Requirements and Analysis Document for HueStew

Version: 0.1.33.7
Date: 2015-04-10
Authors: Adam Andreasson, Daniel Illipe, Marcus Randevik, Patrik Olson

1 Introduction

This section gives a brief overview of the project.

1.1 Purpose of application

This project aims to provide a workspace where the user can produce and play light shows involving an arbitrary number of “smart”, network connected lamps. Each show is linked to a specific song.

1.2 General characteristics of application

The application will be a single user desktop application with a graphical user interface for Windows, Mac and Linux.

The application will resemble other creative softwares such as Adobe Photoshop and Sony Vegas. As such, the interface will upon startup be mostly empty with areas that can gradually be filled by the user. There will be no limit as to how vast a project can be. In order to support great productivity, controls and tools should be easily accessible through accelerators as well as physical manipulation.

1.3 Scope of application

The application does not include the ability to, given a song, automatically create a show whereas the key frames match the intensity of the song. Shows will only be able to be stored locally as the application should not feature any integration with cloud storage.

The basic application also does not control any hardware lamps, which is instead controlled by additional plugins. A plugin for the Philips Hue system is included along with the HueStew Studio software.

1.4 Objectives and success criteria of the project

- It should be possible to import a song and create a show that is synchronized with the song.
- The application should be able to import a file with an already created show and play it.
- When playing a show, the application should be able to control Philips Hue lights. (Note: this is not a part of the main software, see 1.3)

1.5 Definitions, acronyms and abbreviations

- Java: Platform independent programming language
- JRE: The Java Runtime Environment, software needed to run a Java application.
- Key frame: Event that triggers a lamp transition (change brightness, color etc...).
- Light track: Track where key frames associated with one or several lamps are presented in their chronological order.
- Show: A song and a collection of one or several light tracks.
- Track cursor: Vertical line that corresponds to the elapsed time in the show.

2. Requirements

2.1 Functional requirements

The user should be able to;

1. Create a show
2. Add multiple light tracks
3. Add key frames to a track. A key frame can
   a) turn on or off the lamp
   b) change the color, hue and brightness of a lamp
4. Play the show, either by
   a) Simulating lamps on the screen.
   b) Communicating with network connected lamps.
5. Save the show as a file that can be later opened and played
6. Exit the application. Will prompt the user to save.

2.2 Non-functional requirements

2.2.1 Usability

Since this application is intended for non-professional light show creators, the interface should not be too advanced since it could scare away new users while at the same time supporting the creative flow by having all the necessary tool nearby. Physical manipulation of data is to be prefered as it is more natural to humans.

As this application is intended to be solely used on a desktop or a laptop, the interface won't be designed for use with small screens and low resolutions.

2.2.2 Reliability

The application should not permit the user to do action which are not legal, such as having several key frames in a light track at the same timestamp or to set the brightness to over 100%. Data saved by the user should always remain accessible at a later time and not be corrupt. The application should not crash.

2.2.3 Performance

Any actions performed by the user should not exceed a 1 second response time in worst case. The length and size of a show should not impact the performance of the application.

2.2.4 Supportability

The data representation should be designed in a way so that it later could be used with an independent player.

2.2.5 Implementation

To achieve platform independence the application will use the Java environment. All hosts must have the JRE installed and configured.

