VOTSH

Architecture and Requirements

Waves Project

April 3, 2014

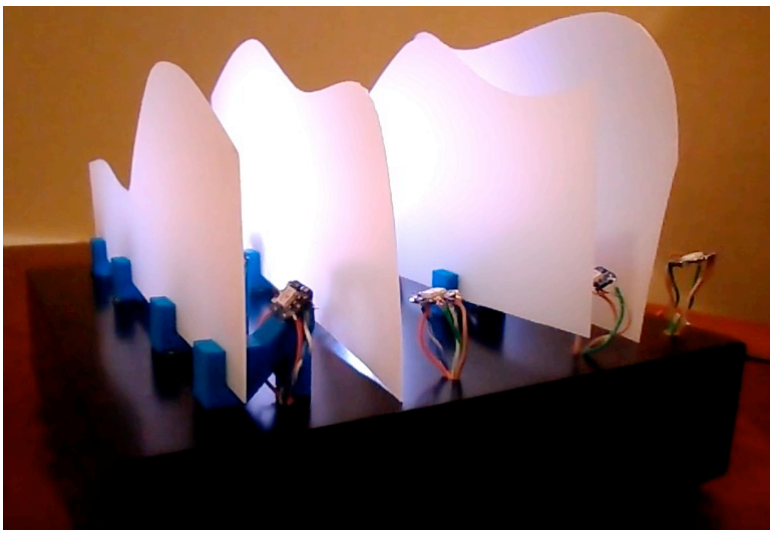
Waves is a table top wireless computing platform that uses color, light, and animation to entertain and bring meaning to people's lives. Waves talk to one another over wireless networks to organize and present detailed multi-location shows. This document defines the architecture, show schema, and Web interface to create and edit Waves shows and events.

This document uses the following terminology:

1. **Grid** – Sprites and Background exist within a large 3 dimensional space called "the grid". People building shows identify regions within the grid that map to the location of Wave units.
2. **Sprite** – A sprite is a rich media element placed on a grid of pixels that span across one or more regions. A region is an area that maps to a Wave unit. Sprites may be colored objects (square, circle, line), audio recording, video recording, and animated cell.
3. **Background** – a Specialized Sprite that is the default media for the grid.
4. **Cue** – An instruction to one or more sprites on how to animate, move, and sound within the grid of Wave unit regions. An audio cue is music or sound effects.
5. **Sequence** – A grouping of cues.
6. **Show** – A grouping of cues and sequences.

Wave Machines

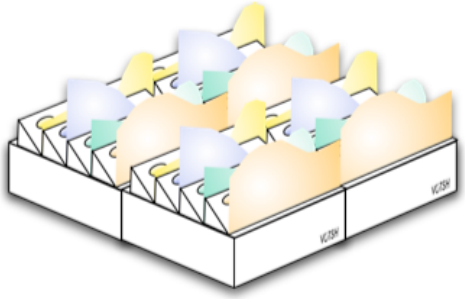
Votsh Waves prototype is a 12 x 12 x 3 inch box. It sits on a table. It uses LEDs and diffusion filter (the Chillovan effect) and has a speaker inside the box. The filter sits on top of the box. View a video of the prototype at https://vimeo.com/86288970



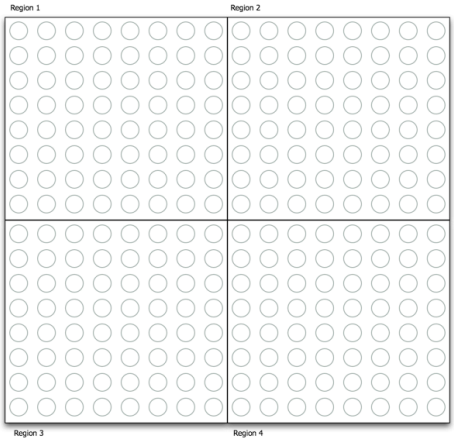
Sprites and Background exist within a large 3 dimensional space name the grid. A pixel is an abstract measure of space within the grid. Within the grid are 3 dimensional regions, where each Wave unit maps to a unique region. Map regions to the grid using the Construction page.

Pixels within each Wave unit region map to the Wave unit devices, including lights, speakers, video projectors, lasers, fire emitters, and smoke emitters.

For example, 4 Waves units positioned in a square grid:

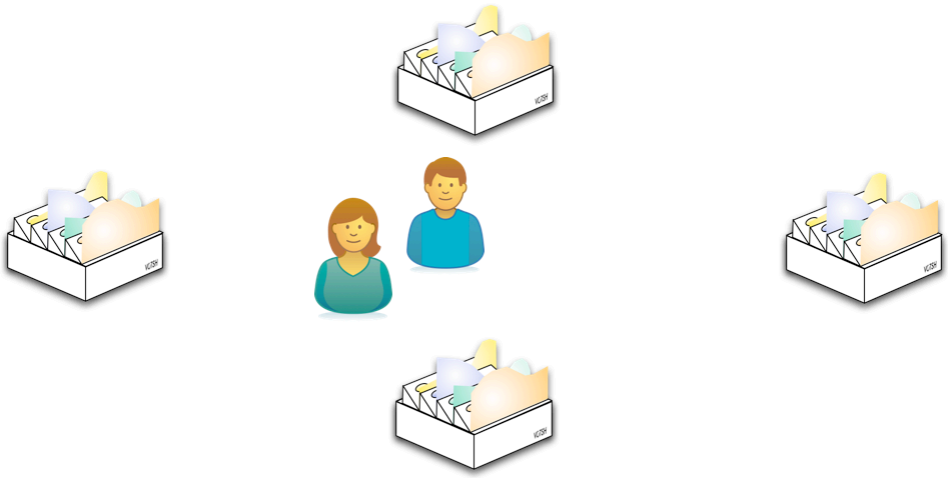


The region map for the above 4 units:

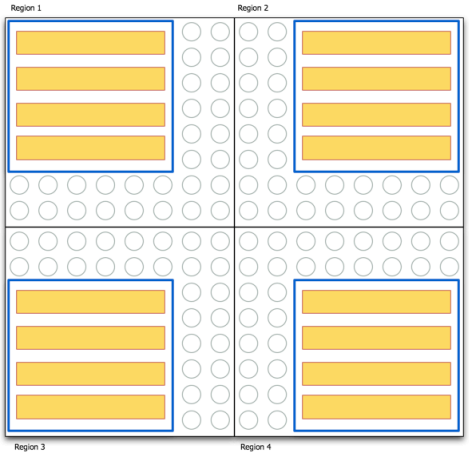


Regions do not need to be square. Wave OS maps the relationship to the device to provide smooth unpixelated animation.

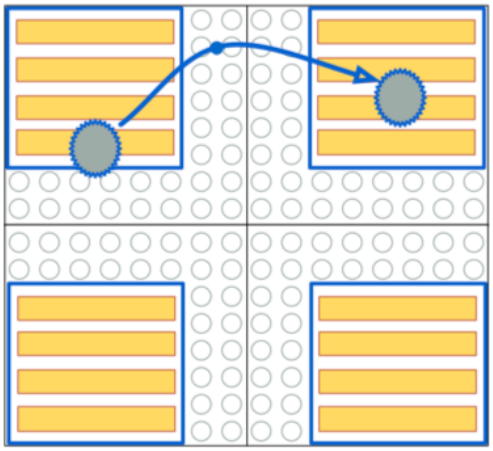
For example, when 4 Waves units are positioned around people:



The grid is:



Regions allow cues of sprites to move from one Wave box region to another smoothly, including audio music and sound effects.

