

Choreographing Events Programmers Guide

05/07/2019

Maggie Krummel

1 Project Overview

- 1.1 Definition..... pg. 3
- 1.2 Approach..... pg. 3
- 1.3 Solution Connectivity Diagram..... pg. 3

2 Software Installation/Setup

- 2.1 Visual Studio 2017..... pg. 3
- 2.2 SQL Server Express..... pg. 4
- 2.3 Arduino Uno..... pg. 5
- 2.4 Arduino IDE..... pg. 5

3 Data Storage

- 3.1 SQL Server Express - Song Table Definition..... pg. 6
- 3.2 Local FS - Events Storage..... pg. 6

4 Visual Studio Application

- 4.1 Application User Interfaces..... pg. 7
- 4.2 Startup Flow..... pg. 7
- 4.3 Application User Interfaces Navigation Data Flow..... pg. 8
- 4.4 Search Songs View (SongsSearchView.xaml) pg. 9
- 4.5 Add/Edit Song View (AddEditSongView.xaml) pg. 10
- 4.6 Events View..... pg. 11

5 How Events Fire

- 5.1 Arduino IDE/Uno Relationship..... pg. 11
- 5.2 Event Execution..... pg. 11

6 Exceptions

- 6.1 View Queue..... pg. 12
- 6.2 A Seconds or So Delay in Lights Execution..... pg. 12

7 Helpful Hints

- 7.1 Memory..... pg. 12

8 Royalty Free Audio and Icons

- 8.1 Music..... pg. 12
- 8.2 Icons..... pg. 12

1. Project Overview

1.1 Definition

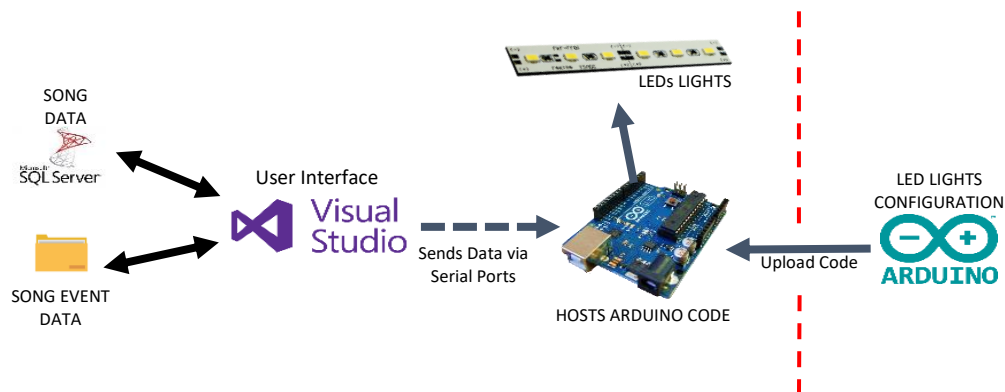
This application allows a user to synchronize LED events with the playing of an audio file.

1.2 Approach

This application uses a WPF app in C# in Visual Studio for the user interface. MySQL express for the database as well as local fs for the storage of larger files such as the audio, images, and event configuration.

For the LED light configuration, the Arduino language is used and is hosted by an Arduino Uno.

