MyTunes - Compulsory Assignment #4

Handed-in by Group E:

1. Christian Hansen
2. Frederik Flagstad
3. Mario Ampudia Valdes
4. Roula Bakr
5. Tienesh Kanagarasan Sivasubremaniyam

Date: 18/12-2019

Table of Content

1. Introduction

The fourth compulsory assignment is about designing and constructing a JavaFXML application, able to provide administration of various songs and playlists and capable of playing songs. Furthermore, is the application expected to contain songs and playlists, allowing the user to choose between various playlists and songs alike.

For the duration of this assignment, groups of 3-4 members are put together. Though, a few groups were not able to be fully composed of the designated number of members, which led to said groups' dispersal and they were put into other groups, resulting in groups to be composed of five members instead.

2. State of Delivery

A GUI is made, a graphical user interface is important for a media player to have. Our MyTunes have 36 songs (right part of the interface), where the user can create a new song by pressing the *New…* button, as well as delete a song by pressing the *Delete*

button. Edit a song doesn't work, the user can perform this action by pressing the *Edit…* button.

3. Functionality

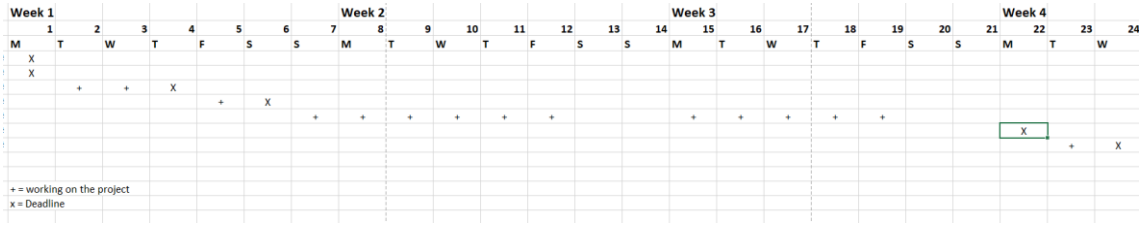
The functionality of the player must be described. You may refer to requirements in the assignment (1-7 mandatory and a) to g) optional).

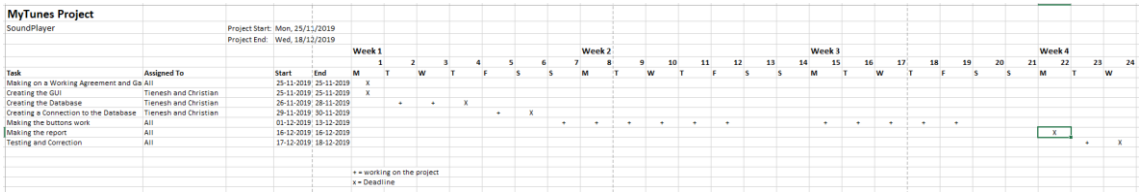
1. The MyTunes project is expected to be a desktop application with a graphical user interface.
2. MyTunes is a traditional music player application, allowing the user to manage the songs and playlists by themselves. For this to be possible, it is expected that there is a song table.
    a. All songs are to be displayed.
    b. The user can sort through this via build-in functionality
    c. The user is allowed access to both edit and create a new song to be added.
        i. A dialog is expected to appear to state the user's choice and aid them. This is to be the same when the user wants to delete a song.
3. A filter function is to be added and allows the user to sort through their playlists for songs by either *artist* or *title.*
4. A dialog is to appear to ensure that the user is permitted to create and edit a new playlist.
5. When a playlist is chosen, the songs contained in that playlist is to be displayed.
6. It has to have the most common function of a music player, meaning that it has to be able to play the next song in the playlist if the previous one is done playing.
7. Playlists and the song lists must be saved to a database using JDBC. This ensures that the songs can be loaded when the program opens.
8. The filetypes the program is allowed to open and play are music files like ".wav" and ".mp3" files.

4. Process Documentation

For the project's duration, we followed the process design, the Gantt Model, allowing the group to assign tasks, to whom should complete said tasks and assign deadlines for the individuals and their work. The Gantt Model also allows us to make a starting date for the tast and the end date of said task.