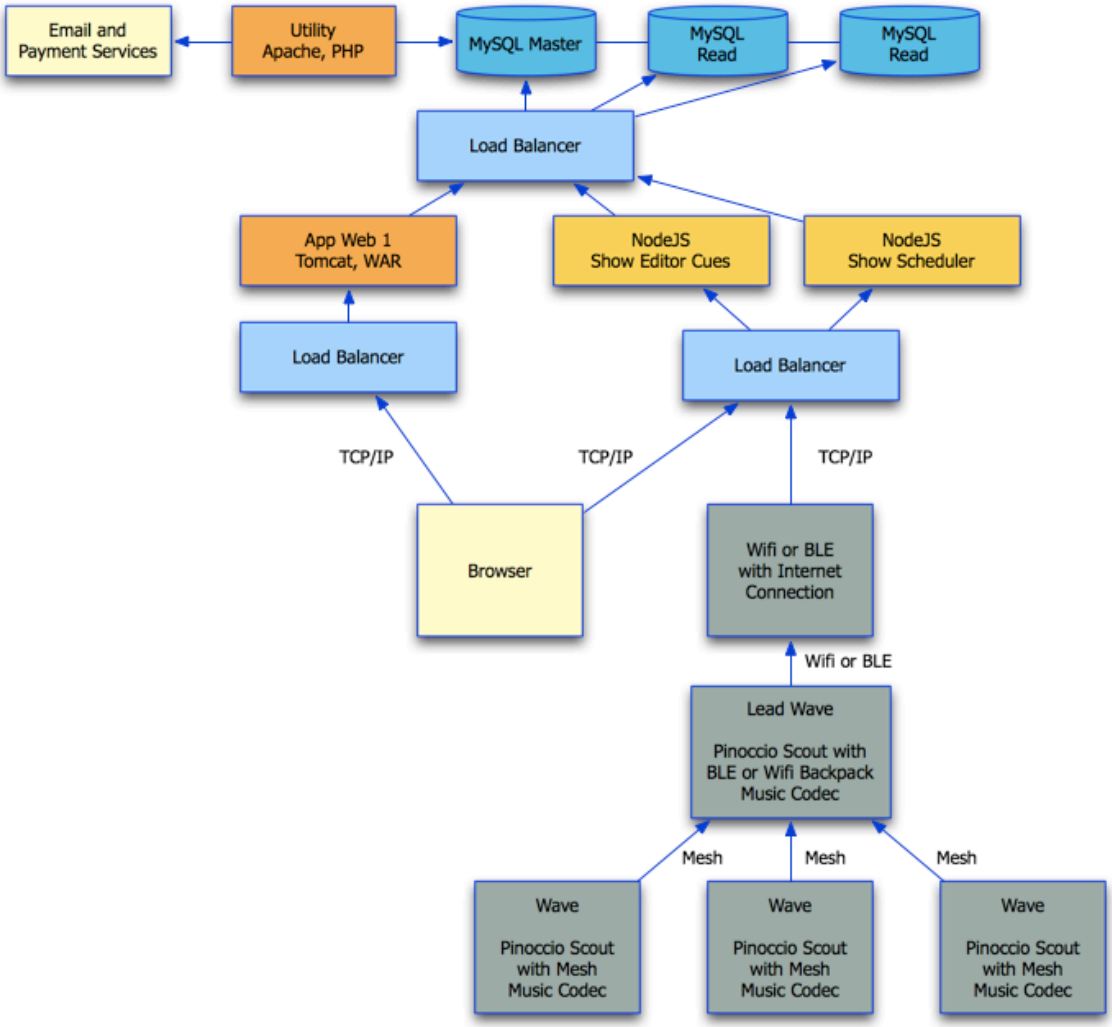


Architecture

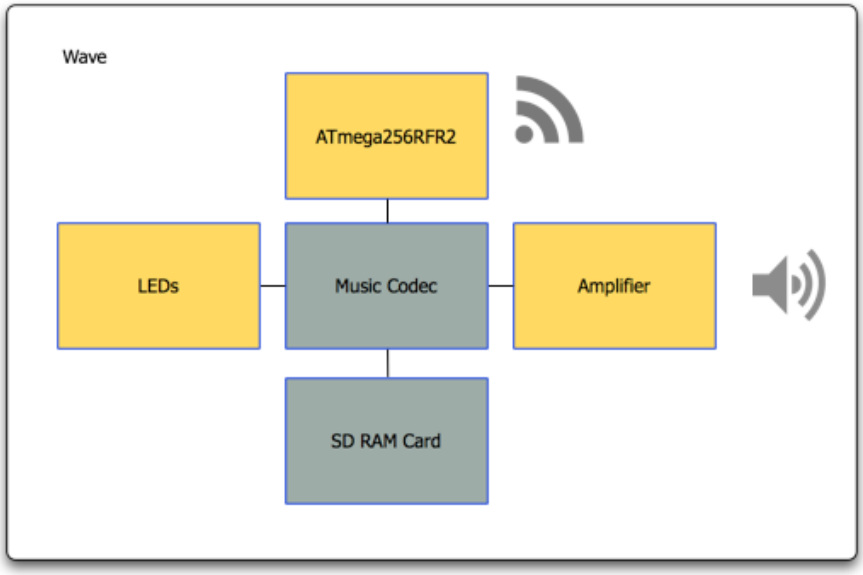
Waves implements a service oriented architecture to service 2 consumers: People using browsers and Wave units presenting shows. People use a browser to configure, schedule, and build shows. Wave units receive shows and schedules from Node.js servers.

People use browsers that connect over a TCP/IP network connection to the Waves Load Balancers. The Load Balancers and services run on servers on the public Internet or on grid-based servers inside an organization's datacenter.

Wave units communicate over Atmel 802.15.4 radios using Light Weight Mesh (LWM, Details at http://www.atmel.com/tools/LIGHTWEIGHT_MESH.aspx). A Wave unit with a Wifi or Bluetooth (BLE) backpack defines a Lead Wave. The Lead Wave operates portions of shows and is a gateway to the Wave units. The Lead Wave is a transformation engine to pre-parse and prepare elements of a show for individual distribution to each Wave.



Lead Wave and Wave units implement the following hardware design:



Lead Wave units have a Wifi or BLE backpack.

Sprite Definition

The core object definition for everything that runs in Waves is a Sprite. Sprites have the following elements:

- **Start Location** – X, Y, Z coordinates for the starting location of a Sprite. Coordinates define 1 pixel within a 3 dimensional space – named the grid. A pixel is an abstract measure of space within a large 3 dimensional space. Within the grid are 3 dimensional regions, where each Wave unit maps to a unique region. Pixels within each Wave unit region map to the Wave units light(s), speaker(s), video projector(s), fire emitter(s), smoke emitters(s), blower(s), and vent(s). Regions enable a Sprite to begin on one Wave unit's mapped pixels and end on another Wave unit's mapped pixels.
- **End Location** – X, Y, Z coordinates for the end location of a Sprite.
- **Dimensions** – Width, Height, Depth coordinates for the Sprite definition. Sprites occupy space within the 3 dimensional grid.
- **Scale** – Value from 1 to 500 defining the linear scale of the Sprite definition within the grid.
- **Speed** – a percentage value from 0 to 99 to cross the grid at 1 pixel per second.
- **Compass** – an X, Y, Z coordinate or other Sprite location for the Sprite to automatically point towards within the grid.
- **Spin** – a percentage value from 0 to 99 to rotate the sprite 360 degress in a 10 second period. The second part of the spin value determines direction of the spin around 3 dimensional coordinates.
- **Coriolis Spin** – a percentage value from 0 to 99 to rotate the sprite perpendicular to the direction of motion and to the axis of rotation.
- **Burst** – a value to indicate an outward animation of the sprite. Similar to fireworks.

