

"Ismen"

Technical Specification for Desktop Application

Contents

1. The goal.....	3
2. Used Terminology.....	3
3. System Requirements.....	4
4. Used Terminology and Abbreviations.....	5
5. User interface requirements.....	5
5.1 MainWindow.....	6
5.2 Description of tabs.....	6
5.2.1 Show Tab.....	
5.2.2 Actor Track:.....	7
5.2.3 Timeline Widget.....	9
5.2.4 Project file structure.....	10
5.3 Suit Editor	
Tab.....	11
5.4 Effect Editor	
Tab.....	12
6. Script and audio playback.....	13
7. User stories:.....	15
7.1 Create new project (scenario).....	15
7.2 Upload project to suit boards.....	16
8. Possible features that will be implemented later (TBD).....	16
9. Requirements for acceptance and delivery of the project.....	17
10. History of changes.....	18

Light theater needs a program for building light shows based on wireless technologies for managing light sources. Supposed light sources are LED, LED with chip 2812, electroluminescent wire (elwire), electroluminescent panel (elpanel).

The program should fully satisfy the theater's needs in creating light shows based on the above technologies. Light show can consist of costumes (people, animals, robots, etc.), scenery (portals, trees, mechanisms, clouds, etc.), small and medium props (sword, feather, hammer, stick, etc)

The Application is designed for constructing scripts for light show theater. This Application has features for syncing these suits and effects with audio playback. Supported audio file formats: mp3, wav.

Since the tasks assigned to the program are quite extensive, it needs to be divided into several parts-modules: suite configurator (Configurator) module, module for creating and playing show (Show), module for creating light effects (Effects)

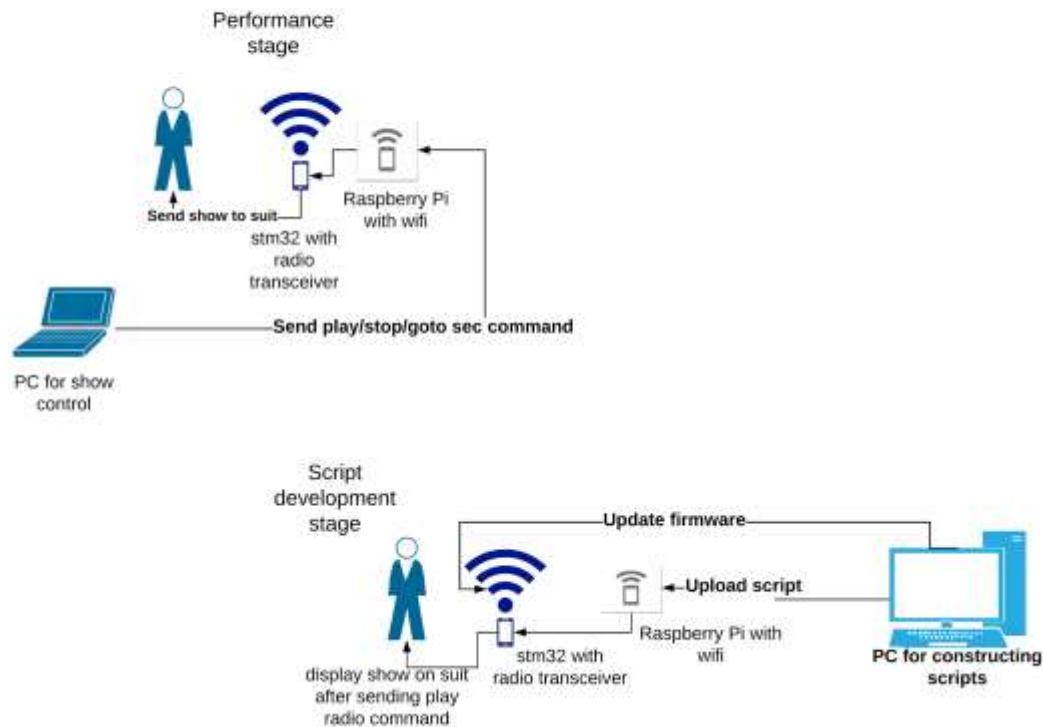
1. The goal:

- Complete implementation of Qt-based Desktop Application for light theater. The goal is to reimplement existing solution using QML as much as possible to make GUI more modern and user-friendly.
- This technical specification implies that only the desktop application needs to be implemented. All the interfaces with software and hardware should be described

2. System requirements:

- The Application should be able to run minimum on a 2 core CPU (Intel core i3 or i5 as an example), with not less than 3 gb of RAM with support of GPU memory not less than 512mb and OpenGL support for simulated effects rendering.