2.2.6 Packaging and installation

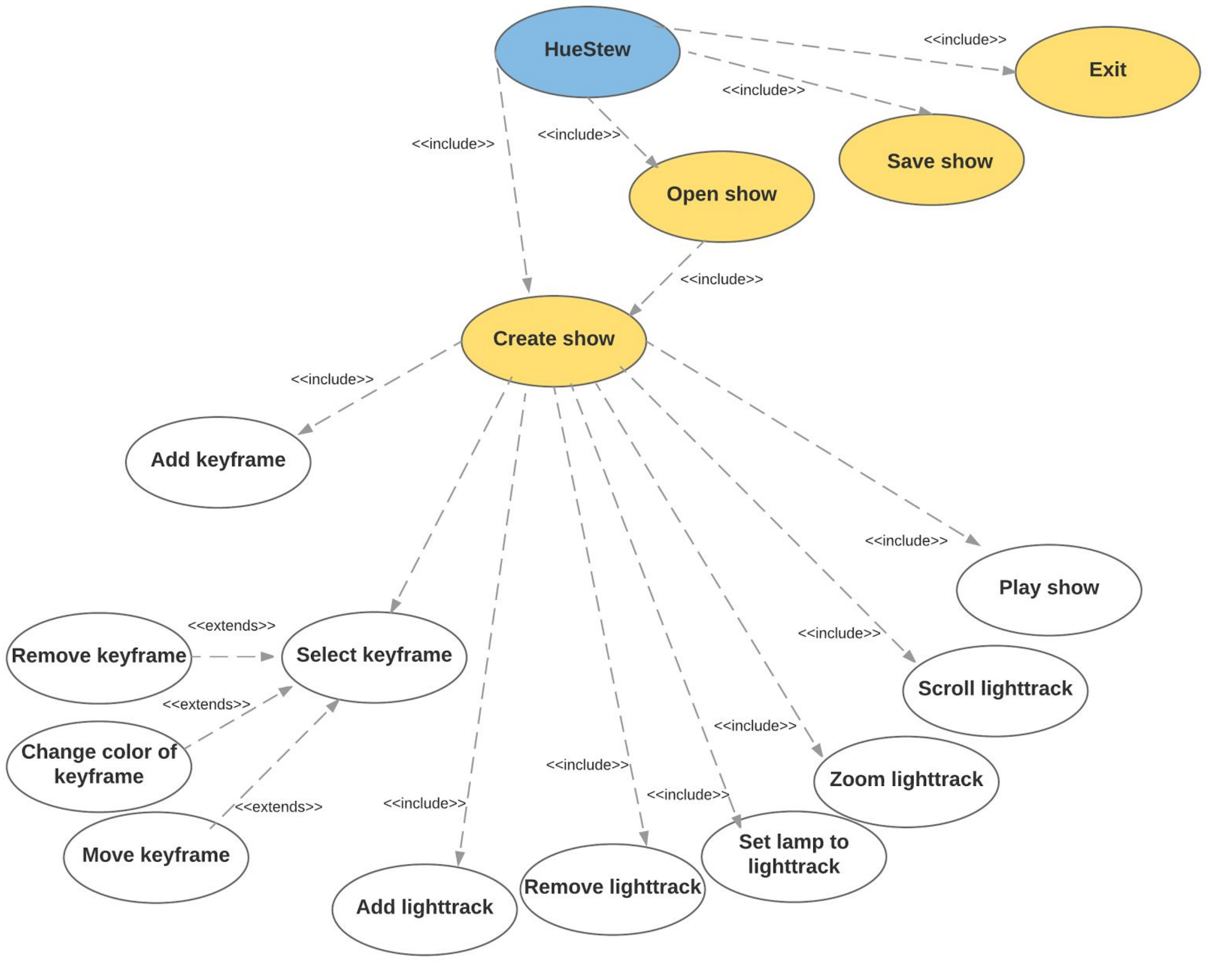
The application will be delivered as a stand-alone, runnable Java jar file.

2.2.7 Legal

The lamps and the API used to control them is the property of their respective company.

