Film viewer project report

Name: Film viewer Author: Daniil Kovalenok Group number: 224-2 Submission date: June 10, 2023

Problem statement

There was a need for a GUI application to manage in some way a list of films. Functionality desired:

- Filtering by name
- Filtering by property
- Adding, Removing, Editing an entry
- Saving, Loading, Creating lists

Implementation details

The app is a generic Qt MVC program with bits of procedural code sprinkled in because I couldn't be bothered to create methods every time I needed something.

The core of film viewer consists of two models:

- `FilmListModel`, which inherits from `QAbstractListModel`;
- `FilmFilterProxyModel`, which inherits from `QSortFilterProxyModel`. The former is responsible for the management of film objects in-memory, their deletion, addition and editing. The latter proxies the first model and controls which films are displayed to the user.

Qt provided most of functionality needed for this task; however, property filters had to be implemented from scratch, thus justifying the usage of a superclass in this case.

Addition and editing of an entry both use the same Modal widget.

The rest of the functionality contains no notable details.

Results and discussion

This project was my first experience with Qt. Naturally, it was iterated on quite a lot. My first approach utilized a table which would provide similar user experience to a sheet engine (Excel, Google sheets, etc.) It, however, was was quickly scrapped due to the complexity of this approach and issues with signals I had been facing as a new Qt developer.

The next design ended up being the final one. Discovering `QStringList` inspired me to go with it. The challenge I needed to overcome is the sorting and filtering of data, which both worked well until property filters were introduced. Afterwards, I discovered that overriding model methods require the inheriting class to call special functions and emit special signals. This was a big roadblock, which I solved by separating filtering and the management of data into two separate models (`FilmListModel` and `FilmFilterProxyModel`). Afterwards, I faced no extra issues.

Conclusion

It's neat. Here's a list of obvious extra features that would enhance user experience

- Batch deletion
- Merging csv files
- More than one filter in query per property
- Selectable text in the panel on the right
- Edit, delete hotkey
- Configuration file to allow hardcoded values in the add dialog to be added/removed

Thank you.

rev1.