3. Overall System architecture



System in general consists of desktop application - Show constructor which runs on a PC, PC for show control - can be used also standard PC and also can be launched on Raspberry Pi or any other single board PC which connects to MCU board via UART and SPI that works as a bridge between radio transceiver module and PC and sends control commands for play, stop, pause, go to specific time mark. The hardware part is practically implemented. The file format of script will be attached to this Document.

4. Used Terminology and Abbreviations:

Term	Definition
App	Current Application that needs to be developed
QML	Qt Markup Language
WS2812	WS2812 Pixel Led Strip
LED	Light Emitting Diode
RGB	Red-Green-Blue
OS	Operating System

5. User interface requirements:

The current solution is implemented using Qt Widgets and lacks functionality and better separation between business logic and UI. We want to implement our Desktop App using QML as it is a better alternative for an existing variant. Also QML separates core logic written in C++ and UI layer which is good for future improvements.

It is preferred to implement all UI in a tab-based widget where every tab is responsible for its respective functionality. Tabs should be located on MainWindow. For now, we need to implement three main tabs inside main window: **Show Tab, Suit Editor Tab, Effect Editor Tab.**

User interface:

5.1 MainWindow

Main window is a window where all the functional tabs, docks, widgets and menus will be located. It should be scaled to the full desktop display (with ability to scale proportionally to full HD screen resolution and bigger). the minimum size of window must be non less than 320x240 with ability to hide main timeline during window minimizing using mouse. If QML supports dock widgets, it is recommended to implement four side dock widgets - for tree project structure (every track nodes has subnodes - effect marks), for file system structure, for effects library and for suits library (optional, may be changed to something another in the future).

5.2 Description of tabs:

5.2.1 Show Tab:

Approximate view of show tab is following:



Description: Main window is used to display the whole show. There should be displayed all project tracks - these tracks are related to every actor or requisite (for now only for actors) which is involved in a script.

What needs to be implemented in Main Show Tab:

5.2.2 Actor Track:



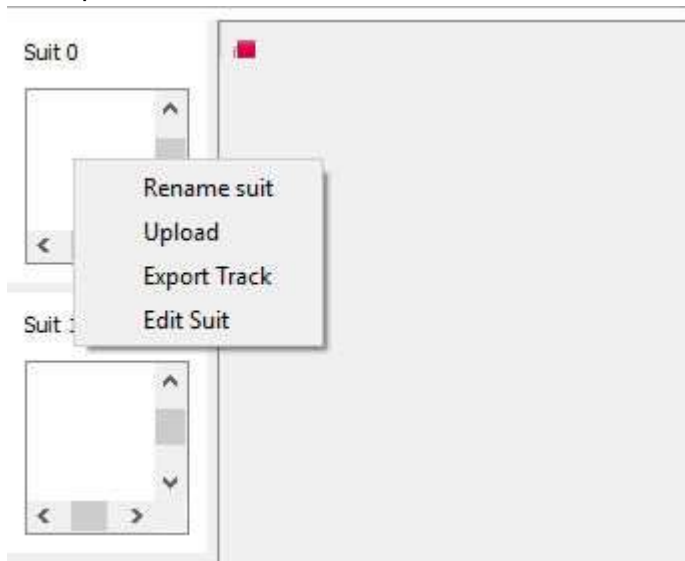
Actor Track Widget displays actor's thumbnail as a background image on canvas.