1.3 Solution Connectivity Diagram

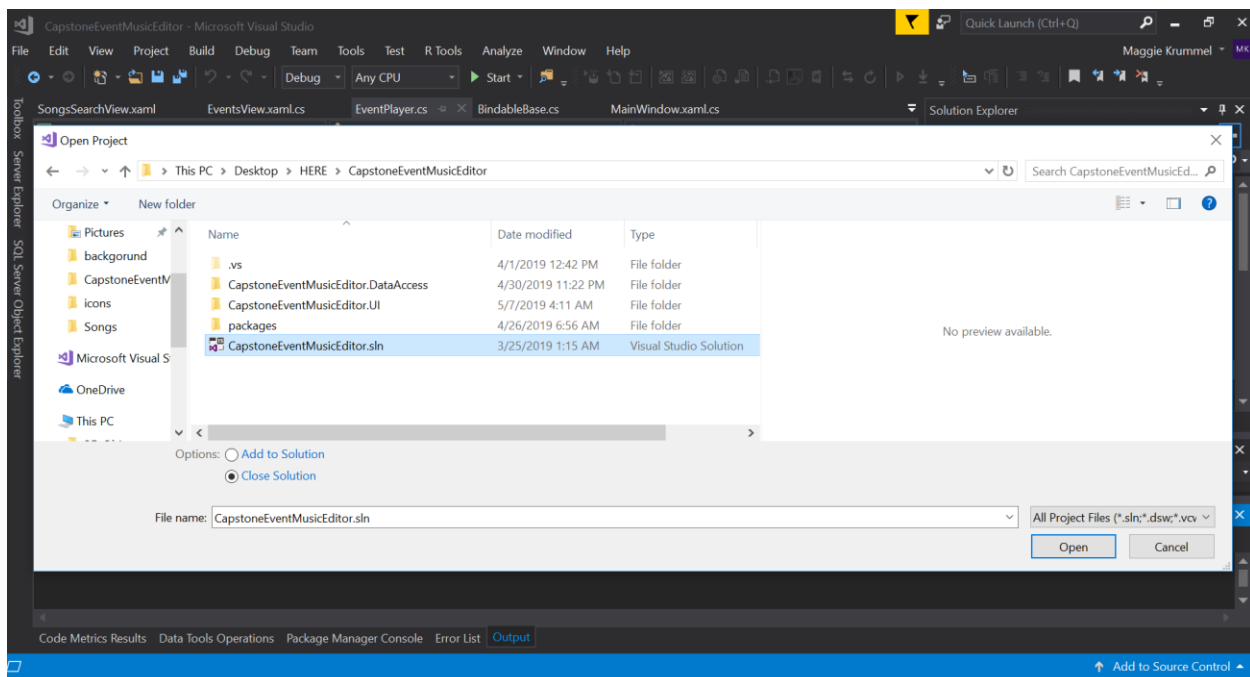


2 Software Installation/Setup

2.1 Visual Studio 2017

Download Visual Studio 2017 from <https://visualstudio.microsoft.com/vs/> and the folder labeled User Application from <http://compsci02.snc.edu/cs460/2019/krummm/project-resources.html>.

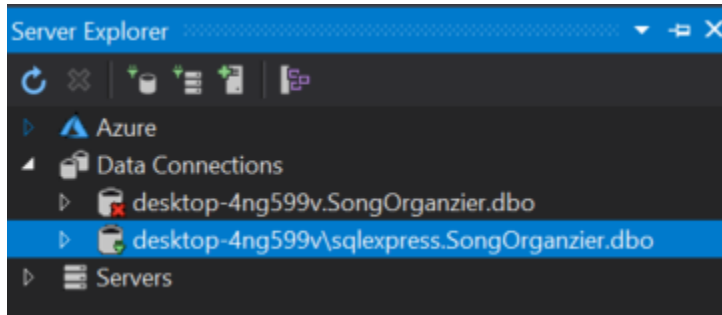
Unzip the folder then open the solution in Visual Studio. You must set up the database before you can successfully run the application.



2.2 SQL Server Express

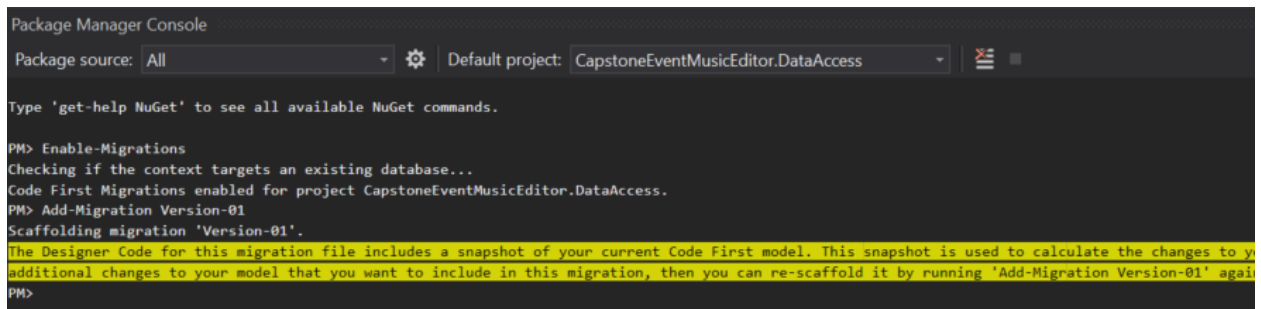
Download SQL Server Express edition from <https://www.microsoft.com/en-us/sql-server/sql-server-editions-express>.

After you install and run SQL Server Express on your machine and establish a connection. In Visual Studio open the Server Explorer and add the sqlexpress connection to it and choose a database name.



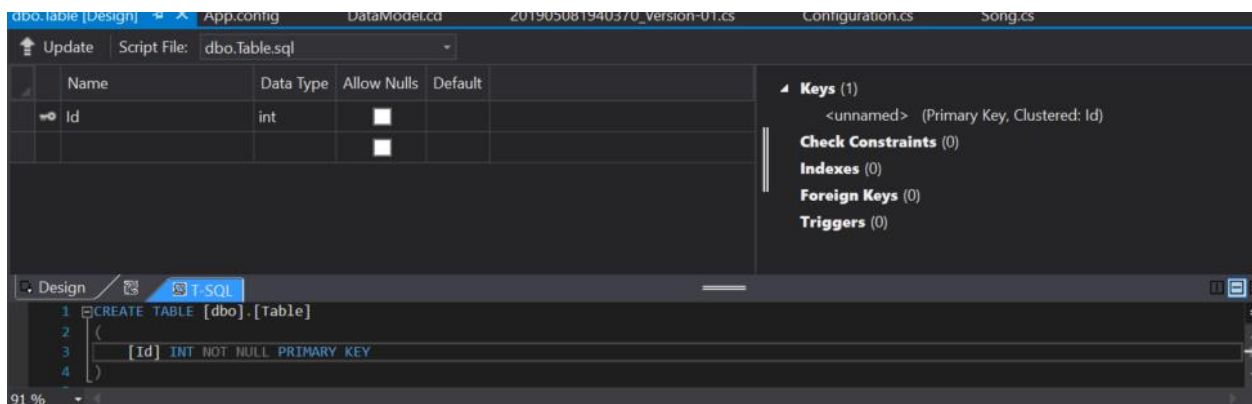
If you choose a name different then SongOrganzier.dbo then you must go into *CapstoneEventMusicEditor.DataAccess/App.Config* and change the *connectionstring* on line 8 where it reads *database=SongOrganzier*.

