Movies Tracker GUI Application

Oganesyan Tigran, 214 – 2

12.06.2022

Supervisor: Carrasquel Julio

Problem statement

My task was to create a GUI application using C++ together with Qt library and Qt creator. I had to implement range of different features related to operations with csv file and make them via user-friendly GUI.

Problem specification

Description In this text, “movie” will denote both movies and serials. In the app there should be the list with the names of movies with different statuses. The necessary fields, which must be considered:
1. Title a text field The title of the movie itself.
2. Type a text field It shows if this is a movie or a tv show.
3. Director a text field The name of the movie director.
4. Country a text field The country, where the movie was produced.

5. Year a number field The year, when the movie was released.
6. Rating of user float data type The score should be a number from 1 to 10.
7. Status a text field It’s a text, which can have only of of these 4 texts: “Want to watch”, “Watching”, “Stopped”, “Watched”.

Basic features

1. Adding movies to the list, deleting them.
2. Editing the status of the movie.
3. Showing the total number of movies.
4. Backup of the list. Saving it to a file, loading from a file.

Additional features

1. Master-detail. After clicking on the name of the file it should show a page with detailed info about fields.

2. Filtering. There is a field where a user inputs a string or a number. The table should only show the movies that have this string / number in one of the fields.

3. Saving current filter to a file as user settings. Loading settings.

4. Sorting the movie list by any field.

Implementation details

First of all, the data is shown in the app via QtableView and QStandardItemModel and all the implementation are linked with that model/view.

Basis feature 1:

To add the movies to the list I created a new window “addWindow” that has several fields where user can input the data and add it to the list by clicking “add” button. Then pointer to the model is taken which allows to link the inputted data with the QStandardItemModel which automatically update QTableView.

In order to delete instance from the view, a button was created. It takes the index of the instance and removes it from itemModel and from the vector with data.

Basic feature 2:

To edit the status of the movie “editWindow”is called. There user finds textEdit and “Edit” button which allows to change the status. Then pointer to the model and index of the element are taken. After that pointers are set in the window where data is shown and the model is updated.

Basic feature 3:

To show the total number of movies was created an ineditable textEdit where the number of elements in data vector is set. Every time the number of elements is changed (for example, when user adds/deletes elements or uploads a new file) that number is updated.

Basic feature 4:

To save the file I created a function that creates a file, asks the name of that file, and then writes data into it via stringstream line-by-line.

To load the file, there exists another function that is pretty similar to the function that initially loads given data (“netflix-titles.csv). It asks to user which file to upload and then goes through the file line-by-line delimiting the elements and appending to the instance of data object. Finally that object are appended to the data vector which is set to the model and view is updated.

Additional feature 1:

In order to show all the info about the element in table a new window is executed. It contains several ineditable textEdit’s that contain data from element. There also exists a function that takes data object as argument and sets all the data to textEdit’s.

Additional feature 2:

To use search through the table, there was created a custom model “proxyModel” that has 2 main functions. First one is “lessThan” function which is a simple comparator and second one is “filterAcceptRow” function that is a filter itself. It takes indices from source model and returns values that are set by predefined conditions.

Results and conclusion

Finally, I created Qt-based GUI application, that provides user with a variety of options on manipulating data and gives opportunity to have all the interesting movies in an app. Still, there are several things that should be added and improved. Firstly, adding 2 “master” features: sorting by order and saving user settings. Second thing is, making some unit tests to ensure that app works properly under different conditions and make a little code refactoring in order to delete some unused fragments of code and unused headers to make code more readable and presentable.