SceneScape Scripts v2

Dave Beyer

Contents

1 Introduction to SceneScape Scripts

SceneScape scripts (also called “Sceneplays”) are used to specify the relationships, presentation and timing of the photo, sound, video, text, and annotation elements of individual scenes, and can also be used to describe the hierchical arrangement among scenes within a SceneScape.

Scenes are presented on a SceneScape “stage,” which is implemented by specific HTML, CSS, and Javascript code that knows how to parse, render, animate, and interact with the viewer to present a scene, possibly along with sub-scenes.

The following examples leverage three built-in stage types: *simple*, *image map“, and”listing.” See the Initial Stage Types section at the bottom for a description of these.

SceneScape scripts are typically expressed in the SceneScape Script XML language (though they can be easily converted into JSON, e.g., for storage in a JSON-based database or communicating over a REST API from server to client).

2 Examples

2.1 Example 1 - Simple, Multimedia Scene

The following simple example includes an audio clip followed by a video clip. The audio clip, which is played for 20 seconds, contains two text elements and a photo element which are presented during the audio, and the photo element has its own text element (caption) which is presented along with that photo. That’s followed by a video which is played in full, and contains three photos which are also presented, e.g., overlaying the video in a corner. The video includes

```
<scene stage="simple">
  <audio src="http://media.scenescape.com/ex/I'm_Too_Sexy.mp3" duration="20">
    <text> Can you name the charicatured minions? </text>
    <photo src="http://media.scenescape.com/ex/minion3.jpg">
      <text> It's Dave!? </text>
    </photo>
    <text>
      Minions are small, yellow, cylindrical, creatures who have one or two eyes.
    </text>
  </audio>
  <video src="http://media.scenescape.com/ex/minion.mp4">
    <photo src="http://media.scenescape.com/ex/minion4.jpg"/>
    <photo src="http://media.scenescape.com/ex/minion5.jpg"/>
    <photo src="http://media.scenescape.com/ex/minion6.jpg"/>
  </video>
</scene>
```

2.2 Example 2 - A Story of a Cowboy.

Note that on this “simple” stage, which shows only one video or photo element at a time, the parent video element is initially hidden while the first two photos are being shown, so is only used for its audio during that portion of the scene. At 18 seconds into the scene, a gap between the photos causes the video to be shown for 7 seconds.

So, the first two photos are shown for 9 seconds each (overriding the default photo duration of 10 seconds, set for the scene), and the final two photos are shown for the default of 10 seconds (specified by the default "duration" attribute of the `<scene>` element). After 45 seconds (9 + 9 + 7 + 10 + 10), the scene ends, regardless of how much of the video has been played (or replayed, if it needed to loop back to the start to fill the time).

Note also that two of the photos include captions, one which waits for 2 seconds after the presentation of its photo to appear.

```
<scene stage="simple" photo-duration="10">
  <sceneprops>
    <period begin="2016-01-01" end="2016-06-01"/>
    <description>
      John James was born, and lived all his life, in Grizzly
      River, Wyoming.  He ...
    </description>

    <!-- feature photo and sound, e.g., used in super-scenes -->

    <sceneshot src="http://media.scenescape.com/ex/cowboy1.jpg"/>
    <scenecast src="http://media.scenescape.com/ex/cowboy-montage.mp3"/>
  </sceneprops>

  <!-- Start of scene content -->

  <video src="http://media.scenescape.com/ex/cowboy.mp4">
    <photo src="http://media.scenescape.com/ex/cowboy1.jpg">
      <text start="2">John James is an original...</text>
    </photo>
    <photo src="http://media.scenescape.com/ex/cowboy2.jpg">
      <text>... born and bred in
            planes, rolling hills, and big sky of Grizzly
            River, Wyoming.
      </text>
    </photo>

    <gap start="18" duration="7"/>

    <photo src="http://media.scenescape.com/ex/cowboy3.jpg"/>
    <photo src="http://media.scenescape.com/ex/cowboy4.jpg"/>
  </video>
</scene>
```

The `<sceneprops>` elements provide inforamation about this scene that can be used in parent scenes.

2.3 Example 3 - A Story of a Cowboy, Multiple Videos

This example is similar to the previous Cowbory Story, but includes both a background video and multiple foreground videos, as well as some foreground photos. This allows for presentations like the this story about Prince performing in the rain at the Super Bowl.

```
<scene stage="simple" photo-duration="10">

  <video src="http://media.scenescape.com/ex/cowboy.mp4">
```

```
        <!-- show background video for 3 secs -->
        <gap   duration="3"/>

        <!-- show a photo-->
        <photo src="http://media.scenescape.com/ex/cowboy1.jpg"/>

        <!-- show another video, turn volume of background video -->
        <video src="http://media.scenescape.com/ex/cowboy2.mp4">

        <!-- show background video for 5 secs starting at time 15 -->
        <gap   start="15" duration="3"/>

        <!-- show another photo -->
        <photo src="http://media.scenescape.com/ex/cowboy2.jpg"/>

        <!-- show another video -->
        <video src="http://media.scenescape.com/ex/cowboy3.mp4">

        <!-- show background video to the end -->
        <video/>

      </video>
    </scene>
```