Next you open the package manager in Visual Studio and run type in *Update-Database* making sure the Default project is set to *CapstoneEventMusicEditor.DataAccess*.



Once all of these steps are complete you should be able to run the application. There will be no songs in the database as sharing music is frowned upon. To download free songs please go to <https://www.bensound.com/royalty-free-music/2> or you are more than welcome to use songs you already have on your computer.

***In case of failure table definition is located on <http://compsci02.snc.edu/cs460/2019/krummm/project-resources.html>. Simply right click on the tables folder in the Server Explorer under your database and click *Add New Table*. Then put in the table definition and click the *Update* button.



2.3 Arduino Uno

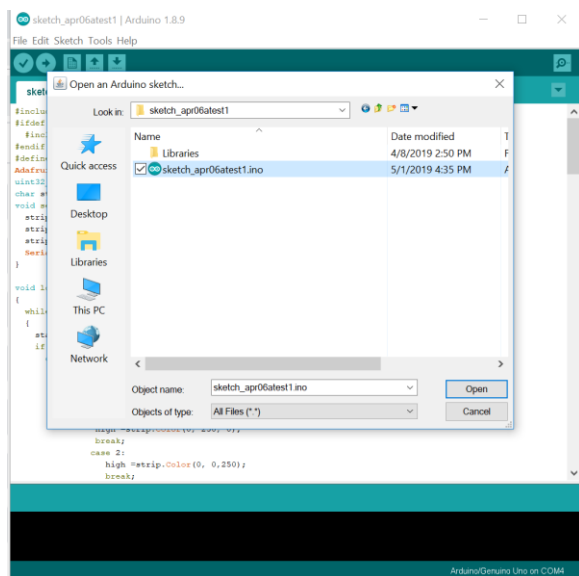


2.4 Arduino IDE

2.4.1 Setup

Download the Arduino IDE from <https://www.arduino.cc/en/Main/Software> and the project folder labeled Arduino Code from <http://compsci02.snc.edu/cs460/2019/krummm/project-resources.html>.

After the downloads are complete unzip the Arduino code folder and open the project in the Arduino IDE.

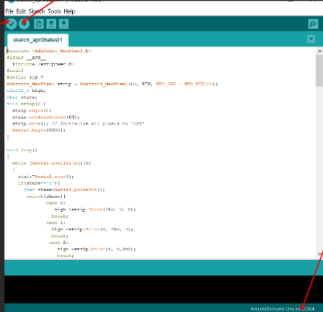


2.4.2 Uploading the Code

Once the project is uploaded you can simply plug in the Arduino Uno and compile and upload your code to it. If you are not using com4 you must go into the C# application under *CapstoneEventMusicEditor.UI/EventEditor/EventPlayer.cs* and change the static variable named *mySerialPort* to equal your new serial port.

3. Upload Code to Arduino Uno

1. Compile code



2. Make sure your Arduino is plugged into com4 (otherwise you must change code in the App (in Visual Studios)).

If port number is changed go to CapstoneEventMusicEditor.UI/EventEditor/EventPlayer.cs and change the static variable named mySerialPort to equal your new serial port.

3 Data Storage

3.1 SQL Server Express - Song Table Definition

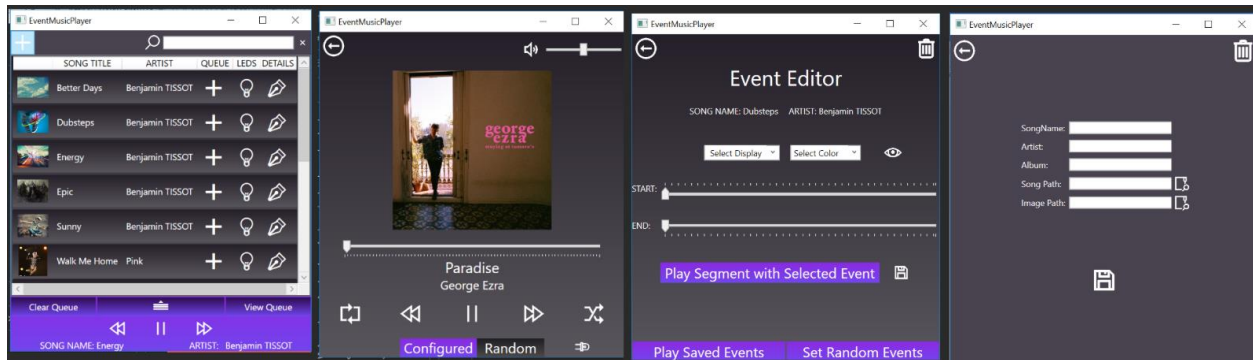
dbo.Song				
Name	Data Type	Allow Nulls	Description	Value Set By
Id	Int	Not Null	Primary key, unique identifier	Code
SongName	Nvarchar	Not Null	Name of Song	User Input
Album	Nvarchar	nullable	Name of Album	User Input
Artist	Nvarchar	Nullable	Name of Artist	User Input
Image	Nvarchar	Nullable	File path to image on local fs	User Input
PathToSong	Nvarchar	Not Null	File path to song on local fs	User Input
SongLength	Int	Not Null	Song length	Code
FilePathToEvent	Nvarchar	Nullable	File path to event configuration on local fs	Code
DoesEventExist	Bit	Not Null	Has the user created an event configuration file for the song	Code
EventSeconds	Int	Not Null	How frequently events occur during the song. Eventually this could be configurable by user input on a per song basis	Code