MyTunes Project				
SoundPlayer		Project Start:	Mon, 25/11/2019	
		Project End:	Wed, 18/12/2019	
Task	**Assigned To**		**Start**	**End**
Making on a Working Agreement and Gantt	All		25-11-2019	25-11-2019
Creating the GUI	Tienesh and Christian		25-11-2019	25-11-2019
Creating the Database	Tienesh and Christian		26-11-2019	28-11-2019
Creating a Connection to the Database	Tienesh and Christian		29-11-2019	30-11-2019
Making the buttons work	All		01-12-2019	13-12-2019
Making the report	All		16-12-2019	16-12-2019
Testing and Correction	All		17-12-2019	18-12-2019

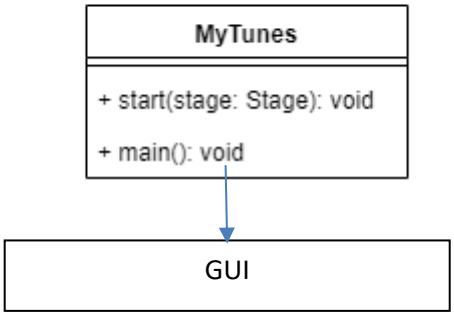




5. Application Structure

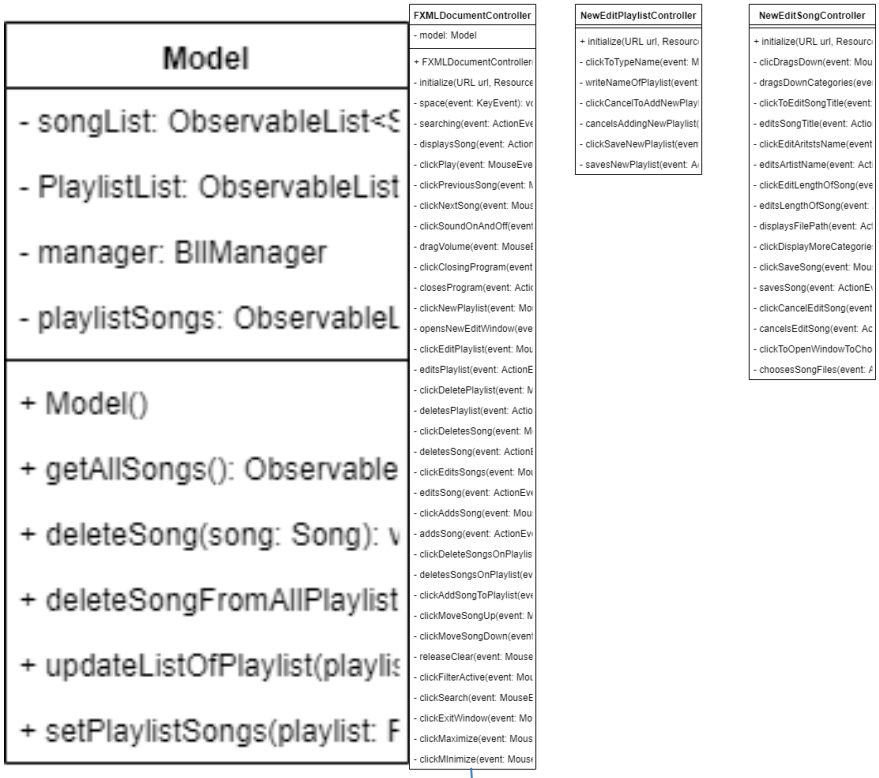
Present one or more UML class diagrams for your mediaPlayer. Remember to comment your diagrams.