2.3 Application models

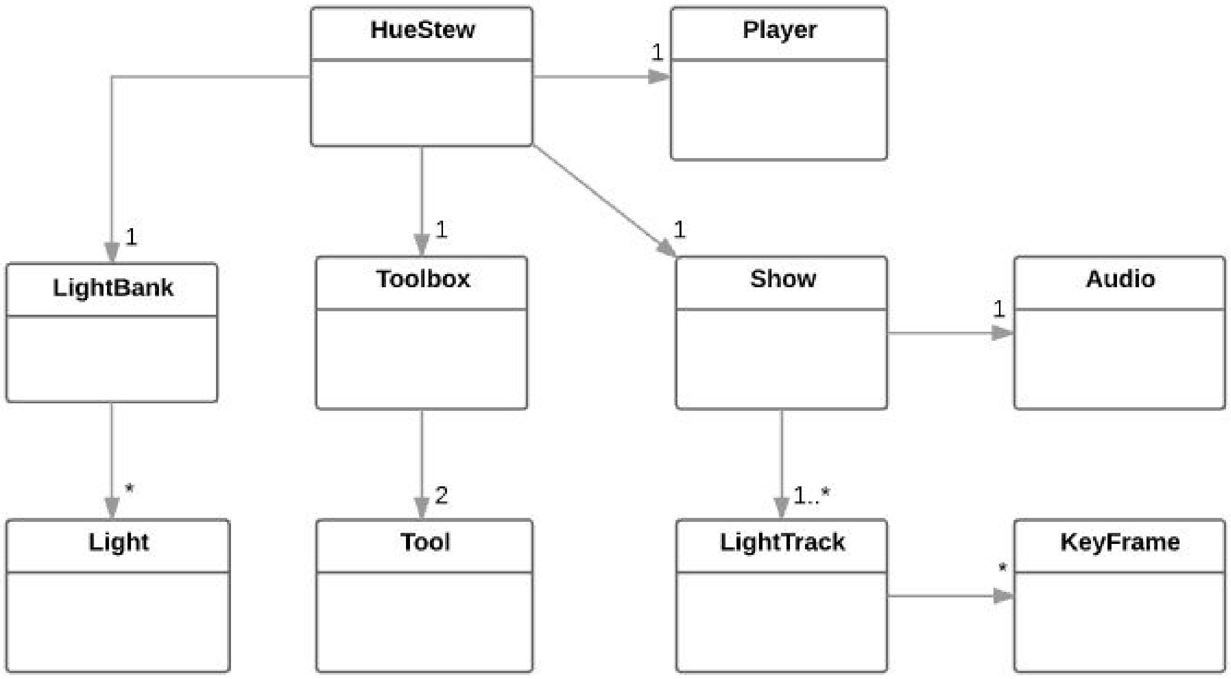
2.3.1 Use case model



2.3.2 Use cases priority

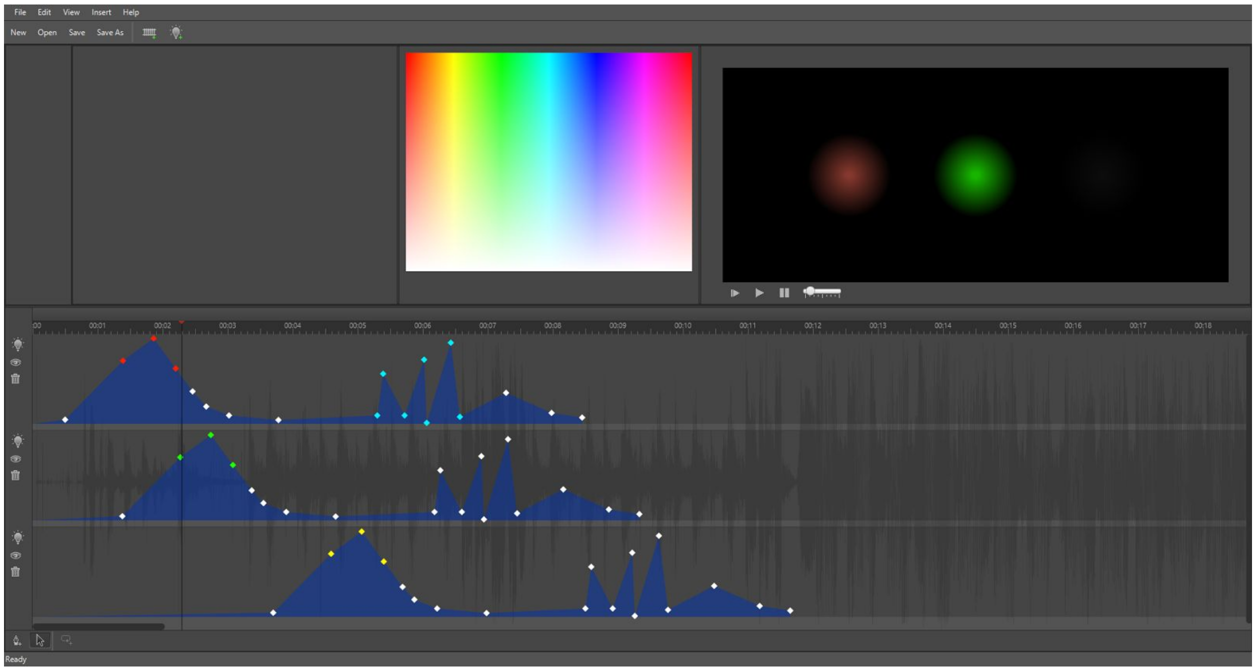
1. Add key frame
2. Select key frame
3. Change color of key frame
4. Move key frame
5. Set lamp to light track

2.3.3 Domain model



2.3.4 User interface

The application has a resizable GUI and follows standard conventions.