3.2 Local FS - Events Storage

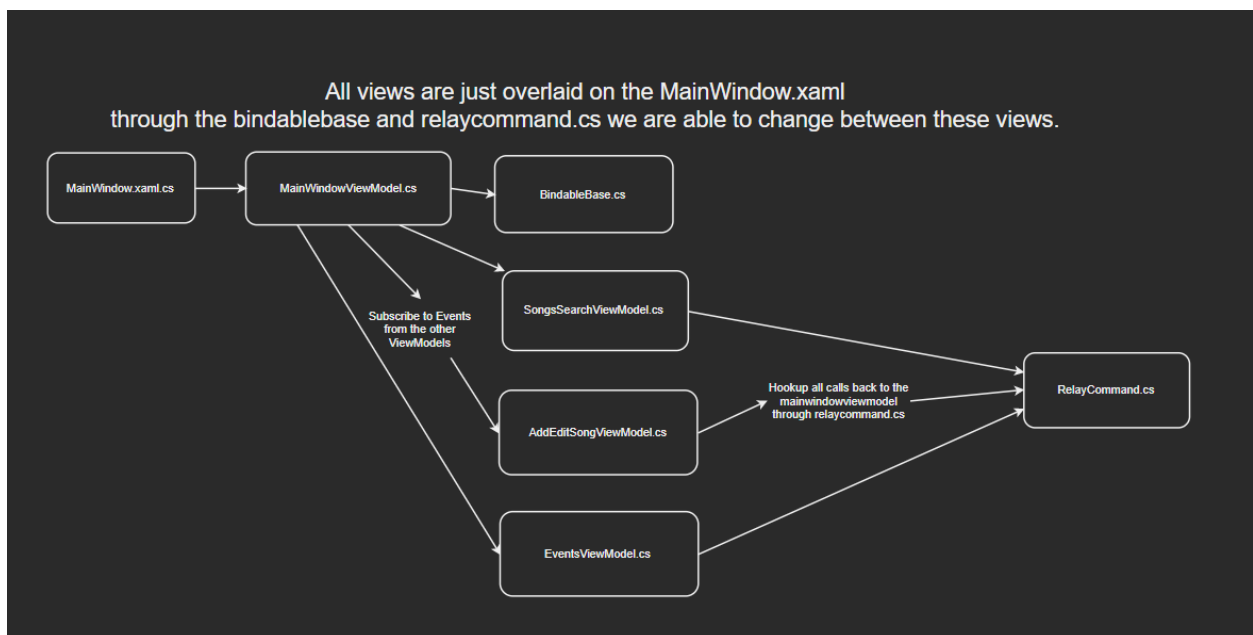
Mini Example: <pre><?xml version="1.0" encoding="utf-8"?> <ArrayOfEventConfig xmlns:xsi="http://www.w3.org/2001/XMLSchema-instance" xmlns:xsd="http://www.w3.org/2001/XMLSchema"> <EventConfig> <FcnNumber>1</FcnNumber> <Color>0</Color> </EventConfig> <EventConfig> <FcnNumber>2</FcnNumber> <Color>2</Color> </EventConfig> </ArrayOfEventConfig></pre>
<p>This is an array of the EventConfig struct. You can find this in CapstoneEventMusicEditor.UI/EventEditor/EventPlayer.cs. It stores what events are played when during a song. In this version of the event storage the user can only decide what lighting function they wish to call (FcnNumber) and what color they wish that to display in (Color). This is easily expandable though if you wish to add a configuration setting such as speed.</p>