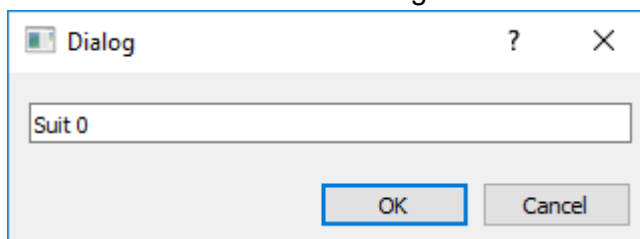
Every Actor (Performer) has segments related to his suit. Segments can be set on head, neck, hands, arms, torso, legs, and feet. Every segment is related to corresponding body part which can be set from suit editor. Segments on every body part can be divided into sub segments. You can set colors for every corresponding segment from suit editor. When playback marker comes to an effect on this segment, the segment in this thumbnail must trigger and highlight itself (can be implemented by changing alpha channel or repaint from black to selected color) probably with a signal-slot mechanics

Also every Actor Track Widget has a context menu which has options to edit current actor's suit or upload current scenario to actor's suit (all required options are implemented in a current app)

Example of suit context menu.

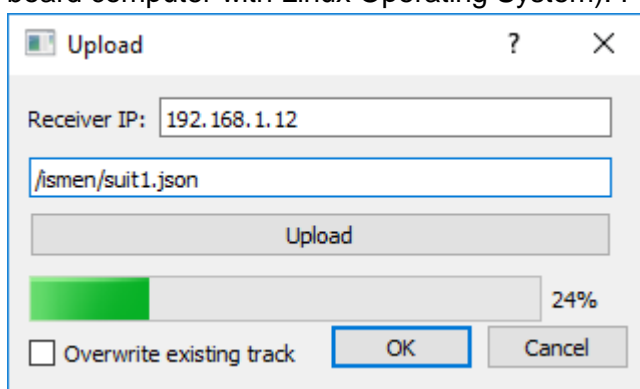


“**Rename suit**” shows the dialog where we can set another name to our suit.



Upload:

After clicking “**Upload**” user uploads project or separate suit file to a client (Client is a Single board computer with Linux Operating System). For simplicity we can use FTP client.

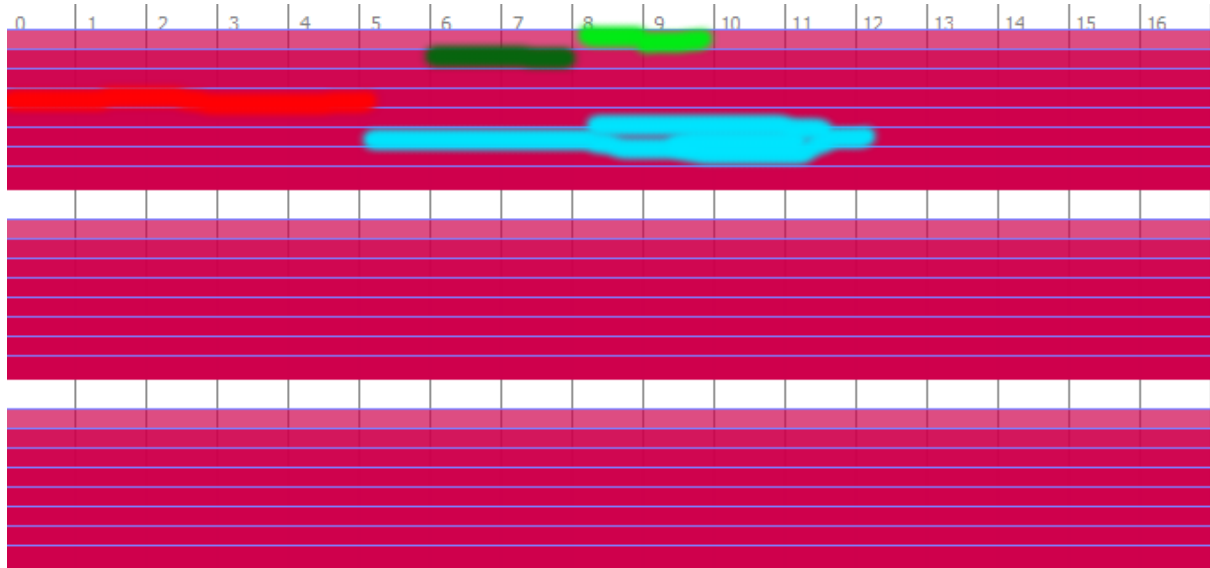


When we select “**Export Track**” we can export to json or binary data only selected suit and then upload it via ftp or any similar way to client.

After selecting “**Edit Suit**” Application switches to Suit Editor tab.

