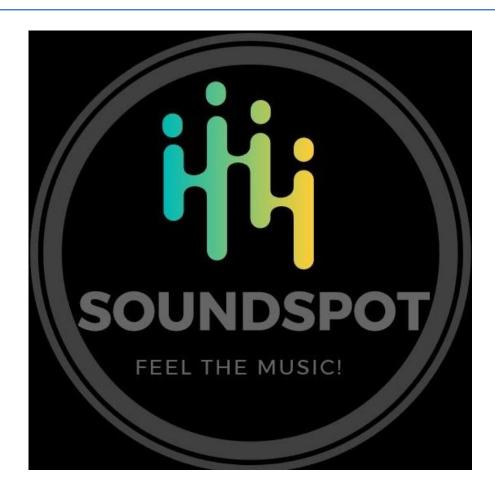


# MyTunes - Compulsory Assignment #4



# **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**

TAE	BLE OF CONTENT	2
1.	INTRODUCTION	2
2.	STATE OF DELIVERY	2
3.	FUNCTIONALITY	3
4.	PROCESS DOCUMENTATION	4
5.	APPLICATION STRUCTURE	4
6.	DATA STORAGE	8
7.	IMPLEMENTATION DETAILS	10
8.	SOURCE DETAILS	10
9.	SOURCE CODE	10

#### Introduction 1.

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.

#### **State of Delivery** 2.

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

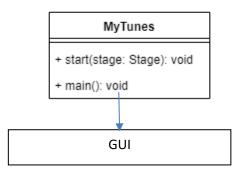
For the project's duration, we followed the process design, the Gantt Model, allowing the group to assign tasks, to whom should complete said tasks and assign deadlines for the individuals and their work. The Gantt Model also allows us to make a starting date for the tast and the end date of said task.

M	yΤι	ıne	s P	ro	jed	t																						
Sou	ındF	Play	er																Pr	roje	t Sta	art:	М	on,	25/1	1/2	2019	
																			Pr	roje	t En	d:	W	ed,	18/1	2/2	2019	
Tas	k										Δ	ssig	ne	ч.	Γο								Sta	art		Er	nd	
		n a V	Vorki	ing A	Agree	emer	nt a	nd G	ant	t	Al		,												-2019			2019
	iting				.6					_			sh	and	l Ch	rist	ian								-2019	_		
			)atab	ase							Tie	ene	sh	and	l Ch	rist	ian								-2019			
			nnec		to th	ne D	atal	base							d Ch										-2019	-		
			itton								Al	All												-2019	-			
	ing t										All									16-12-2019			1	16-12-2019				
Test	ing a	nd C	orrec	tion							Αl												17	-12	-2019	1	8-12-	2019
																										i		
Week		2	3	4	5	6	7	Week 2 8		9	10		11	12	2	13	14	Veek 3		16	17	18	19	2	0 21	Weel		23 2
M X	Т	W	Т	F	S	S	- 1	М	Т	W		т	F		S	S	- 1	И	Т	w	Т	F	S		S	М	Т	w
Х	+	+	X	+																								
					Х		+	+	+		+	+		+				+		+	+		+			X		
																											+	X
+ = work	ing on the	project																										
x = Dead																												
	s Project																											
SoundPlaye	r				Project Start: Project End:																							
								Week 1	2	3	4	5		6	Week 7	8	9 10	11	12	13	Week 3	16	17	18	19 20	2	Week 4 1 22	23 24
Task Making on a	Working Agree	ment and Ga	Assigned To All			25-11-2019	End 25-11-20	M 19 X	т	w	T	F	S	S	М	т	w	T	5	s	М	т (	w T		S	S	м т	w
Creating the Creating the	Database		Tienesh and Tienesh and	Christian		25-11-2019 26-11-2019	28-11-201	19			х																	
Creating a Co	nnection to th uttons work	e Database	Tienesh and I	Christian		29-11-2019 01-12-2019	13-12-20	19				•	X						٠									
			All			16-12-2019																					X	
Making the r Testing and			All			17-12-2019																					_	+ X

# 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

	FXMLDocumentControll
**II	- model: Model
Model	+ FXMLDocumentControll
	- initialize(URL url, Resour
I	- space(event; KeyEvent);
<ul><li>songList: ObservableList&lt;§</li></ul>	- searching(event: ActionE
_	- displaysSong(event: Acti
	- clickPlay(event: MouseE
<ul> <li>PlaylistList: ObservableList</li> </ul>	- clickPreviousSong(event
,	- clickNextSong(event: Mo
	- clickSoundOnAndOff(eve
<ul> <li>manager: BIIManager</li> </ul>	- dragVolume(event: Mous
managor. Diimanagor	- clickClosingProgram(eve
	- closesProgram(event: Ad
- playlistSongs: ObservableL	- clickNewPlaylist(event: N
- playlistourgs. Observablet	- opensNewEditWindow(e
	- clickEditPlaylist(event: M
	- editsPlaylist(event: Actio
· Madal/\	- clickDeletePlaylist(event
+ Model()	- deletesPlaylist(event: Ac
	- clickDeletesSong(event:
	- deletesSong(event: Actio
+ getAllSongs(): Observable	- clickEditsSongs(event: N
	- editsSong(event: ActionE
	- clickAddsSong(event: Mi
+ deleteSong(song: Song): v	- addsSong(event: Action8
	- clickDeleteSongsOnPlay
	- deletesSongsOnPlaylist(
+ deleteSongFromAllPlaylist	- clickAddSongToPlaylist(e
activity of the state of the st	- clickMoveSongUp(event
	- clickMoveSongDown(eve
+ updateListOfPlaylist(playlis	- releaseClear(event: Mou
· updateListOff laylist(playlis	- clickFilterActive(event: M
	- clickSearch(event: Mous
+ cotBlaylistCongs/playlist: E	- clickExitWindow(event: N
+ setPlaylistSongs(playlist: F	- clickMaximize(event: Mo
	- clickMlnimize(event: Mou