4 Visual Studio Application

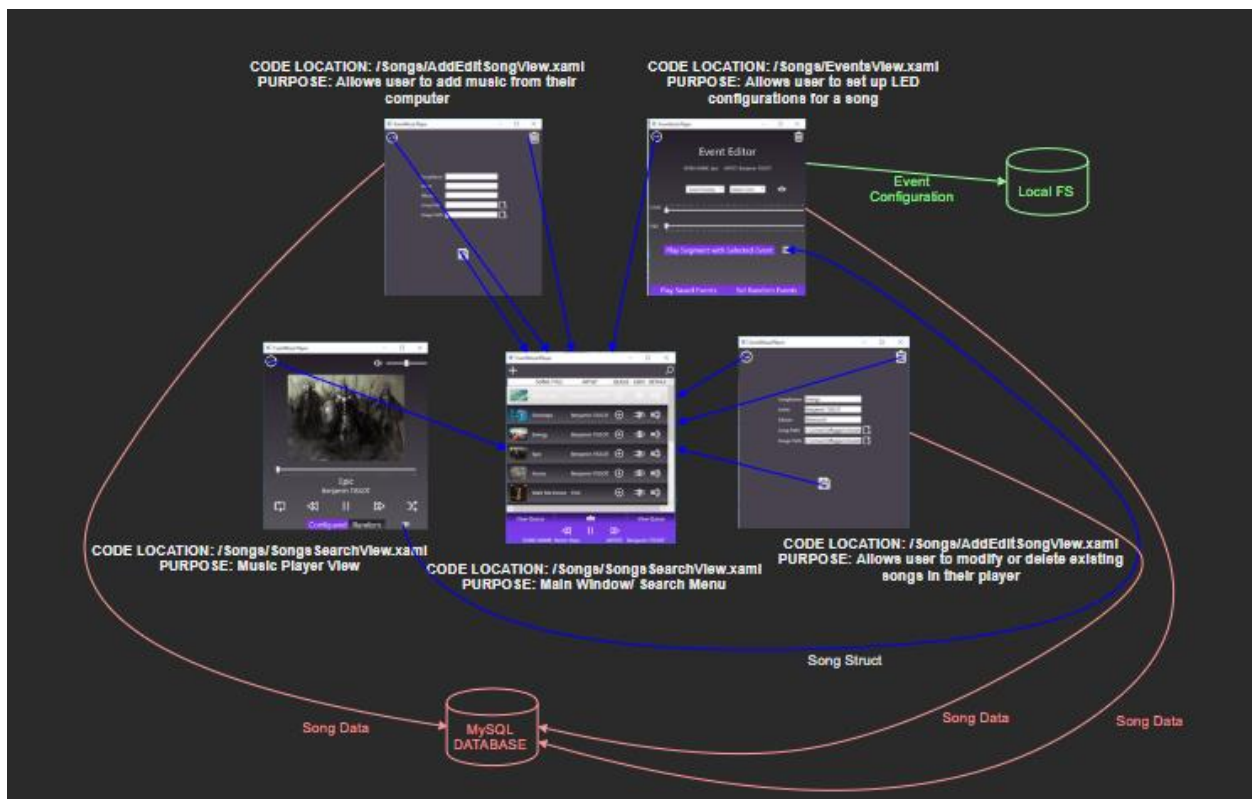
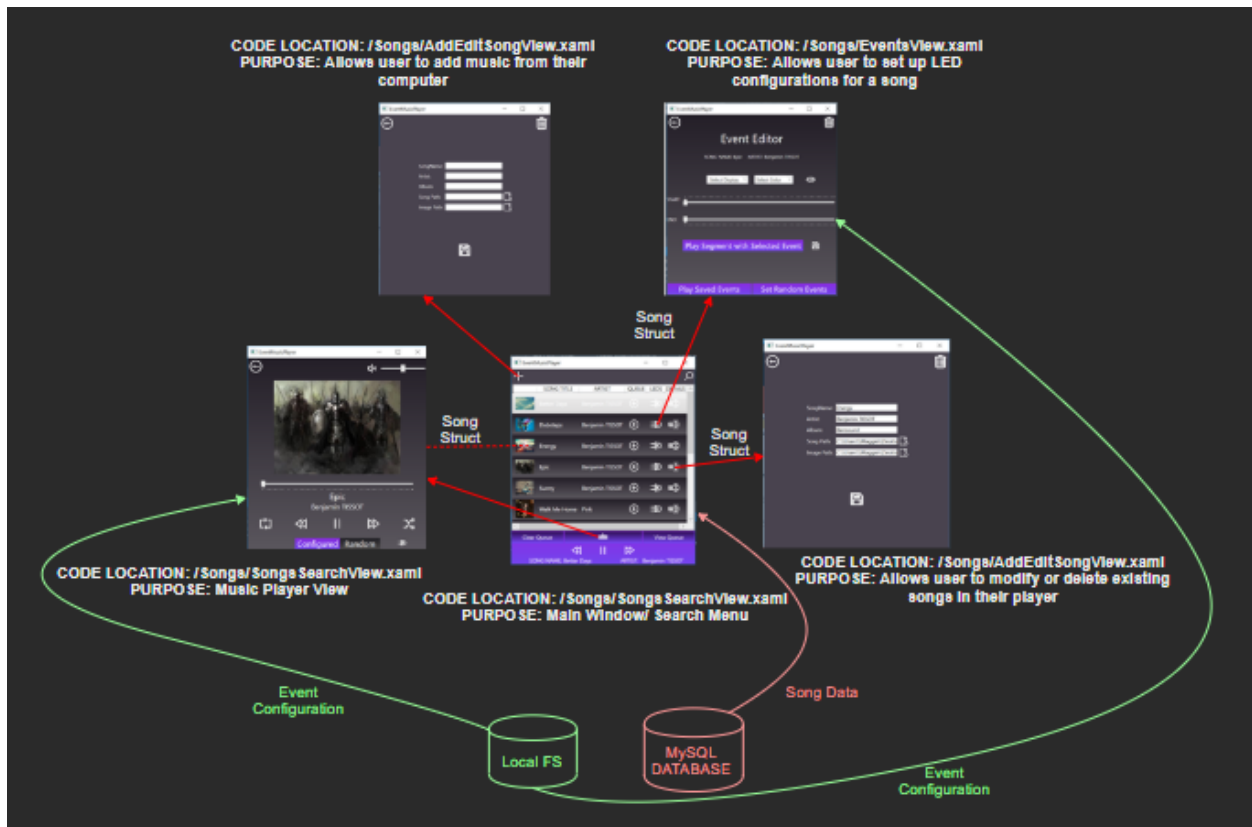
4.1 Application User Interfaces