Sprite Types

Sprites come in the following types:

- **Light** – a circular pool of color.
- **Image** – a static JPEG or PNG image
- **Video** – an MPEG 3 encoded media file
- **Type** – using standard TrueType fonts. Type sprite defines the font, size, and color.
- **Shape** – triangle, square, circle, line. Shape sprite types also define size and fill pattern.
- **Music** – OGG file
- **Sound Effects** – OGG files pre-installed in Wave units.
- **Cell Animation** – cell based animation, including image rotation, speed, and repeat values
- **Shape** – an STL-based 3 dimensional object definition
- **Projection Onto Shape** – Image mapped to a Shape in 3 dimensions.

Wave Show Runtime

The Wave Show Runtime is responsible for operating shows in the Wave unit and region. It consists of the following components:

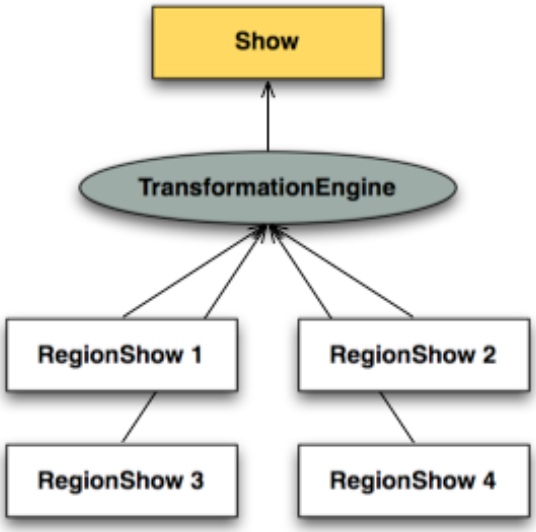
- **Show Manager** – polls the Show Scheduler (implemented in Node.js in the backend server) to identify a Show to run immediately.
- **Lead Wave Transformation Engine** – Downloads the Show from the Show Scheduler and creates Shows for each region.
- **Sprint Render Engine** – Downloads the region show from the Transformation Engine.
- **Snapshot Protocol** – Set of RESTful protocols for communication between the Wave Show Runtime components.

Lead Wave Transformation Engine

Transformation Engine is software code that resides in the Lead Wave. It removes the need for every region to hold the entire Show. TransformationEngine is written in C++ with a target Atmega256RFR2 processor.

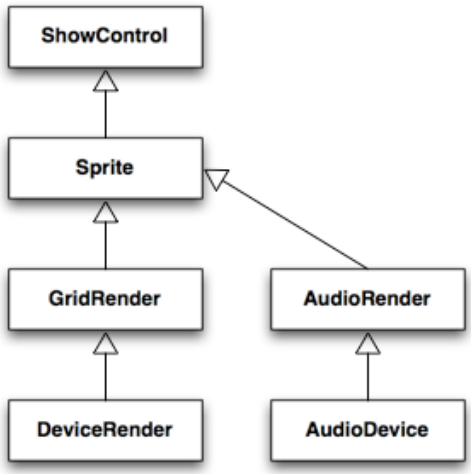
Transformation Engine receives a Show and renders Show files specific to each region in the grid. For example, a cue that moves an image from region 1 to region 2 appears in the RegionShow for region 1 for only the time slice the image is actually displayed in the region 1, the same for region 2.

The same is true for audio sprites. Only the sound that will be produced on the speaker in region 1 is in the Show for region 1.



Sprite Render Engine

Each Wave unit – including the Lead Wave – has the Sprint Render Engine.



- ShowControl – creates Sprite objects for each sprite in the Show. It later deletes Sprite objects when they are done.
- Sprite – creates a GridRender, AudioRender or other render engine object. Sprite processes movement within the region.
- DeviceRender – renders the Sprite from the grid to a RGB LED unit (WS2812 unit) using the Adafruit NeoPixel device library. Details at http://learn.adafruit.com/adafruit-neopixel-uberguide/overview
- AudioRender – plays an OGG audio file on the Wave audio codec unit. Votsh is considering using the VS1053 audio codec at the time of writing this document.

Snapshot Protocol

Provides these services:

- Provides status of a component to upstream components (for example, the Transformation Engine is upstream of the Sprite Render)
- Provides A Transfer Protocol (ATP) to move large files across the mesh network. See definition at https://github.com/Votsh/waves/tree/master/libraries/ATransferProtocol2
- Time service – identifies the time since the epoch to milliseconds. This is a long long value in C++.

Wave Show Definition

Waves uses Javascript Object Notation (JSON) to define shows, cues, schedules and other definitions. The following defines the show schema:

Show Definition

```
{
        "name": "Sinatra Duets Show",
        "favorite": "yes",
        "rating": "3",
        "public": "yes",
        "image": "rome.jpg",

        "entitlement": {}
        "grid":{}
        "schedule": {}
        "sprite": {}
}
```

Entitlement

```
{
        "entitlement": {
                "person": "fcohen",
                "enabled": "yes",
                "start": "38172827",
                "end": " 38179818",
                "email": "fcohen@votsh.com",
                "autorenew": "yes",
                "cc-account": "98478"
        }
}
```

Grid

```
{
        "grid":{
                "size": {
                        "rectangle": {
                                "left": "0",
                                "top": "0",
                                "right": "10000",
                                "bottom": "10000"
                        },
                }
                "region":{
                        "id": "1834-1272-4828-8171",
                        "name":, "tv room 1",
                        "type": "rectangle",
                        "rectangle": {
                                "left": "0",
                                "top": "0",
```

```
                    "right": "1000",
                    "bottom": "1000"
                }
            }
            "region":{
                "id": "1834-1272-4828-3811",
                "name":, "tv room 2",
                "type": "rectangle",
                "rectangle": {
                    "left": "2000",
                    "top": "0",
                    "right": "3000",
                    "bottom": "3000"
                }
            }
            "region":{
                "id": "1834-1272-4828-4716",
                "name":, "tv room 3",
                "type": "rectangle",
                "rectangle": {
                    "left": "0",
                    "top": "2000",
                    "right": "1000",
                    "bottom": "3000"
                }
            }
            "region":{
                "id": "1834-1272-4828-3174",
                "name":, "tv room 4",
                "type": "rectangle",
                "rectangle": {
                    "left": "2000",
                    "top": "2000",
                    "right": "3000",
                    "bottom": "3000"
                }
            }

        }
}
```

Schedule


```
{
        "schedule":{
                "name": "Sinatra Duets",
                "enabled": "yes",
                "start": "38176700",
                "end": " 38176900",
                "every": {
                        "period": "minutes",
                        "value": "10"
                },
                "once": "38176700",
                "musicAvailable": "yes",
                "stopForAnotherShow": "yes",
                "pauseForAnotherShow": "yes",
                "onSocialChange": {
                        "service": "FourSquare",
                        "person": "frankcohen"
                }
        }
}
```

Sprite

```
{
        "sprite": {
                "name": "Child Entry 1",
                "startlocation": {
                        "point": {
                                "x": "0",
                                "y": "0",
                                "z": "0"
                        }
                },
                "endlocation": {
                        "point": {
                                "x": "1000",
                                "y": "1000",
                                "z": "0"
                        }
                },
                "dimensions": {
                        "start": {
                                "x": "0",
                                "y": "0",
                                "z": "0"
                        }
                        "end": {
                                "x": "100",
                                "y": "100",
                                "z": "0"
                        },
                "scale": "1",
                "speed": "10",
                "compass": {
                        "pointingAt", "sprite",
                        "spriteName", "Child Entry 2"
                },
                "spin", "0",
                "coriolisSpin", "0",
                "burst", "0",
                "type", {
                        "name", "image",

                        type specific values go here, see below
                }


        }
}
```