5.2.3 Timeline Widget:

A rough view of timeline is following



Timeline widget has time markers on every second. Time position has to be displayed in a format of mm:ss:mss (e.g 01:35:129) on the bottom dock of a main window.

On a track related to a corresponding actor, we can add effect markers. We can edit those markers by clicking right mouse button and switch to Effect Editor Tab.

Timeline widget should update its length if the user clicks somewhere at the end of a timeline viewport. Also the user can scroll viewport using scrollbars at the bottom of timeline.

The user can set a marker on a timeline with a mouse left click. Timeline should be scaled from 10% to 800% . Scaling redraws time markers and all effects appropriately. When user clicks on some effect on a timeline with right mouse button, call context menu is called where user can edit, copy, paste, duplicate or remove light effect.

5.2.4 Project file structure:

- File structure is JSON with metadata about all actors (names, number of segments, segment properties)

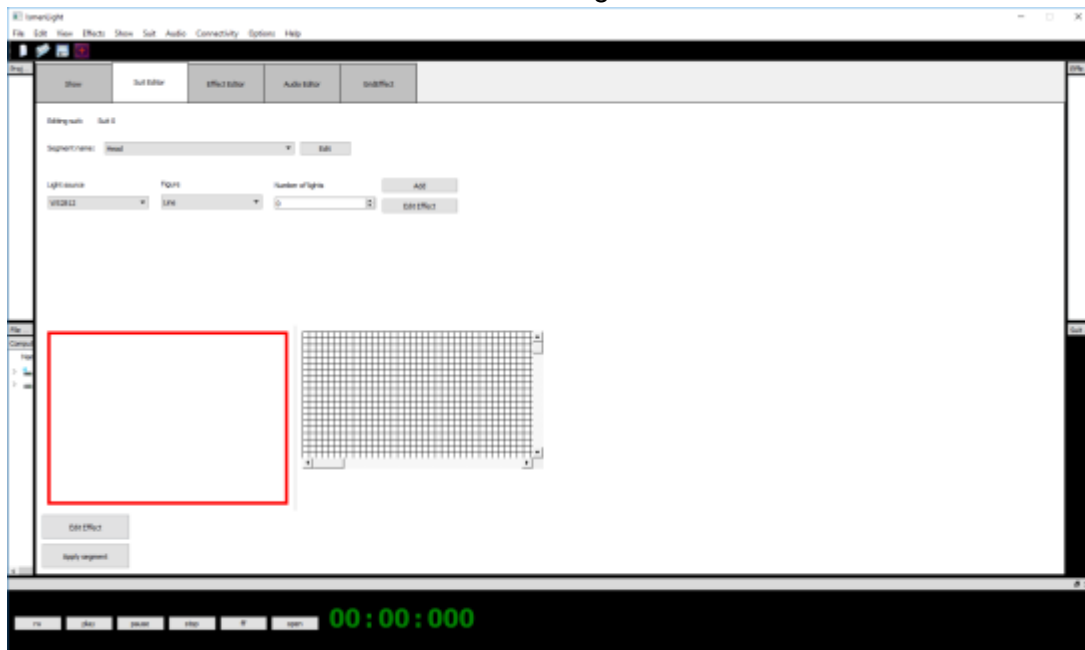
Approximate file structure is below (Proposals and optimizations are welcome!):

```
{  
    "project": "projname",  
    "duration": 20000,  
    "tracks": 4,  
    "trackdata" : [{  
        "trackname": "track1",  
        "effects": [{  
            "name": "effect1.fx",  
            "duration": 1000,  
            "start": 00:01:100,  
            "end": 00:02:100,  
            "data" :  
[255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,2  
55,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,  
255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,25  
5,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0,255,0,0]  
        },  
    ],  
    {"trackname": "track2",  
        "effects": [{  
            "name": "effect3.fx",  
            "duration": 1000,  
            "start": 1000,  
            "end": 2000  
        }],  
    ]},
```

Data is an array where our ws2812 data is displayed. It is an array, where RGB values for every pixel are set. In case if simple light emitting diodes or neon lights were selected - set all values to 255 or 1 (any non zero value in range of uint8_t) - those lights react only to on/off state.