4.2 Startup Flow



4.3 Application User Interfaces Navigation Data Flow

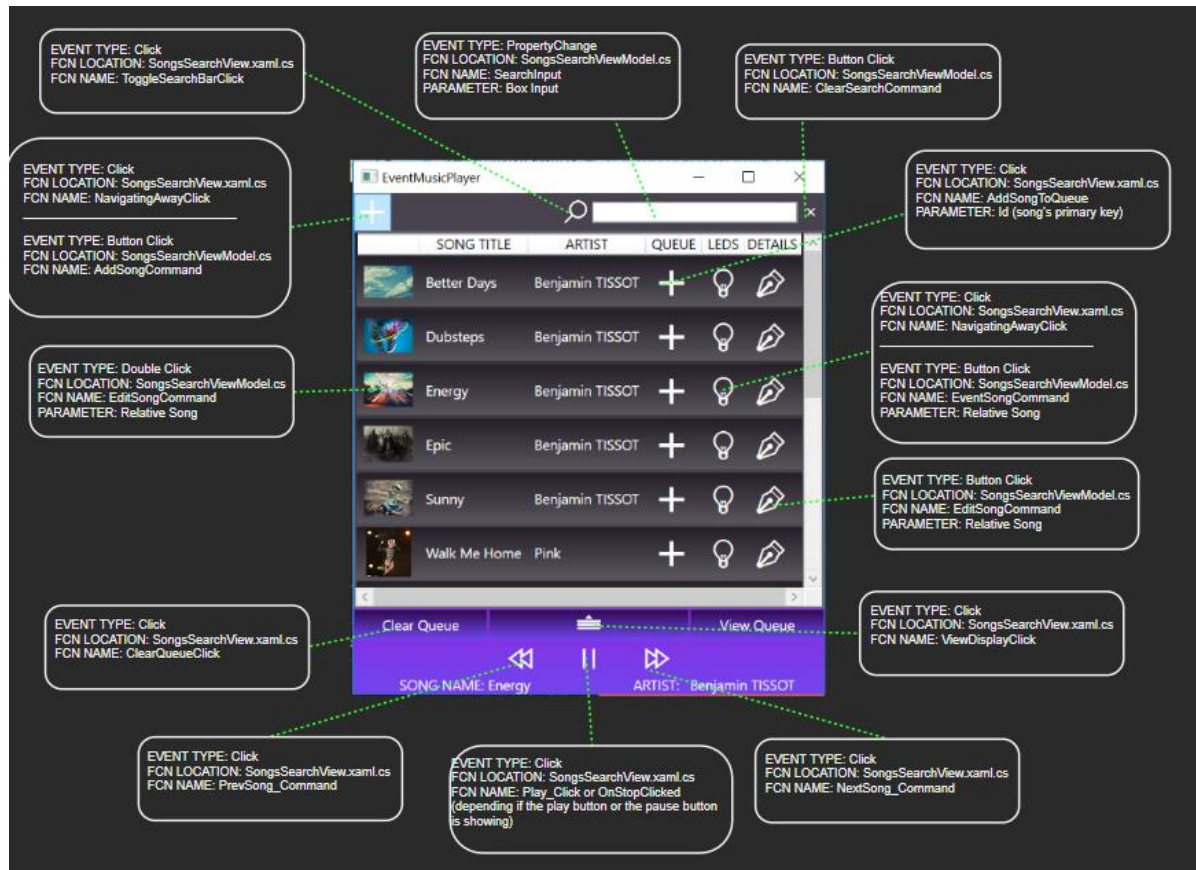


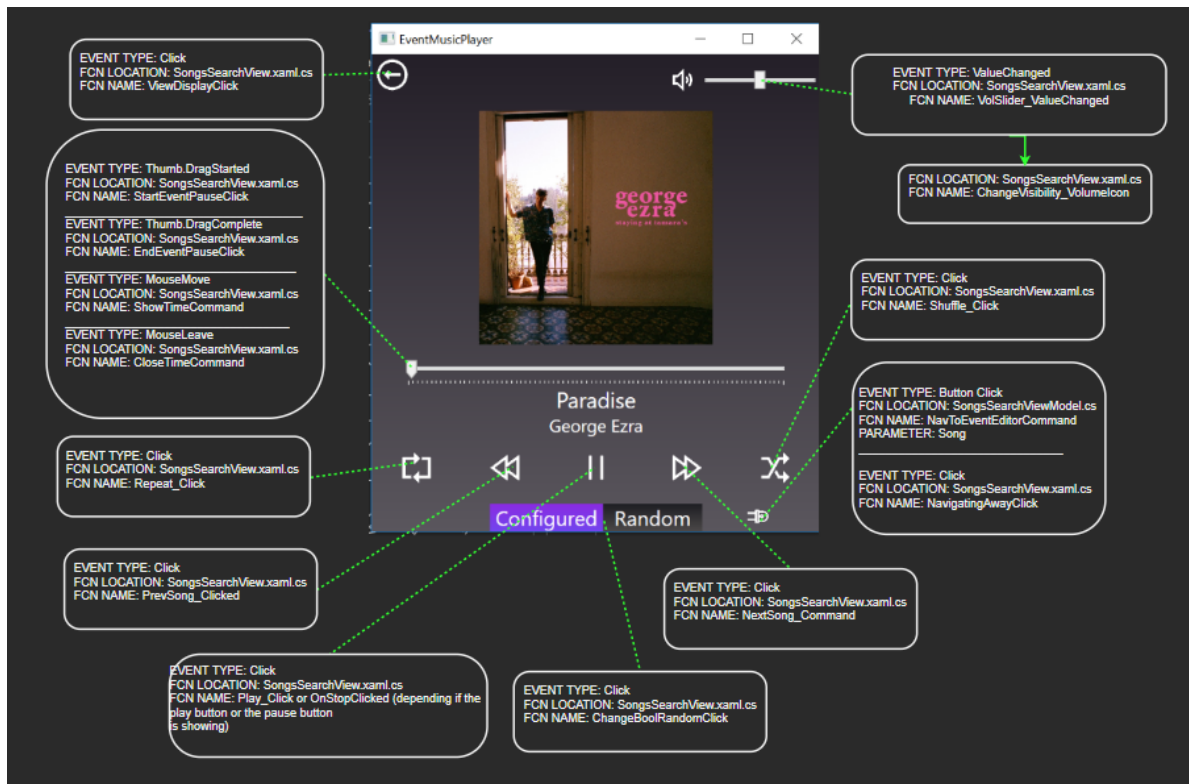
4.4 Search Songs View (SongsSearchView.xaml)

4.4.1 Purpose

This user interface is the startup screen. Here users can toggle between a full-size view of the song that is currently playing or a smaller view