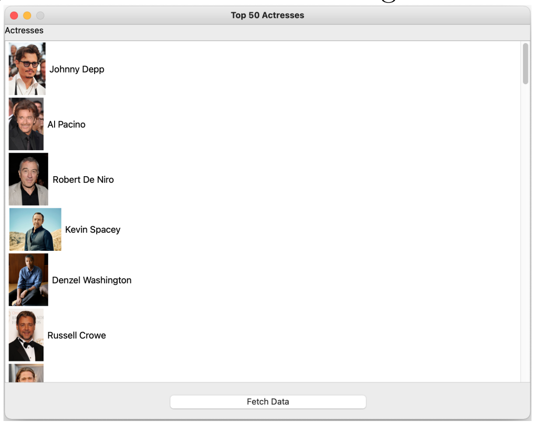


Figure 3: screenshot of the MainView

When the user clicks on the *Fetch Button* the sequence 2 starts and everytime a new actor/actress is published in actresses list the the actresses table view gets updated so the user sees the update process. Here the *IMDB-Api* is used. The extracted JSON file gets translated into a dataframe. Since the fetching is executed in a seperate thread, the main thread is not blocked and the UI is responsive. At any time the user can click on the list element to transition to DetailView of the actress.

Main methods:

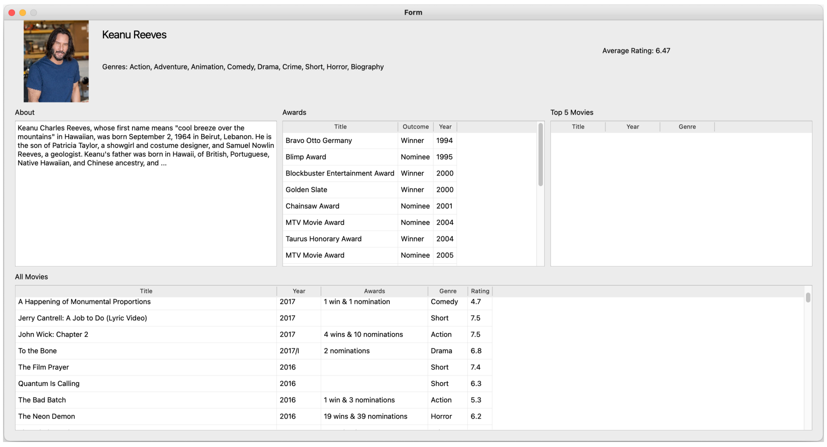
- *fetchActressesDataFromUrl(self)*
- *fetchActress(self)*

4.2 About the actor/actresses

With a new API call to the actress list of the clicked actress/actor in the MainView a new dataframe with Pandas gets created. A short summary for each actor/actress is provided and therfore used in a textfield. You can see the result in the upper left area of fig 4.

Main methods:

- initContent(self, pixmap, name, id)



Keanu Reeves

Genres: Action, Adventure, Animation, Comedy, Drama, Crime, Short, Horror, Biography

Average Rating: 6.47

About

Keanu Charles Reeves, whose first name means "cool breeze over the mountains" in Hawaiian, was born September 2, 1964 in Beirut, Lebanon. He is the son of Patricia Taylor, a showgirl and costume designer, and Samuel Nowlin Reeves, a geologist. Keanu's father was born in Hawaii, of British, Portuguese, Native Hawaiian, and Chinese ancestry, and ...

Awards

Title	Outcome	Year
Bravo Otto Germany	Winner	1994
Blimp Award	Nominee	1995
Blockbuster Entertainment Award	Winner	2000
Golden Slate	Winner	2000
Chainsaw Award	Nominee	2001
MTV Movie Award	Nominee	2004
Taurus Honorary Award	Winner	2004
MTV Movie Award	Nominee	2005

Top 5 Movies

Title	Year	Genre

All Movies

Title	Year	Awards	Genre	Rating
A Happening of Monumental Proportions	2017	1 win & 1 nomination	Comedy	4.7
Jerry Cantrell: A Job to Do (Lyric Video)	2017		Short	7.5
John Wick: Chapter 2	2017	4 wins & 10 nominations	Action	7.5
To the Bone	2017/I	2 nominations	Drama	6.8
The Film Prayer	2016		Short	7.4
Quantum Is Calling	2016		Short	6.3
The Bad Batch	2016	1 win & 3 nominations	Action	5.3
The Neon Demon	2016	19 wins & 39 nominations	Horror	6.2

Figure 4: screenshot of the DetailView

4.3 All time movie names and years

The API call of 4.2 is also used to gather all movies and their release years of the provided dataframe. To fill the movies table in the bottom of fig 4 the dataframe gets iterated using the iterrows property of the dataframe. To display every item correctly, the columns get updated for every insert.

Main methods:

- initContent(self, pixmap, name, id)

4.4 Awards to actor/actresses in different years

4.5 Movie genre of actor/actresses

4.6 Average rating of their movies

4.7 Top 5 movies, their respective years and genre