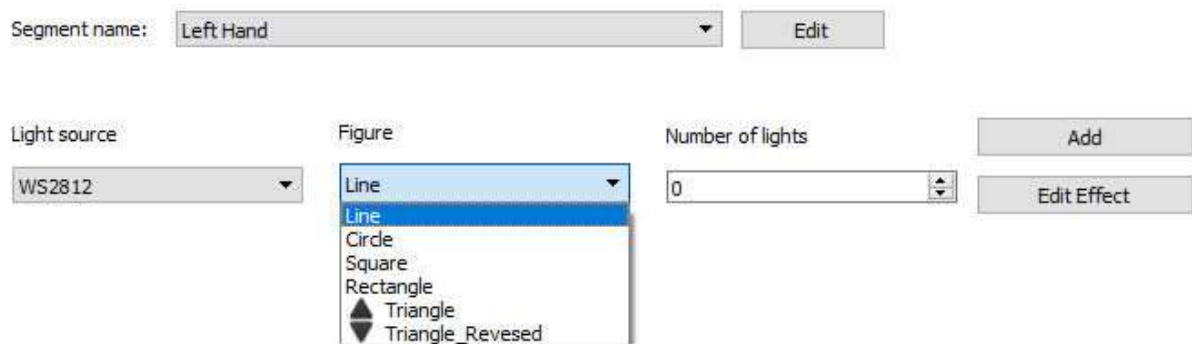
5.3 Suit Editor Tab

The current view of Suit Editor Tab is following



In this tab user can edit suit segments. If user has clicked “Edit suit” on some of the suits the app will pass to this tab. Select the light source from the general basket of all lights (for example: user needs to select segment on a left hand. User clicks on context menu “edit suit”, selects segment in “segment name” and in a dialog window selects new name of a segment. User always needs to know how many total light sources are left.

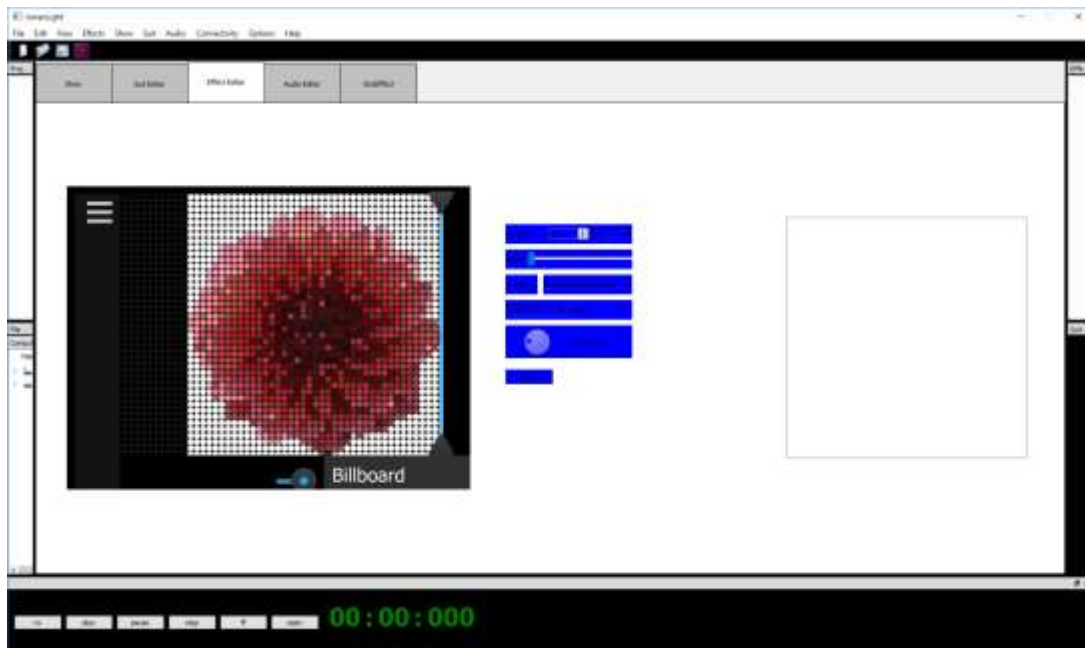
Menu for selecting lights on a segment approximately looks like following:



User can select light source from several types (WS2812, neon, RGB, Standard LEDs), select figure that will be folded from these lights (line, circle, square, rectangle, triangle, triangle reversed) and select number of lights. After user clicks “Add” he can select what leds from basket will be selected and added to a new suit segment. After user sets lights to segments, he can assign some effect to this segment with an “Edit Effect” button.