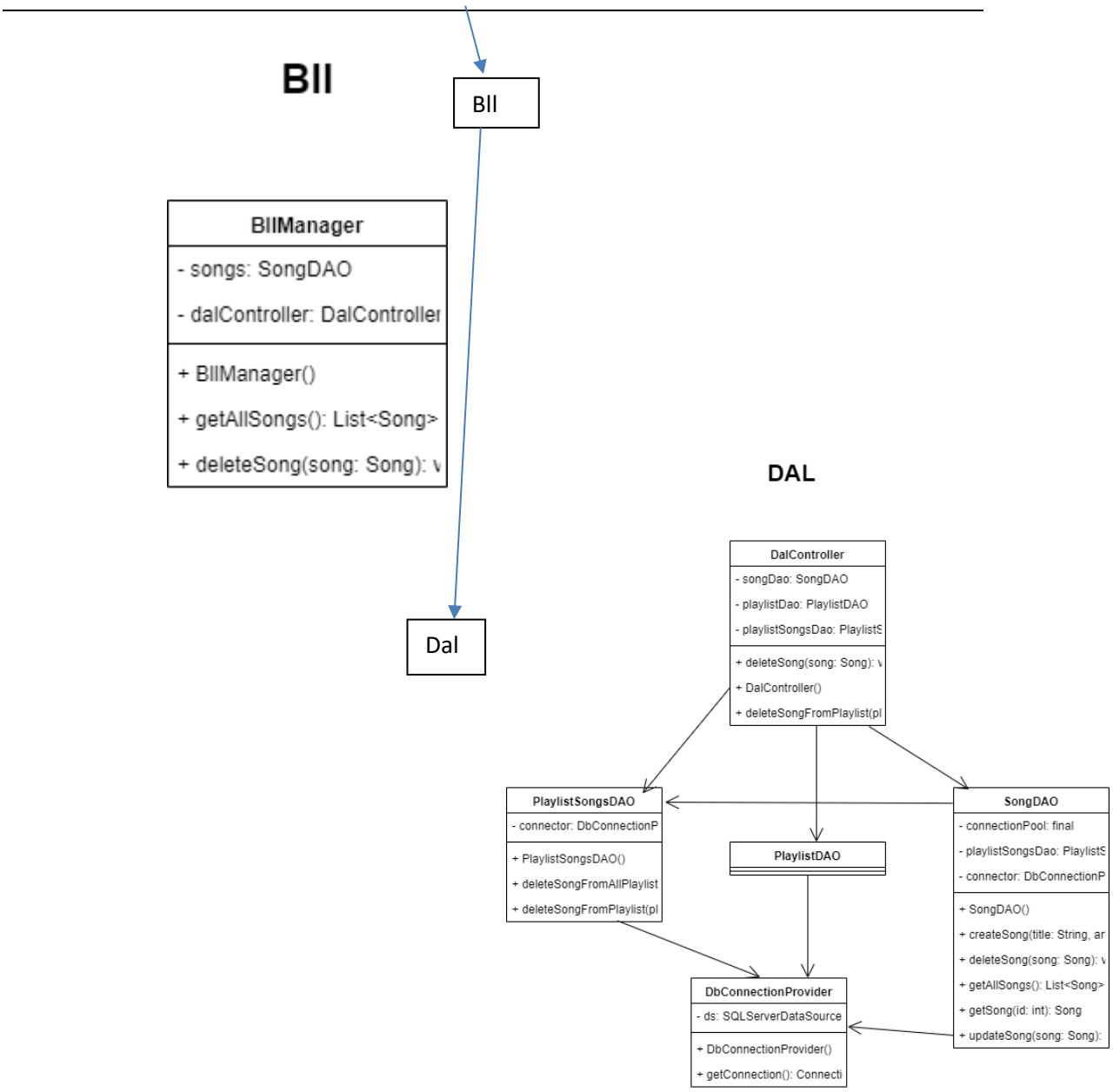
Launching the program



The Model

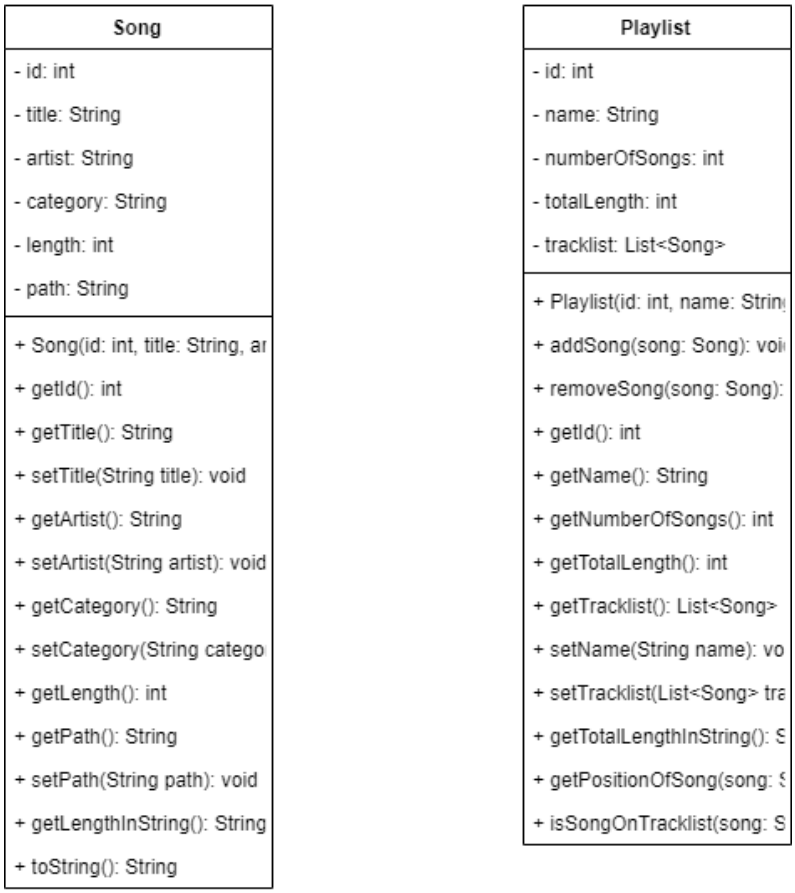
The Controllers




BII
Bll
BllManager
- songs: SongDAO
- dalController: DalController
+ BllManager()
+ getAllSongs(): List<Song>
+ deleteSong(song: Song): v
Dal
DAL
DalController
- songDao: SongDAO
- playlistDao: PlaylistDAO
- playlistSongsDao: PlaylistS
+ deleteSong(song: Song): v
+ DalController()
+ deleteSongFromPlaylist(pl
PlaylistSongsDAO
- connector: DbConnectionP
+ PlaylistSongsDAO()
+ deleteSongFromAllPlaylist
+ deleteSongFromPlaylist(pl
PlaylistDAO
SongDAO
- connectionPool: final
- playlistSongsDao: PlaylistS
- connector: DbConnectionP
+ SongDAO()
+ createSong(title: String, ar
+ deleteSong(song: Song): v
+ getAllSongs(): List<Song>
+ getSong(id: int): Song
+ updateSong(song: Song):
DbConnectionProvider
- ds: SQLServerDataSource
+ DbConnectionProvider()
+ getConnection(): Connecti

