MUSIC PLAYER

Vrushank Mali and Billy Gauldin

Abstract

MUSIC PLAYER is an easy-to-use software that allows you to store and listen to audio files. MUSIC PLAYER allows the user to add and remove audio files individually, or by selecting a folder containing audio files. MUSIC PLAYER also allows the user to play audio files, pause the audio file that is currently playing, repeat a single audio file or a list of audio files, shuffle between a list of audio files, and change the volume of their audio files. MUSIC PLAYER also allows the user to create and save playlists. The user will be able to view the progress of the audio file that is currently playing, as well as view the list of songs being played if the user is listening to multiple songs. MUSIC PLAYER is aimed toward people who enjoy listening to music.

1. Introduction

MUSIC PLAYER is an easy-to-use music playing software that allows the user to add audio files by selecting them individually, or by selecting a folder containing audio files. The user can create and save playlists in order to store lists of music. MUSIC PLAYER will allows the user to play an audio file or files, pause the audio file that is currently being played, repeat a single audio file, repeat a list of audio files, shuffle the order that audio files in a list of audio files is being played, and adjust the volume of their audio files. The user will be able to view the audio file(s) playing as well as the progress of the audio file that is currently being played. MUSIC PLAYER will allow the user to add and listen to their own songs, unlike iTunes which forces you to buy songs stored on their platform. The supported types will be .mp3 and .wav. It will be easy and free to use, and will contain no ads. Our target audience is anyone who enjoys listening to music, but wants a simpler and easier to use software.

1.1. Background

We decided to make a music player because Spotify contains lots of ads and forces you to pay to listen ad-free. We also wanted a simpler software made specifically for audio files.

1.2. Impacts

MUSIC PLAYER will be easier to use, allowing the user to add their own songs instead of needing to buy a song from the iTunes store. It also allows the user to listen to music without ads interrupting them, as with Spotify and youTube.

1.3. Challenges

I believe that making the Music Player work smoothly and truly ”easy-to-use” will be difficult to do. Another challenge will be properly implementing the GUI for the project. A different challenge will be allowing the user to differentiate audio files between non-audio files. Another challenge will be adding support for multiple types of audio files. I don’t want to limit the user to only one type of file.

2. Scope

MUSIC PLAYER will be finished when it is able to add and remove audio files individually and by folder, list the song’s progress, list the audio file queue, change the audio file’s volume, and create playlists. A stretch goal could be a timer the user could set that would pause the current song and tell the user that they’ve been listening to music for too long. Another stretch goal would allow you to import playlists from iTunes.

2.1. Requirements

These are some basic and standard requirements which you can find on any typical music player.

Use Case ID	Use Case Name	Primary Actor	Complexity	Priority
1	Play an audio file	User	Med	2
2	Pause	User	Med	3
3	Add an audio file	User	Med	1
4	Remove an audio file	User	Med	4

TABLE 1. MUSIC PLAYER USE CASE TABLE

2.1.1. Functional.

- MUSIC PLAYER will allow the user to play audio files.
- The user can click on the play or pause button to play or pause the current audio file. The previous and next buttons will move change the audio file to the audio file before or after it, respectively.
- MUSIC PLAYER will allow the user to play an audio file on repeat and allow user to shuffle or repeat a list of audio files.
- The user can add or remove individual audio files and/or folders of audio files.

2.1.2. Non-Functional.

- The user will not be able to interact with audio file's cover art.
- The user will not be able to interact with the background.
- MUSIC PLAYER should load within a few seconds.

2.2. Use Cases

Listed below are the use cases for MUSIC PLAYER and how it is supposed to work. Use Case Index can be seen in Table 1.

Use Case Number: 1
Use Case Name: Play an audio file
Description: The user wants to play an audio file. The user will either double click the specific audio file or right click and select "Play". This will start playing the selected audio file in MUSIC PLAYER.

1) The user loads MUSIC PLAYER.
2) The user double clicks the audio file to play or selects the audio file and right clicks and selects "Play".
3) The program takes a second or two to load and then it starts playing the audio file.

Use Case Number: 2
Use Case Name: Pause
Description: The user wants to pause the audio file. They will click on the pause/play button. This will pause the current playing music.

1) The user is listening to an audio file and wants to pause the audio file.
2) The user left-clicks on the pause/play button.
3) This pauses the current playing audio file.

Use Case Number: 3
Use Case Name: Add an audio file
Description: The user wants to add an audio file. They will left click the "Add" option from the menu. They will browse through the files in their device and will select the audio file and add it.

1) The user wants to add an audio file.
2) The user left clicks on the "Add" button from the menu.
3) This opens another window which allows the user to browse through the files in their device.
4) They select the audio file and left click "Add".
5) This will add an audio file in MUSIC PLAYER.

Use Case Number: 4
Use Case Name: Remove an audio file
Description: The user wants to remove an existing audio file from MUSIC PLAYER. They will select the audio file and will right click on the "Remove" button. This will remove the existing audio file.

1) The user wants to remove an existing audio file from MUSIC PLAYER.
2) The user selects the existing audio file by left clicking on the audio file.
3) The user then right clicks and selects "Remove" from the options. The user left clicks on the "Remove" button.
4) This will remove the existing audio file from MUSIC PLAYER.