Light

```
{
        "color", "#6cc6ff",
        "size", "100"
}
```

Video

```
{
        "file", "KidsAtPlay.mp3",
}
```

CellAnimation

```
{
        "file", "Shere.apng",
        "speed", "100"
}
```

Shape

```
{
        "shape", "Star",
        "color", "#6cc6ff",
        "size", "100"
}
```

Title

```
{
        "file", "Sinatra.png",
        "text", "Sinatra",
        "size", "100"
}
```

Music

```
{
        "file", "MyLove.ogg",
        "volume", "50",
        "service", "Pandora",
        "leftChannel", "off",
        "rightChannel", "off",
        "autoChannel", "on",
}
```

SoundEffect

```
{
        "effect", "Crunch",
        "volume", "50"
}
```

Projection

```
{
        "file", "Flat-topCone.stl",
        "shape", "ImageWithdots.jpg",
        "color", "#6cc6ff",
        "volume", "50"
}
```

STL

```
{
        "file", "Flat-topCone.stl",
        "color", "#6cc6ff",
        "size", "50"
}
```

Technology

Client-side User Interface Technologies:

Ajax page interfaces implemented using http://script.aculo.us

STL Shape type use in Waves implemented using Thingiview.js, http://n0r.org/thingiview.js/examples/index.html for viewing STL files.

Waves implements the Timeline using:

http://timeglider.com/widget/kitchen_sink.html or

http://almende.github.io/chap-links-library/timeline.html

Cloud Hosting:

Amazon Web Services (AWS), Elastic Computing (EC2)

Firewall:

AWS Elastic Firewall

Load Balancer:

http://aws.amazon.com/elasticloadbalancing/

Storage:

AWS Attached Storage

Database:

MySQL

Web server:

Apache Tomcat 7

Server-side framework:

Tomcat and Node.js

Testing libraries:

Appvance PerformanceCloud, Junit, Designer Script

Version control:

GITHub

Project Management:

Basecamp

Credit Card Processing:

Chargify.com + Votsh merchant account

Deliverables

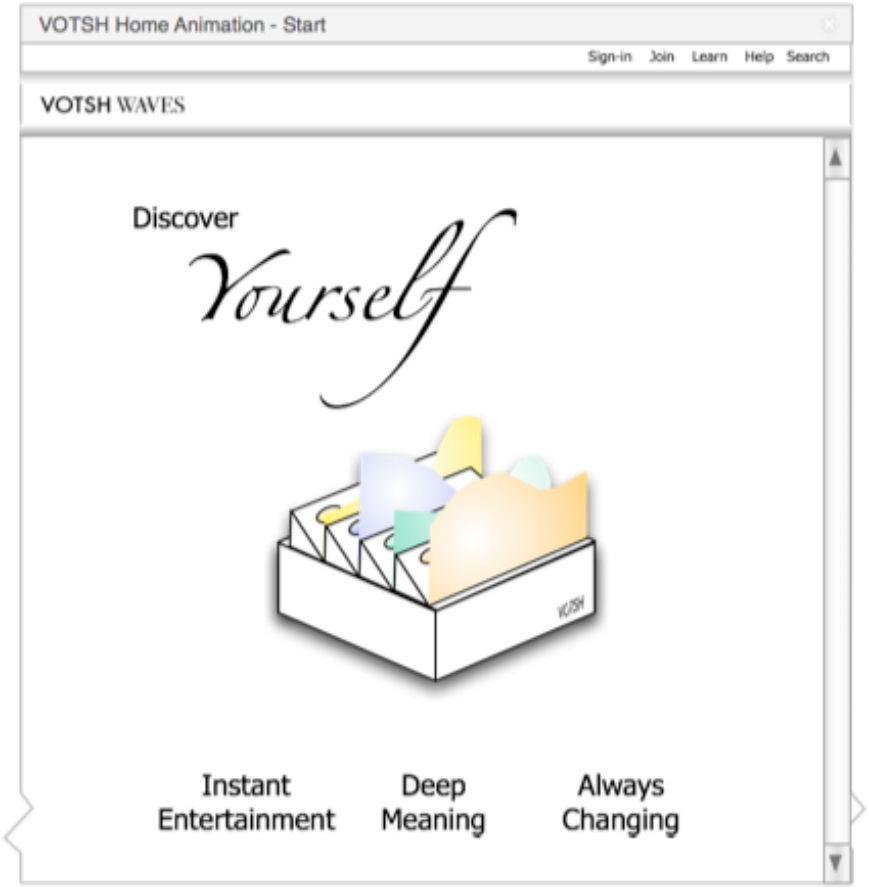
1. Amazon Machine Image (AMI) for backend services. AMI comes with Centos 5.4, Tomcat, Java 1.7, MySQL, Node.js, Apache installed.
2. C++ source code, including Header (.h) files stored in a Github repository
3. Unit test source code for all Wave components and backend services
4. 1 complete show demonstrating all features using 1 Wave unit, 1 region
5. 1 complete show demonstrating all features using 4 Wave units, 4 regions

Functional Requirements

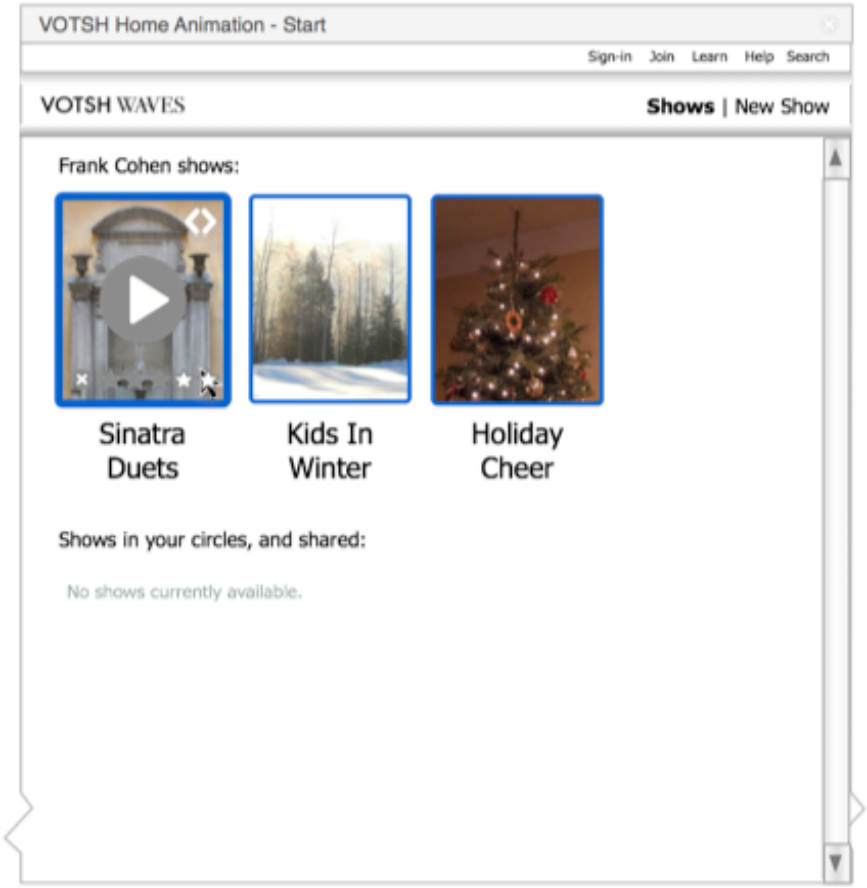
Waves requires the following functions and user interactions:

- Welcome Page (visitors)
- Start Page (customers)
- Sign-in and Registration
- Show Scheduling
- Show Authoring
- Payments
- Administration

Welcome Page



Start Page



Sign-in and Registration

People log-in using an ID (1 to 10 characters and password)

New registrations require Email Verification. Waves emits an email message. The message displays a clickable link back to the email verification servlet.