5.4 Effect Editor Tab

Effect Editor tab is a widget where the user can set different effects to selected segments. Current view of Effect Editor Tab. (View MUST be redesigned)



Effect Editor must have next scope of features:

- Edit current effects (For current moment we need several effects to emulate):
- Select effects from presets inside specific directory which is displayed on a effects dock widget (top right dock)

The next effects are related to WS2812 pixel LED and RGB-LED

Effects to Simulate:

- Random appear; Lights on segment are emitted randomly
- "Snake" - moving pixels in a form of snake
- Brightness - lights on segment are lighting up and out
- Billboard Effect - effect to pixelate image using OpenGL shader and render it using ws2812 leds. (additional functionality will be discussed separately).

All these effects have a corresponding window for selecting effect parameters:

- Random Appear (segment duration, segments to choose, select colors, select gradient on/off)
- "Snake effect" (segment duration, direction, checkbox for inverse direction, first LED, last LED) ;
- Brightness (duration of light increase and decrease, level of brightness from 1 to 100)
- Billboard Effect (the size of pixelation grid and region of selection) Resulting image will be stored in a separate data structure (Similar to bmp image file) to be passed to pixel LED matrix.

Every modeled effect should be simulated in a graphics window/widget. (The best way is to read the same data file that is will be sent to a suit board to reduce duplication of formats).

Approximate design of effect parameters window is here (Example of “Moving lights”) :

Duration

Speed

Direction

Brightness

Effects can also be saved as separate files and stored in json-like document with next structure:

[illegible]

Where “data” is actual rgb-values array and “duration” is a duration of an effect in ms

After the effect is designed, it can be added to our script to multiple suit segments and must be displayed on a timeline - the selected time position using mouse drag n drop

6. Script and audio playback:

Audio playback is synced with effects on timeline. Prepared audio track is uploaded to project using “open” button on the bottom of main window. The project starts to play when the button “play” is triggered.

If the user opens an audio file before we add any effects, timeline will update it's value to the duration of an audio track;

If the user opens one more audio track the app will ask the user if he wants to append to the end of the previous audio track, to put it in the beginning or to replace existing track. Later we will need an ability to insert new audio track inside existing audio track.

Project playback is done via reading the project file: reading “data” and “duration (start) and end” markers. As soon as time marker covers any effect on timeline, corresponding segment on suit should be highlighted on a thumbnail on Actor Track.

As soon as script is ended we can start play it again using play button or if the button repeat was clicked it will play again automatically.