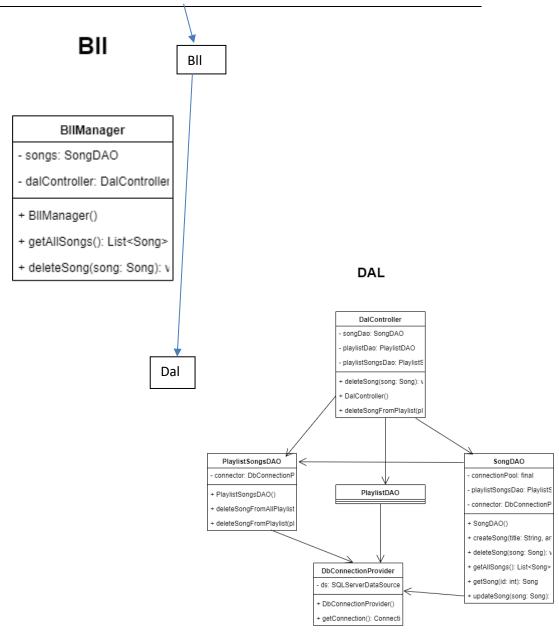
NewEditPlaylistController

initialize(URL url, Resourc
- cikchToTypeName(event. M
- writeNameOfFlaylist(event
- cikckCancelToAddNewPlay)
- cancelsAddingNewPlaylist(event
- cikckSaveNewPlaylist(event
- savesNewPlaylist(event. A)

NewEditSongController

initialize(URL url, ResourccilicPogsDownCevent. Mou
dragsDownCategories(evecilicTogsDownCategories(evecilicTogsTollegevent. ActiocilicLeditArdistsName(event. ActiocilicLeditArdistsName(event. ActiocilicLeditIngsthOrSong(event. ActiocilicLeditIngsthOrSong(event. displaysFilePath(event. ActicilicLicDisplayMoreCategoriecilicLosaveSong(event. Mou
savesSong(event. ActionEcilicCancelEditSong(event. cancelEditSong(event. cancelEditSong(event. cancelEditSong(event. cancelEditSong(event. cancelSong(event. cancelS





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

#### **Business Entities**

Song
- id: int
- title: String
- artist: String
- category: String
- length: int
- path: String
+ Song(id: int, title: String, ar
+ getId(): int
+ getTitle(): String
+ setTitle(String title): void
+ getArtist(): String
+ setArtist(String artist): void
+ getCategory(): String
+ setCategory(String catego
+ getLength(): int
+ getPath(): String
+ setPath(String path): void
+ getLengthInString(): String
+ toString(): String

Playlist
- id: int
- name: String
- numberOfSongs: int
- totalLength: int
- tracklist: List <song></song>
+ Playlist(id: int, name: String
+ addSong(song: Song): voi
+ removeSong(song: Song):
+ getId(): int
+ getName(): String
+ getNumberOfSongs(): int
+ getTotalLength(): int
+ getTracklist(): List <song></song>
+ setName(String name): vo
+ setTracklist(List <song> tra</song>
+ getTotalLengthInString(): S
+ getPositionOfSong(song: §
+ isSongOnTracklist(song: S

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

### Util

#### TimeConverter

- + convertToString(timeInSec
- + convertToInt(timeInString:



# 6. Data Storage

Playlists:

EA	SV-DB2.ProjectMyDB - dbo.Playlists	→   X	
	Column Name	Data Type	Allow Nulls
₽¥	ld	int	
	Name	nvarchar(MAX)	
	NumberOfSongs	int	
	TotalLength	int	

Songs:

EA:	SV-DB2.ProjectMnesDB - dbo.Songs	+ X	
	Column Name	Data Type	Allow Nulls
▶8	Id	int	
	Title	nvarchar(MAX)	
	Artist	nvarchar(MAX)	
	Category	nvarchar(MAX)	
	Length	int	
	Path	nvarchar(MAX)	

PlaylistSongs:

EAS	SV-DB2. Project Mydbo. Playlist Songs	+ X	
	Column Name	Data Type	Allow Nulls
₽Ÿ	Songld	int	
P	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.

:AS	V-DB2.ProjectM	nesDB - dbo.S	ongs 🕆 🗶			
	ld	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	Рор	297	"\\Billie Jea
	5	Finesse	Bruno Mars	Рор	223	"\\Bruno M
	6	Zina Zina	Cheb Khaled	Рор	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	Рор	215	"\\Down Un
	12	Shape of You	Ed Sheeran	Рор	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	"\\lbrahim 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	Рор	300	"\\Mary Ho
	22	Feeling Good	Michael Bu	Рор	242	"\\Michael
	23	They Don't	Michael Jac	Рор	281	"\\Michael J
	24	So What	Miles Davis	Jazz	546	"\\Miles Da
	25	Modern Talk	Brother Lou	Рор	224	"\\Modern
	26	My heart wil	Celine Dion	Рор	305	"\\My heart .
	27	Feed Your	Paul Kalkbre	Techno	377	"\\Paul Kalk
	28	Push the lim	Enigma	Techno	318	"\\Push the I.
	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