Forget Your Password?

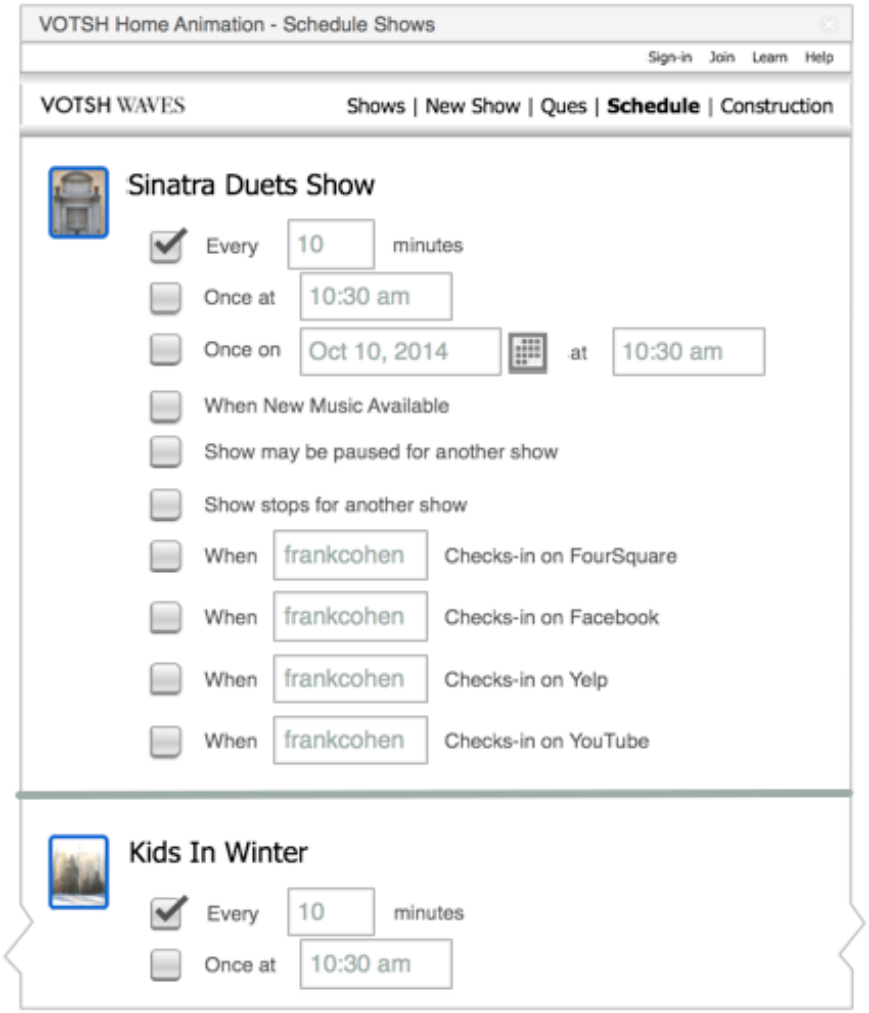
Payments

Buy An Entitlement

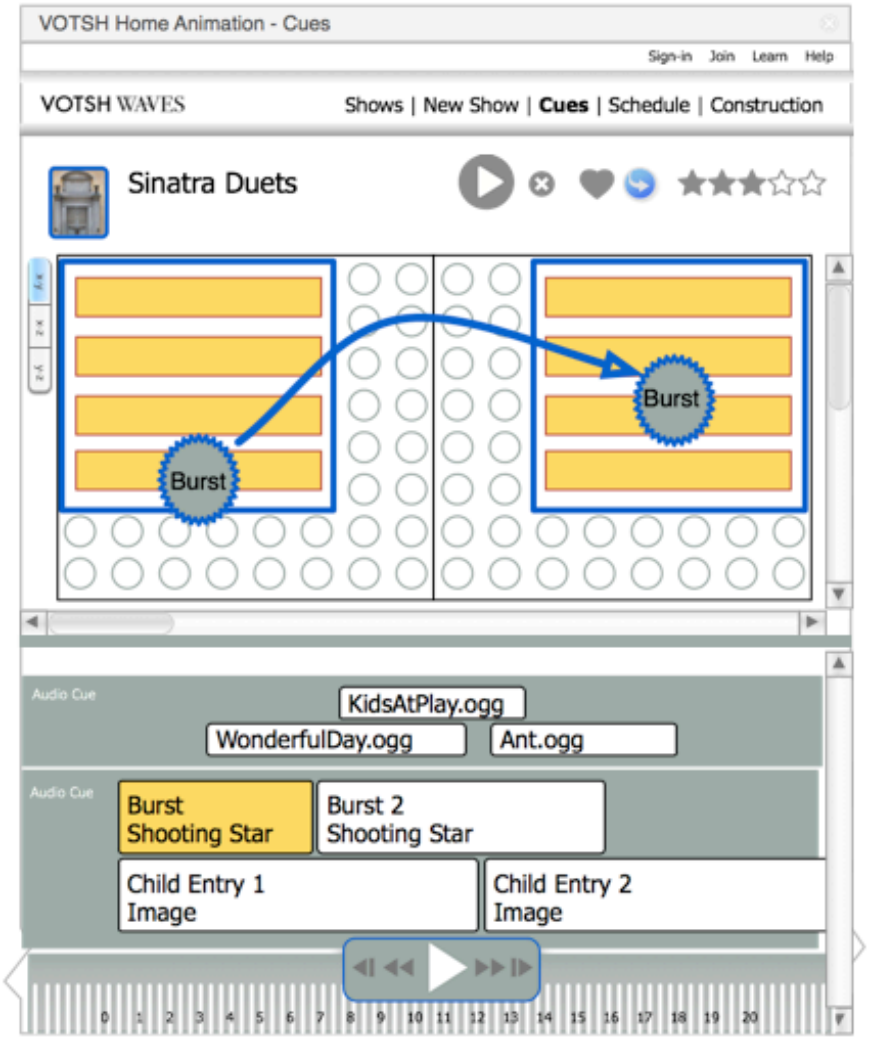
1 Year Entitlement plus 1 Wave unit for $99 USD