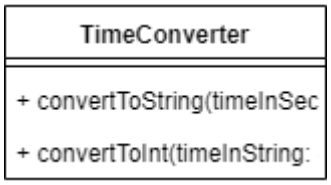
The GUI, Bll and Dal are depending on the Business Entities.

Business Entities



The Song and Playlist classes are depending on the TimeConverter Class.

Util



6. Data Storage

Playlists:

EASV-DB2.ProjectMy...DB - dbo.Playlists

Column Name	Data Type	Allow Nulls
Id	int	☐
Name	nvarchar(MAX)	☐
NumberOfSongs	int	☐
TotalLength	int	☐
		☐

Songs:

EASV-DB2.ProjectM...nesDB - dbo.Songs

Column Name	Data Type	Allow Nulls
Id	int	☐
Title	nvarchar(MAX)	☐
Artist	nvarchar(MAX)	☐
Category	nvarchar(MAX)	☐
Length	int	☐
Path	nvarchar(MAX)	☐
		☐

PlaylistSongs:

EASV-DB2.ProjectMy...dbo.PlaylistSongs

Column Name	Data Type	Allow Nulls
SongId	int	☐
PlaylistId	int	☐
Sequence	int	☐
		☐

There's a foreign key from the table "Songs" to the table "PlaylistSongs".
There is a foreign key between "id" of the table "Songs" and "SongId" of the table "PlaylistSongs".

