A Qt5-based Music Player

Allen Yin asy13
Jieqing Dai jd260

For our project, we will develop a functionally lightweight music player with a GUI based on the Qt5 framework. This music player will have all the basic functionalties of a media player, in addition to playlist support.

1 Functional Requirements

Playback We will use either Qt5's built-in MediaPlayer widget or a different media playback library to handle the decoding of the media files and the system calls to the playback devices. The type of media files will be limited to mp3 files in the basic implementation. The essential functionalities, which also translate to individual GUI elements, include:

- Play/Pause button to start or pause the playback of the current media file.
- A slider that shows the progress of the playback, which can be manually dragged to advance/rewind the song. When the music player is in focus, the left/right arrowkeys can be used instead of manually dragging the slider to advance/rewind the song by 3 seconds.
- Next/Back buttons to skip to the next or previous song in the current playlist. If the current playlist has only 1 song, nothing happens. If it's currently playing the first song of the playlist, *Back* plays the last song of the playlist; if the last song of the playlist is playing, *Next* plays the first song.

Playlist The *Playlist* is a collection of media files that we are currently playing, and is shown via the *Playlist* area. The user can add songs to the playlist via dragging selection from the *Library/File Browser* area and dropping into the *Playlist* area. The *Playlist* area will have columns to display the song's Name, Artist, and Album information, based on the mp3 file's metadata. These information can be modified within this area. A search box will be available for querying the playlist. There will also be checkboxes/buttons for different playlist options including shuffle, repeat, and loop. Playlists can be saved for future use.

Library/File Browser The *Library* area allows the user to navigate the filesystem to add folders/files to his collection. Once a folder/file is selected, the mp3 files contained will be processed for the metadata and added to the area. The Library will display the songs in artist items, which when expanded will show the individual song names. A search box will be available to query through the collection. The data for the Library will be stored via Sqlite, for which Qt5 provides support. The *Library* will also contain a list of known/saved playlists.

Title Area The *Title Area* shows the Name, Artist, and Album of the current song being played, as well as the volumn control.

GUI The GUI will be implemented in Qt5 for its cross-platform support. Ubuntu 14.04 is our default platform. A mockup of the application is shown below.

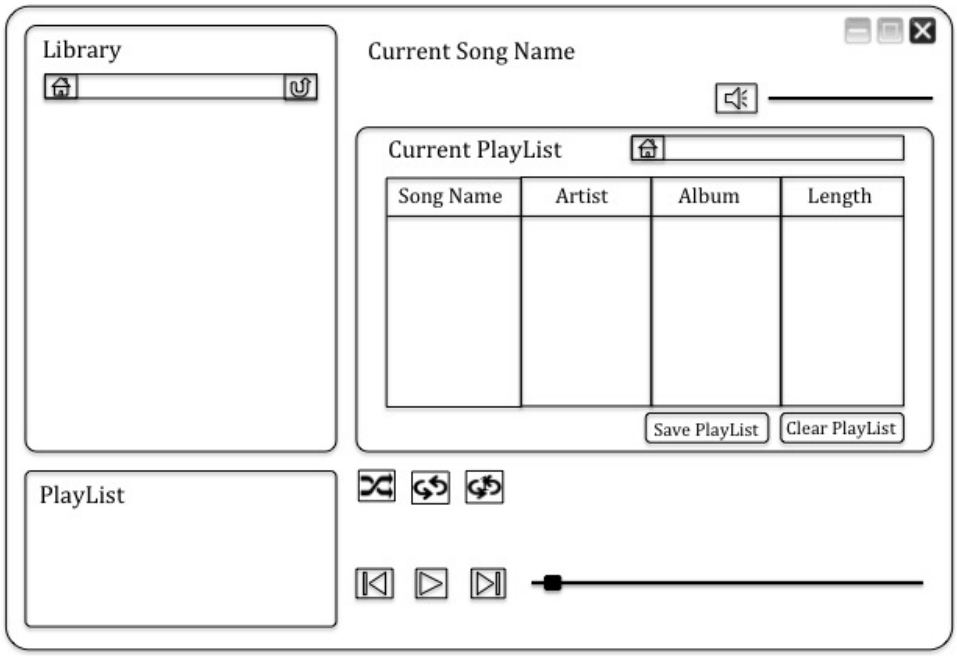

Figure 1: Mock-up of our Music Player's GUI

2 Stretch Goals

Platform Support Our first stretch goal will be to provide support of the application on both Linux and MacOSX, and maybe Windows (as the file structure and playback devices organization is different from Unix systems).

Media Support Our second stretch goal will be to provide support to media formats other than mp3, such as m4a, wav, wma.

Lyrics Fetching Our third stretch goal will be to provide for lyrics fetching. The lyrics for the song current playing will be fetched through select sites from the internet and the *Playlist* area will be resized dynamically to display lyrics.

Task	Expected Person-Hours			Start	End	Person
	Implement	Test	Debug			
Media Playback	6	2	1	Nov 3	Nov 6	Allen, Jieqing
Library	6	2	1	Nov 6	Nov 12	Allen
Playlist	6	2	1	Nov 6	Nov 12	Jieqing
Integration	10	4	3	Nov 12	Nov 17	Allen, Jieqing
Stretch Goals						
MacOS Support	4	1	3	Nov 17	Nov 20	Jieqing
Media Support	3	2	3	Nov 17	Nov 20	Allen
Lyrics Fetching	6	2	2	Nov 20	Nov 25	Allen, Jieqing
System Tray	4	3	3	Nov 25	Nov 27	Allen, Jieqing

Table 1: Summary of work breakdown and time expectations.

System Tray Our last stretch goal is to allow the application to sit in the OS' application tray when the 'x' button is clicked, rather than closing and exiting. In this state, a notification box with the new song's information will pop up whenever the current song is changed.

3 Software Design Plan

Our first step will be to familiarize ourselves with Qt, and to create the core media playback section of our application (as described in the *Playback* section of our functional description and GUI mockup). This first iteratioin will allow us to load one song at a time for playback. This will be split between two people as it is the most important core functionality.

After the development of the core playback, one of us will work on the *Library* area while the other on the *Playlist* area. The Library person will need to integrate Sqlite with our application. We will then work on integrating the two functionalities together, enabling drag and drop between the two areas and saving Playlists.

Then we will finalize our GUI design - adding in the Title Area and tweaking the look and feel. Af this stage, we will focus on testing and optimizing for resource usage. Once we are satisfied with the basic application, we will work down the list of our stretch goals.

Table 1 sumarizes our expected time breakdown by task, as well as who is planning to do each task, and what our exected schedule of starting/completing the tasks is.

4 Specific Expertise

This project requires a proficient knowledge with the Qt framework, especially its support for Multimedia and Sqlite. None of us have used it, however, Qt has a strong community and we should be able to find help online.

We are familiar with both Linux and MacOSX, so platform support should not be too difficult. Since Qt is platform independent, we will also be able to find help online.