Chargify merchant account services

Show Scheduling



Show Authoring



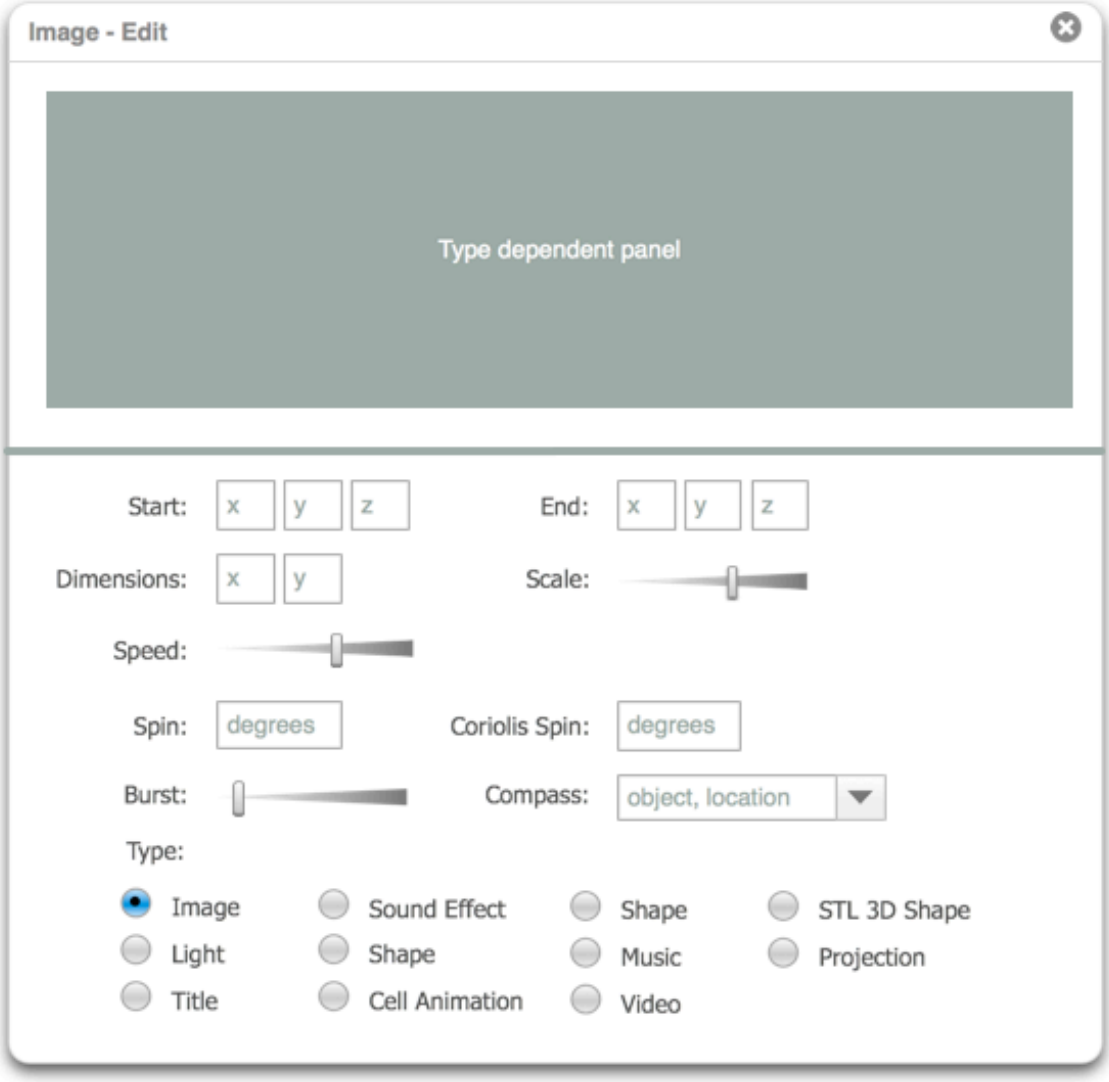
There is no "save" feature. User changes are automatically saved to the repository.

Spline allows multiple points, Remove point by click-and-delete key, Add point by click on spline, Arrow indicates direction of movement

Multiple Selections allowed using Shift-Click, Right-click displays helper pop-up menu

Helper menu offers: Group/Un-Group, Add New Type...

Sprite Detail Editor



User opens the lower panel as an "Options" panel, panel slides down from type dependent panel area.

User clicks X icon or anywhere outside of panel to close

If options panel is visible when panel is closed, then any other type dependent panel opens with options panel open.

For example,

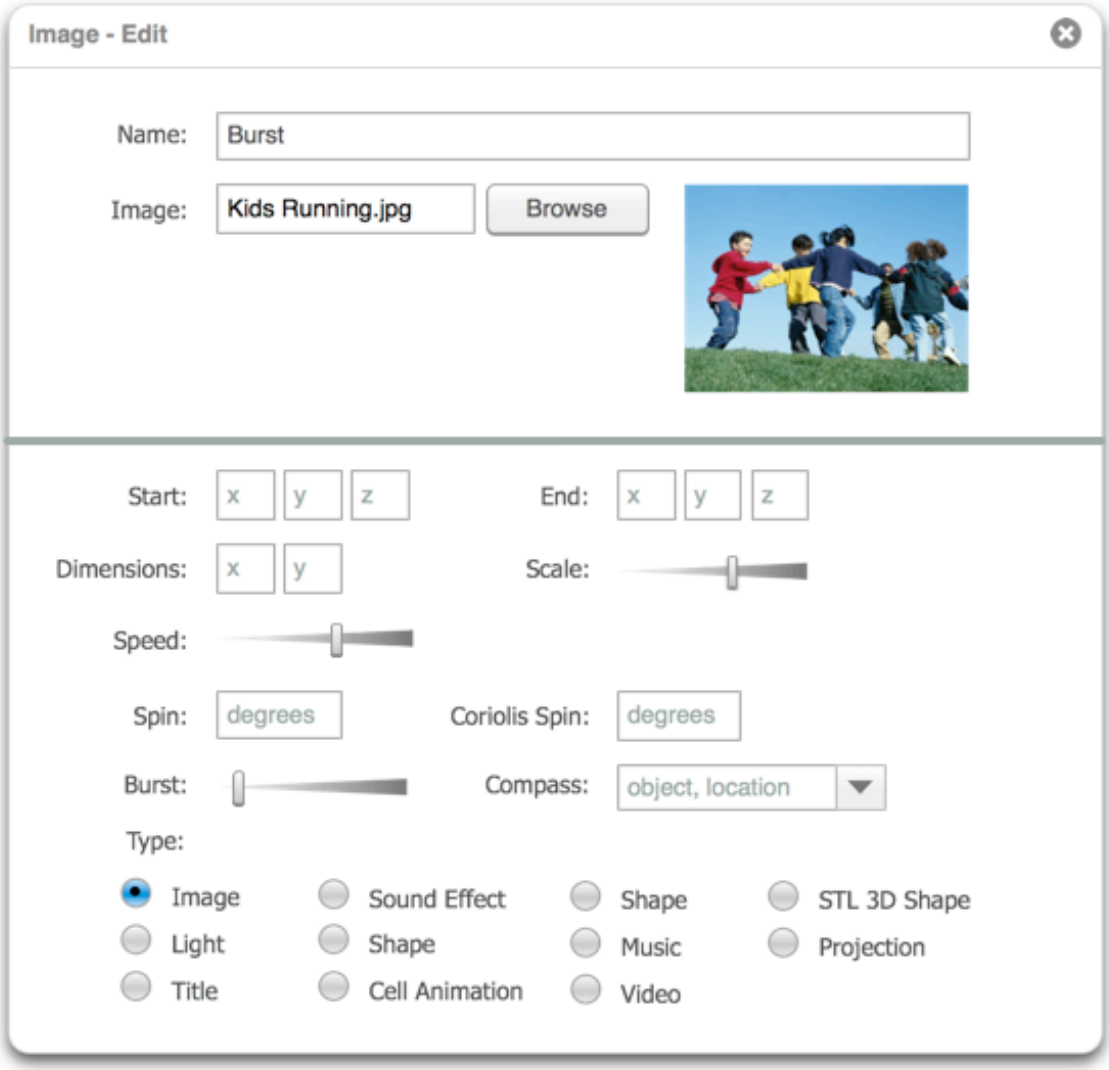
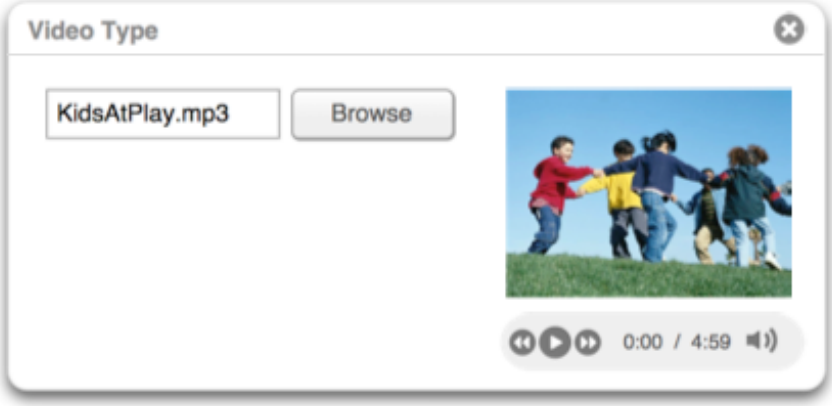
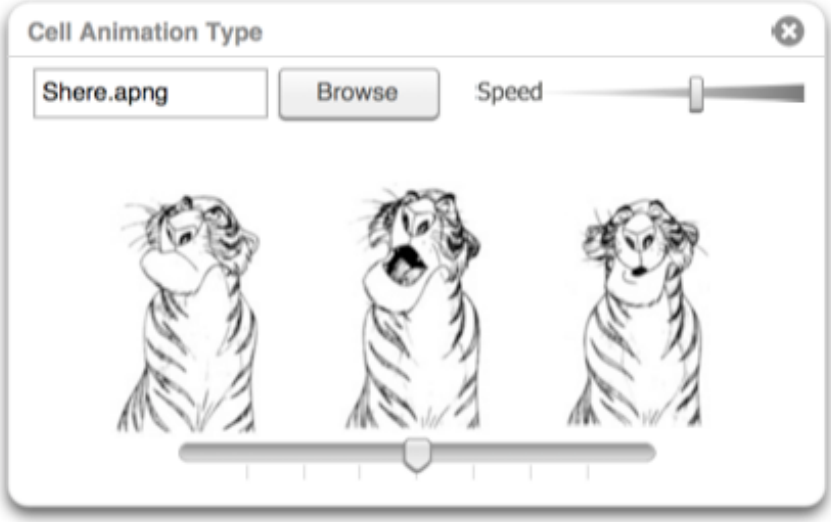