There is a foreign key between “id” of the table “Playlists” and “PlaylistId” of the table “PlaylistSongs”.

The “Id” of the songs will be assigned automatically by the database.

Database with Songs, including, id, title, artist, category(genre), length of song in seconds and filepath.

EASV-DB2.ProjectM...nesDB - dbo.Songs

Id	Title	Artist	Category	Length	Path
1	For the glory	All the goo...	Rock	286	"\\All the G...
2	On my own	Ashes Rema...	Rock	171	"\\Ashes Re...
3	Be Careful	K-Rino	Hip-Hop	271	"\\Be Caref...
4	Billie Jean	Michael Jac...	Pop	297	"\\Billie Jea...
5	Finesse	Bruno Mars	Pop	223	"\\Bruno M...
6	Zina Zina	Cheb Khaled	Pop	224	"\\Cheb kha...
7	Cant take m...	Frankie Vallie	Jazz	226	"\\Can't tak...
8	Xeribi	Ciwan Haco	Techno	289	"\\Ciwan Ha...
9	Enjoy the sil...	Depeche M...	Techno	455	"\\Depeche ...
10	Devil's Work	Joyner Lucas	Hip-Hop	293	"\\Devil's W...
11	Down Under	Menatwork	Pop	215	"\\Down Un...
12	Shape of You	Ed Sheeran	Pop	263	"\\Ed Sheer...
13	Blessed	Gigi	Hip-Hop	210	"\\Gigi - Ble...
14	When Lege...	Godsmack	Rock	172	"\\Godsmac...
15	Comin in Hot	Hollywood ...	Rock	225	"\\Hollywo...
16	Aglama	Ibrahim Tatl...	Techno	251	"\\Ibrahim T...
17	In the Army ...	Statusquo	Rock	273	"\\In The Ar...
18	ISIS	Joyner Lucas	Hip-Hop	237	"\\ISIS - Joy...
19	Mesmerize	Ja Rule	RnB	347	"\\Ja Rule - ...
20	Animals	Martin Garrix	Techno	191	"\\Martin G...
21	Those Were ...	Mary Hopki...	Pop	300	"\\Mary Ho...
22	Feeling Good	Michael Bu...	Pop	242	"\\Michael ...
23	They Don't ...	Michael Jac...	Pop	281	"\\Michael J...
24	So What	Miles Davis	Jazz	546	"\\Miles Da...
25	Modern Talk...	Brother Lou...	Pop	224	"\\Modern ...
26	My heart wil...	Celine Dion	Pop	305	"\\My heart ...
27	Feed Your ...	Paul Kalkbre...	Techno	377	"\\Paul Kalk...
28	Push the lim...	Enigma	Techno	318	"\\Push the l...
29	Put Your He...	Harrison Cr...	Jazz	250	"\\Put Your ...
30	Rain and Wi...	K-Rino	Hip-Hop	285	"\\Rain and ...

7. Implementation Details

We couldn't implement any sorting functionality for the song columns, so the user can't sort them by name or duration. The filter query also couldn't be implemented, as we focused on that we consider the important things first.
The next stage, playlists, although the interface is there, and the application lets you write a name for a playlist, we couldn't make it to work, therefore the playlist can't be saved. Consequently, songs can't be added to a playlist. Our user interface contains controls for playing songs, but we only could make the play button to work. Go back and skip song buttons are not implemented. Th application can play both .wav and .mp3 formats.

The application works with a three-layer-model architecture, a GUI layer controls the interaction with the user, a business layer contains the business logic, and the songs are available through the data access layer, which contains the access to the file system. Everything is on a Github repository, the one used to develop the application by the 5 members

8. Source Control

Throughout the duration of the compulsory assignment, the group utilized the tool GitHub for sharing the coding of the project. Each group member connected to GitHub and downloaded the desktop application, "GitHub Desktop". Via this, the group was able to download the original MyTunes project created in Christian's repository. Any improvements done by any member will be committed to Github, allowing the other members to receive the improved project with the new added changes.

The name of the repository is: "MyTunes"

9. Source Code

The link for Group E's GitHub repository during the fourth compulsory assignment: "https://github.com/ChristianH321/MyTunes.git