which allows them to also view all song they have in their library.

4.4.2 User Interface Walk Through



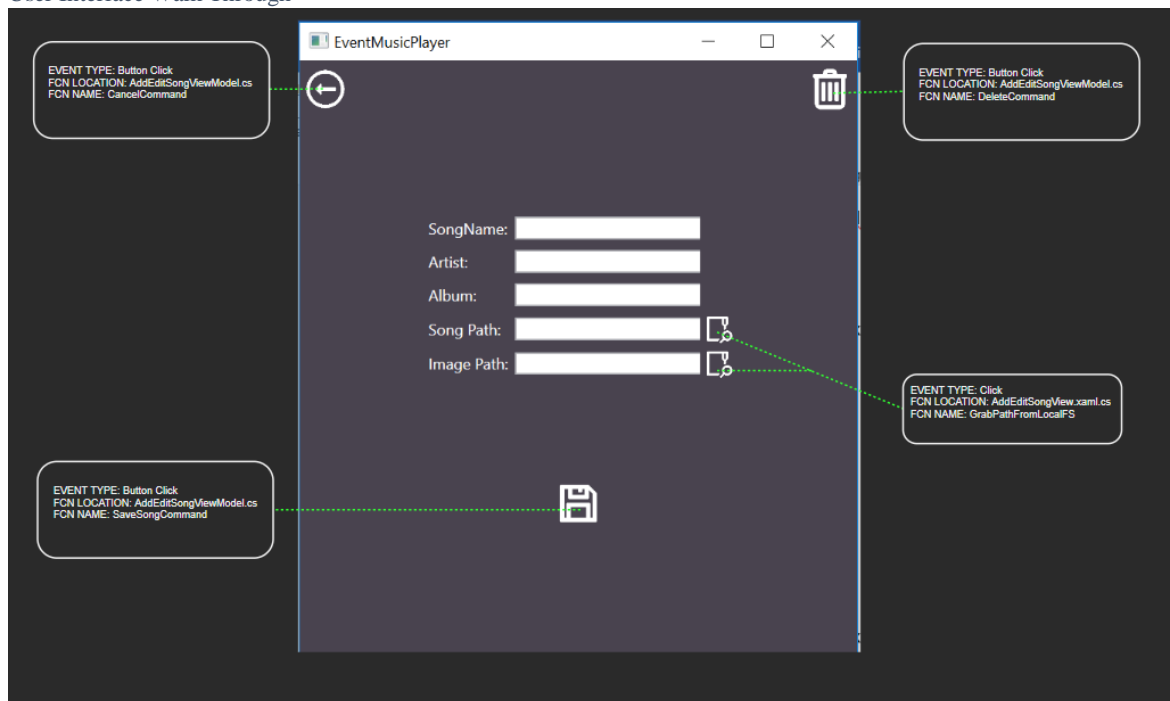


4.5 Add/Edit Song View (AddEditSongView.xaml)

4.5.1 Purpose

This user interface allows the user to upload a new song from their own song collection or edit details about songs they currently have in their library.