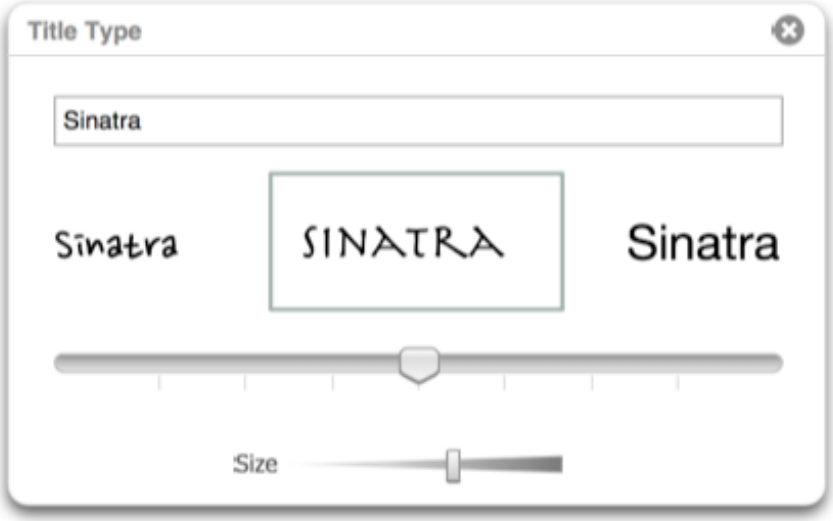
Image - Edit
Name:
Burst
Image:
Kids Running.jpg
Browse
Start:
x
y
z
End:
x
y
z
Dimensions:
x
y
Scale:
Speed:
Spin:
degrees
Coriolis Spin:
degrees
Burst:
Compass:
object, location
Type:
Image
Sound Effect
Shape
STL 3D Shape
Light
Shape
Music
Projection
Title
Cell Animation
Video

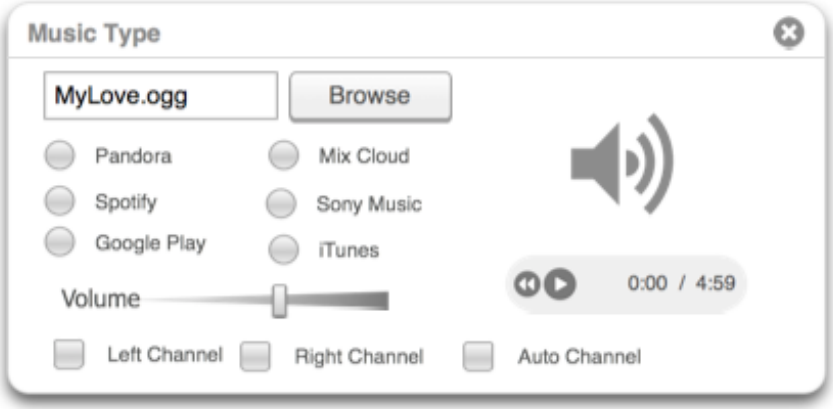
The following are user interactions for each Sprite Type.