7. User stories:

7.1 Create new project (scenario):

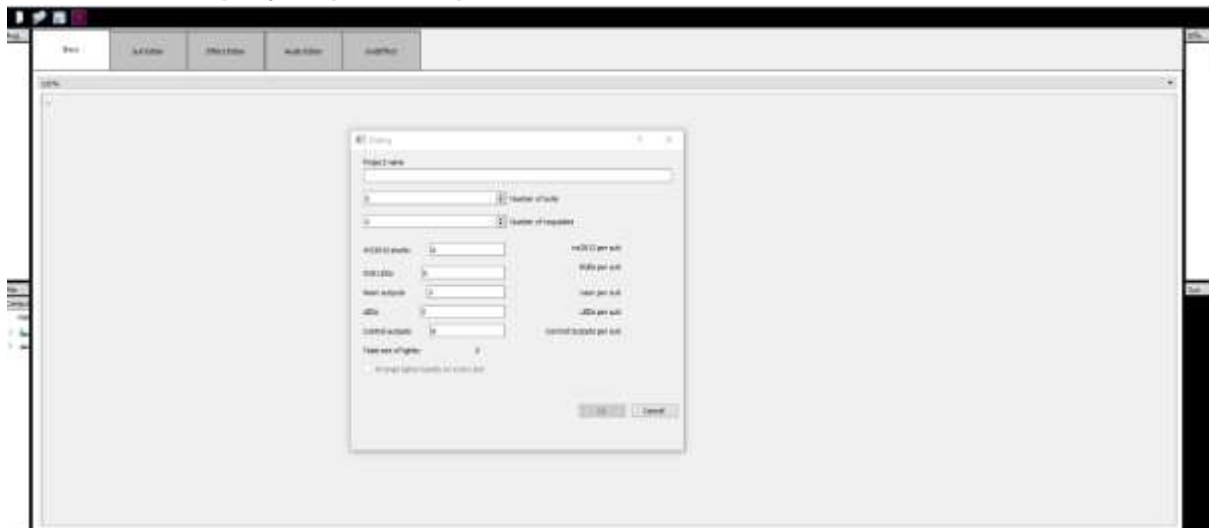
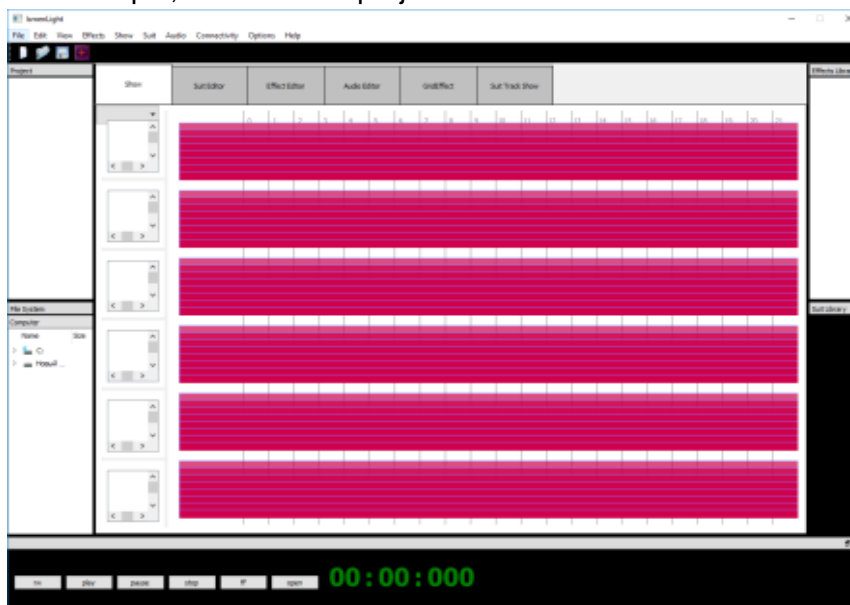


Image 7.1 New project dialog window

To create new scenario, we need to set name to a project, set the number of actors. Set the number of all light sources that will be used in project (RGB-leds, WS2812-leds, Neon Lights, regular LEDs) Also we need to add an ability to divide the number of all light sources between all actors using a checkbox.

After we created a project all our tracks are rendered on main show tab with start number of segments equal to 8 (head, left hand, right hand, left arm, right arm, torso, left leg, right leg). These segments can be subdivided (e.g. we can have 2 segments on left hand, or three on torso, etc). Also we need to have an ability to selected and “mute” some segments or to make some selected segments “solo” with corresponding buttons near the segment track. For example, user created project with 6 actors.



Then if we want to modify some segment on track no 1, we will trigger “Edit Suit” on track no 1 and go to segment editor. Here we will select what segment we need to modify, select light

source, add the number of lights to a segment and then will move to effect editor to apply an effect from library or some custom effect to this segment.

7.2 Upload project to suit boards.

After project was created it must also be saved into temporary folder in case of Application crash to restore the last successful file. To upload project to suit board, the board should be connected to the same subnetwork. The easiest way is to use ftp client for transferring to ftp server on board

8.Possible features that will be implemented later:

- Implement trial or guest mode where user can only use this Application to play show and scenarios but not to modify them.

TBD

9. Requirements for acceptance and delivery of the project:

The Contractor shall provide the following set of delivery upon delivery of the project:

- Terms of Reference
- System Source Code
- Executable System Modules
- Test scenarios
- User documentation

The Customer and the Contractor document the progress of the acceptance tests in the Test Report.

Based on the Testing Protocol, the Contractor in cooperation with the Customer signs the Acceptance Certificate.

Acceptance (UI, functional) tests for software should be conducted at each stage.

10. History of changes

Date	Version	Author	Changes
14.06.2018	0.1	Ivan Petrivskyi	Beginning of the document