2.4 References

NA

APPENDIX

Use case texts

Add key frame

Summary:	Add a key frame to a light track.
Priority:	High
Extends:	-
Includes:	-
Participators:	The actual user

Normal flow of events:

User adds a key frame

	Actor	System
1	Selects the populator tool in the toolbar below the trackview.	
2		The button representing the tool in the toolbar is marked as active.
3		The cursor changes to a pen when it is hovering over the light tracks
4	Left click on one of the light tracks	
5		A diamond shape representing a key frame is added to the light track at the clicked position

Alternate flow:

User tried to add a key frame at a timestamp already used by another key frame.

	Actor	**System**
4	Left click on one of the light tracks at a timestamp where there already exists a key frame.	
5		No diamond shape is added to the light track.

Alternate flow:

User tried to add a key frame at the exact position of another key frame.

	Actor	**System**
4	Left click on one another key frame	
5		The already existing key frame is marked as selected

Select key frame

Summary:	Select one or several key frames to be manipulated in some way.
Priority:	High
Extends:	-
Includes:	-
Participators:	The actual user

Normal flow of events:

User selects one key frame.

	Actor	**System**
1	Selects the selector tool in the toolbar below the trackview.	
2		The button representing the tool in the toolbar is marked as active.
3		The cursor changes to the default cursor when it is hovering over the light tracks
4	Left clicks on one of the key frames.	
5		The clicked diamond shape representing a key frame is outlined.

Alternate flow:

User selects several key frames by holding ctrl while left clicking on key frames.

	Actor	**System**
4	Left clicks on several of the key frames.	
5		The clicked diamond shapes representing key frames are outlined.

Alternate flow:

User selects several key frames by holding down the left mouse button and drawing a selector box around key frames.

	Actor	**System**
4	Left clicks and holds the mouse down while moving it in some direction.	
5		A blue rectangle with corners in the initial position and the current position of the cursor is drawn.
6	Releases the left mouse button	
7		The key frames within the blue rectangle are now marked as selected (have outline).

Change color of key frame

Summary:	The user changes the color of one or several key frames.
Priority:	High
Extends:	Select key frame
Includes:	
Participators:	The actual user

Normal flow of events:

User changes the color of one key frame.

	Actor	**System**
1		User has already selected a key frame(s) which now has an outline (see UC: Select key frame) The key frame diamond shape is also displayed in the color picker frame above the tracks.
2	Clicks on a specific point in the color picker	
3		The indicator of the color changes to the position that was clicked and the key frame(s) has its color changed accordingly.

Move key frame

Summary:	Moves one or several key frames to a different timestamp.
Priority:	High
Extends:	Select key frame
Includes:	
Participators:	The actual user

Normal flow of events:

The user moves on key frame

	Actor	System
1		User has already selected one or several key frames (see UC: Select key frame)
2	Left-clicks one of the selected key frames and drags the key frames to their intended timestamp	
3		Changes the position of the key frames.

Alternate flow of events:

The user tries to move a key frame past another key frame.

	Actor	System
2	Left-clicks and drags the key frames past a non-selected key frame.	
3		Changes the position of the key frames until they reach another key frame. All key frames will “hit” a wall and movement in the same horizontal direction will not be possible.

Set lamp to light track

Summary:	Set a lamp to a light track so that it follows it.
Priority:	Medium
Extends:	-
Includes:	-
Participators:	The actual user

Normal flow of events:

User clicks on the lamp button and checks one of the checkboxes.

	Actor	**System**
1	Left click on the lamp button to the left of the light track to be followed.	
2		Shows a small panel with all the lights connected and used in the program. Light that are not yet connected to a light track will be shown in white. Lights already associated with the light track are checked.
3	Left click in one of the checkboxes	
4		A checkmark is placed in the checkbox and the light is now connected to this light track.