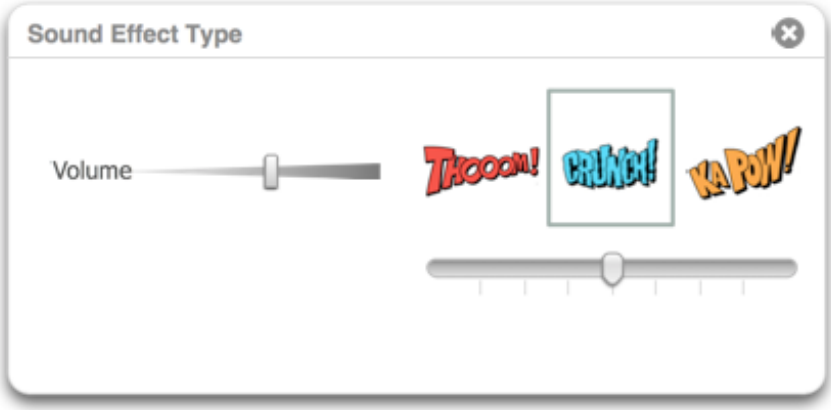


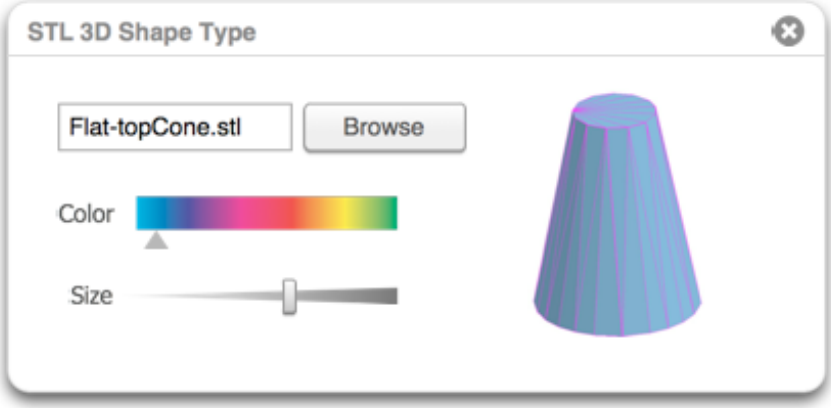


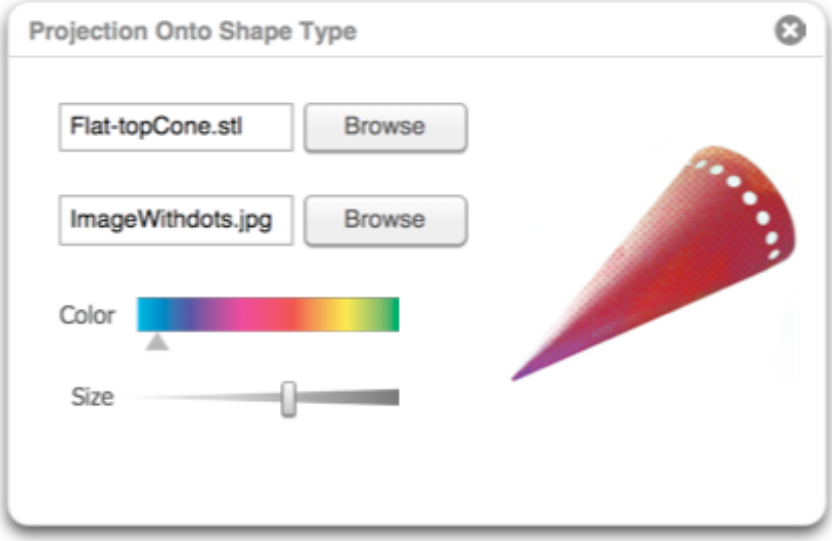

Shape Type
Color
Hex Value
#6cc6ff
Size


Title Type
Sinatra
Sinatra
SINATRA
Sinatra
Size

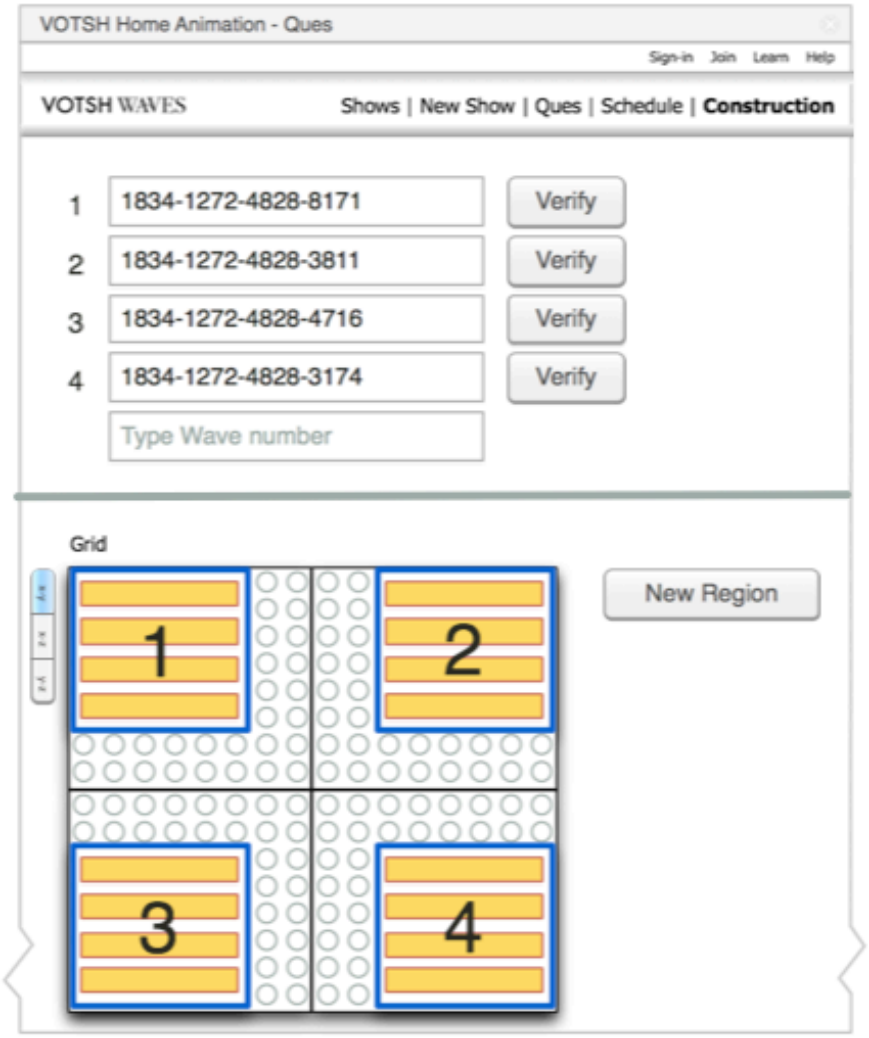

Music Type
MyLove.ogg
Browse
Pandora
Mix Cloud
Spotify
Sony Music
Google Play
iTunes
0:00 / 4:59
Volume
Left Channel
Right Channel
Auto Channel


Sound Effect Type
Volume
THOOOM!
CRUNCH!
KA POW!


STL 3D Shape Type
Flat-topCone.stl
Browse
Color
Size

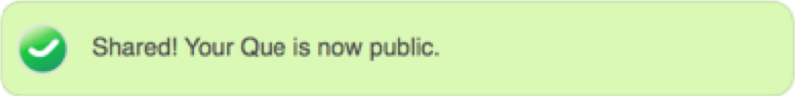

Projection Onto Shape Type
Flat-topCone.stl
Browse
ImageWithdots.jpg
Browse
Color
Size

Construction



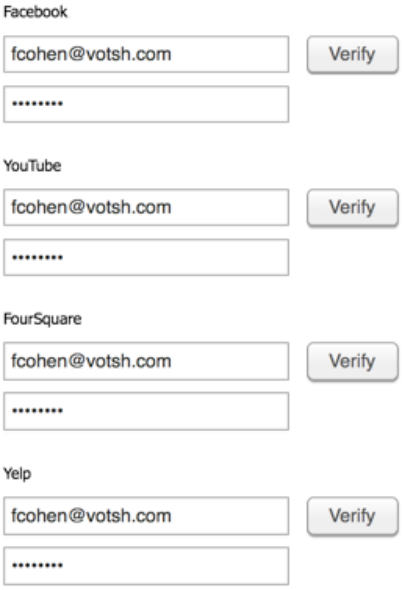
Standard Dialog

Standard Acknowledgementz



Social Media Share

Social Mobile Construction Settings



Administration Requirements

- Enable, disable user account
- Make Show private, public, disabled, delete
- Reset Logs
- View Logs
- Change password

Source Code Copyright Notice

All source code files must have the following copyright notice:

```
/*********************************************************************
* Votsh Confidential
*
*  Copyright 2014 Votsh. All Rights Reserved.
*
* NOTICE:  All information contained herein is, and remains
* the property of Votsh and its suppliers,
* if any.  The intellectual and technical concepts contained
* herein are proprietary to Votsh
* and its suppliers and may be covered by U.S. and Foreign Patents,
* patents in process, and are protected by trade secret or copyright law.
* Dissemination of this information or reproduction of this material
* is strictly forbidden unless prior written permission is obtained
* from Votsh.
*
* This file is subject to the terms and conditions defined in
* file 'LICENSE.txt', which is part of this source code package.
*/
```

For More Information

Please contact Votch at: phone +01 408 364 5508 (California, USA, GMT-8) email fcohen@votsh.com. Additional knowledge is available at www.votsh.com