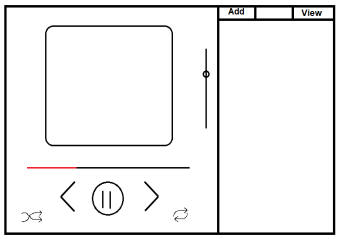

Figure 1. Use Case 1, Play

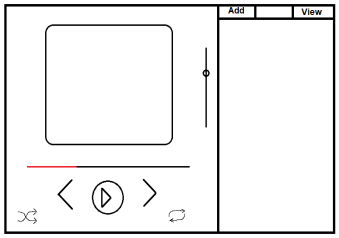

Figure 2. Use Case 2, Pause

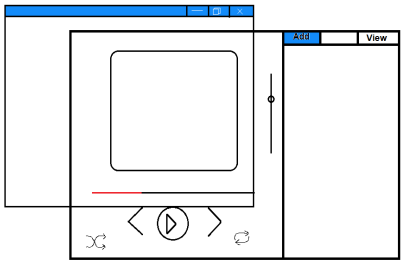

Figure 3. Use Case 3, Add an audio file

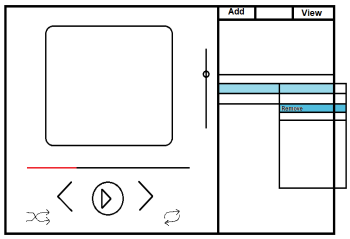

Figure 4. Use Case 4, Remove an audio file

2.3. Interface Mockups

These are basic images of what our program will look like. We don't know what else we will have in the menu other than "Add" and "View" for now. "View" will let the user select or create a playlist as well as let the user select an option which will show all the added or existing audio files stored in the music player.

3. Project Timeline

- Date 09/13/21, We submitted our group project idea, a brief introduction, a background, its social and global impacts, challenges, and the project's scope
- Date 09/27/21, In this update, we rewrote our proposal to make it look better. We also worked on interface mock-ups, use cases, and functional and non-functional requirements.
- Date 10/14/21, We successfully planned the final mock-up designs. We may make some slight changes in the future but otherwise the mock-ups are ready.
- Date 10/27/21, We worked on the timeline, the project's structure, came up with some names, and designed the basic UML structure.
- Date 10/31/21, We will finish the UML strucutre if we need to make changes, and will start implementing the project.
- Date 11/15/21, We will have 50% of the project implemented.
- Date 11/23/21, We will be almost done with the project, with most of the project being functional.
- Date 11/30/21, The project will be successfully working, meeting all of its basic requirements.

4. Project Structure

MUSIC PLAYER will use a Factory Pattern to load each audio file. The pattern will load each file depending on what file extension it uses. MUSIC PLAYER will support .mp3 and .wav files. If the audio file is not implemented, it will throw an exception saying that the audio file's extension is not supported.

4.1. UML Outline

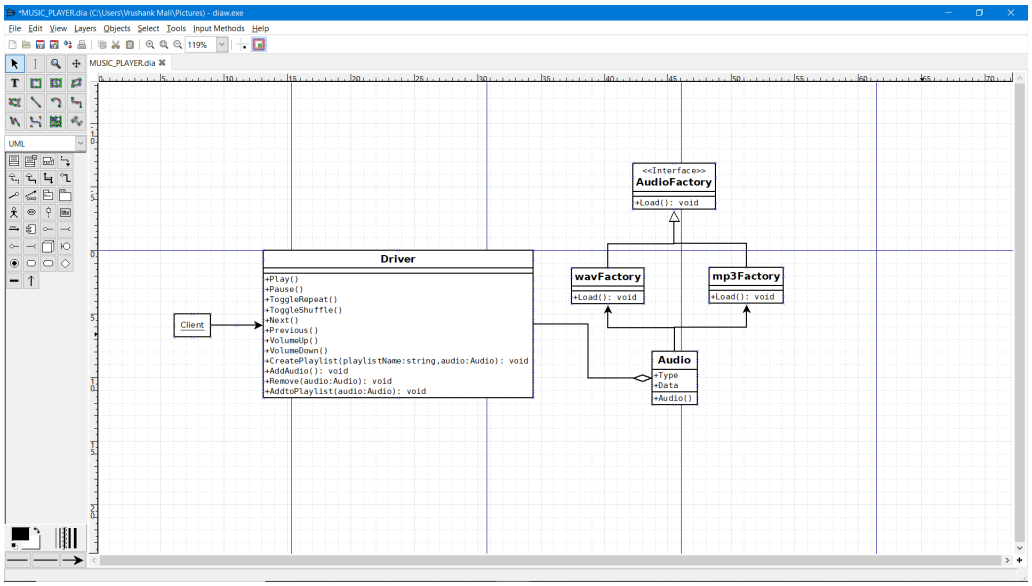

Figure 5. Basic UML design for MUSIC PLAYER

4.2. Design Patterns Used

Make sure to actually use at least 2 design patterns from this class. This is not normally part of such documentation, but largely just specific to this class – I want to see you use the patterns!

5. Results

This section will start out a little vague, but it should grow as your project evolves. With each deliverable you hand in, give me a final summary of where your project stands. By the end, this should be a reflective section discussing how many of your original goals you managed to attain/how many desired use cases you implemented/how many extra features you added.

5.1. Future Work

Where are you going next with your project? For early deliverables, what are your next steps? (HINT: you will typically want to look back at your timeline and evaluate: did you meet your expected goals? Are you ahead of schedule? Did you decide to shift gears and implement a new feature?) By the end, what do you plan on doing with this project? Will you try to sell it? Set it on fire? Link to it on your resume and forget it exists?

References

[1] H. Kopka and P. W. Daly, *A Guide to LaTeX*, 3rd ed. Harlow, England: Addison-Wesley, 1999.