4.5.2 User Interface Walk Through

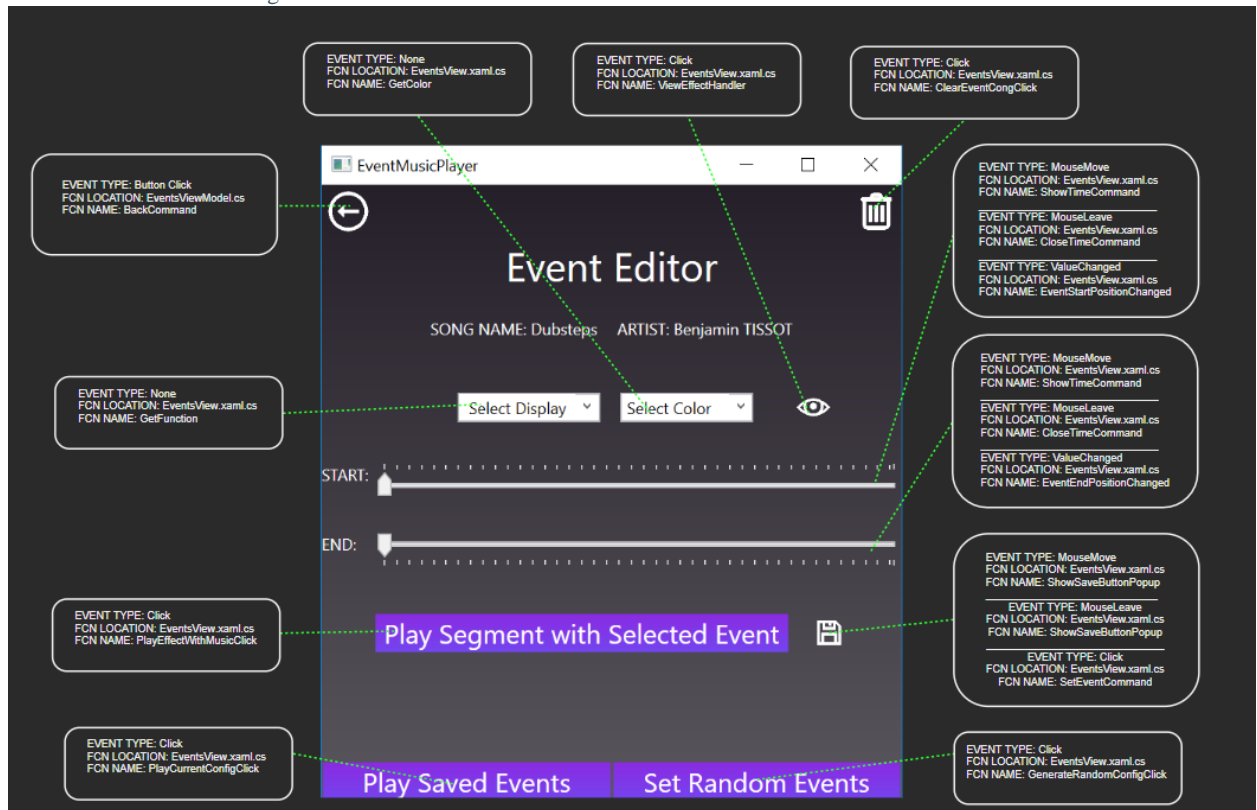


4.6 Events View

4.6.1 Purpose

This user interface allows the user to set events for a specific song. Currently the configuration is set for every 3 seconds meaning that the user can pick a lighting function to run for any 3 second interval during the song.

4.6.2 User Interface Walk Through



5 How Events Fire

5.1 Arduino IDE/Uno Relationship

The code is compiled on the Arduino ide and then uploaded to the Arduino Uno. The Arduino Uno runs the program. The IDE is no longer necessary once the program is uploaded.

5.2 Event Execution

First the C# app checks if an existing configuration is set for the song in the file directory. If yes, then it will use that otherwise it will fire random events. The C# application writes out to Arduino every time an event is fired (sends an interrupt). This places the bytes into the outgoing buffer, and the buffer is then emptied asynchronously in the background by the TX-ready interrupt on the Arduino. Once received Arduino parses it and executes the Function calls.

6 Exceptions

6.1 View Queue

Did not get around to implementing a view for the queue. The implementation could simply be a toggle between all songs and view queue, with the same population technique.

6.2 A Seconds or So Delay in Lights Execution

There is a little less than a second delay each time a new light function is called. I haven't found a way to solve this except for using a different board, but a way to lessen the impact would be to send in less commands rather just tell the functions how many times to execute before a new function is called.

Ex: If you are using the strobe for a 21 second duration send in strobe and then 7 because 7 events occur in 21 seconds. That way the function can just keep looping without having to process anything.

7 Helpful Hints

7.1 Memory

Make sure you properly dispose of both the reader and the WaveOut every time you start a new song otherwise you will run out of memory.

8 Royalty Free Audio and Icons

8.1 Music

<https://www.bensound.com/royalty-free-music/2>

8.2 Icons

<https://icons8.com/>