2.4 Example 4 - Simple, Multi-Scene Sceneshow

This example presents a scenescape that plays three sub-scenes in sequence on the “simple” stage:

```
    <scene stage="simple">
      <text duration="2">Sights and Sounds of Costa Rica</text>

      <scene src="http://scenescape.com/dave/costarica/playagranda.scene"/>
      <scene src="http://scenescape.com/dave/costarica/lafortuna.scene#t=8,14"/>
      <scene src="http://scenescape.com/dave/costarica/matapala.scene"/>
    </scene>
```

Note that only a portion of the second scene is shown, from 8 seconds to 15 seconds.

Also, the first scene has a title defined as an attibute, which will override any scene title provided in the references sceneplay script.

2.5 Example 5 - Image-Stage, Multi-Scene Sceneshow

This is the same scenescape, but shown on the Image stage

```
    <scene stage="image">
      <sceneprops>
        <!-- background, backdrop image, e.g., distorted map -->
        <stagedrop src="http://scenescape.com/dave/costarica/CostaRicaMap.png"/>

        <!-- background, underscore audio, quieted when listening to sub-scene -->
        <stagescore src="http://scenescape.com/dave/costarica/SoundsOfCostaRica.mp3"/>
    </sceneprops>

    <scene variable="lat:8;lon:14"
           src="http://scenescape.com/dave/costarica/playagranda.scene" />
    <scene variable="lat:10;lon:23"
           src="http://scenescape.com/dave/costarica/lafortuna.scene#t=8,14"/>
    <scene variable="lat:45;lon:88"
           src="http://scenescape.com/dave/costarica/matapala.scene"/>
  </scene>
```

On an image stage a `<stagedrop>` is required to provide the background backdrop image for the stage. A `<stagescore>` is optional, to provide a background, underscore sound track.

The variable attributes indicate where on the stagedrop image to anchor the scenes (in % from left, % from top).

Note that the sub-scenes may be on a simple stage, or may themselves be on an image map stage.

2.6 Example 6 - Scene Employing String Templating

SWIG-style, string templating instructions can be included to permit string replacements, looping, conditionals, and other instructions in the SceneScape scripts.

Here's a simple example (using browser-side templating, see the String Templating section below for details), which loops over a number of photos in a sprite image file (with individual images 400px wide by 200px high), showing each image for a duration that varies ("eases in") from 0.1 to 1.0 seconds as the image number varies from 0 to 75 using the "f_range()" and "f_easeIn()" SceneScape template methods.

(The "f_" prefix is used to help identify functions which are available within the template and are expanded while compiling the template. System variables available in the template are prefixed by 'v_')

```
<scene>
  {% for i in f_range(75) %}
    <photo src="http://media.scenescape.com/dave/costarica-sprite.png"
           duration="{{f_easeIn(i, 75, 0.1, 1.0)}}"
           spritedim="400,200", sprite="{{i}}">
  {% endfor %}
</scene>
```

2.7 Example 7 - Auto-Generated Scenes

Leveraging the String Templating, with the SceneScape extension "f_search" methods, scenes (and superscenes, or scenescapes) can also be auto-generated, consisting of a list of scenes determined by a search criteria, such as the following two cases:

```
<!-- most recently added 100 scenes with geolocations -->
{% for scene in f_sceneSearch('recents', 'limit:100', 'type:geo') %}
  <scene stage="map" src="{{scene}}" />
{% endfor %}
```

or

```
<!-- most popular 100 scenes from this week geolocations -->
{% for scene in f_sceneSearch('popular', 'since:7 days', 'type:'geo') %}
  <scene stage="map" src="{{scene}}" />
{% endfor %}
```

3 Timing & Effects

3.1 Timing & Transition Examples

Some example considerations of scene timing & transitions include:

- when to show text and comments, in association with the display of a particular photo of a sceneshow, at a particular time during the playing of a sound, or at a particular time during the playing of a movie.
- when to transition to new photos in a sceneshow, either to occur when the duration of the previous photo completes, or at a particular time in the accompanying sound or associated video.
- when to switch to displaying or focusing on a new sub-scene within an auto-running SceneScape.

3.2 Relative, Positioned, and Absolute Timing

The *start* and *duration* attributes can be used to specify the timing for the appearance or playing of any element in the scene:

- start - start time, in seconds, for this element (default 0)
- duration - number of seconds to play this element (defaults to what's required to fill the parent element, or for parent video or audio elements, to its natural duration).

The start time can be relative, positioned, or absolute, defined similarly to the corresponding CSS positioning terms as follows (ordered by the order they are considered when computing the schedule):

- *absolute* - the start time is set with the specified offset from the start time of the nearest containing (displayed) parent element, and, analagous to CSS absolute positioning, such elements do not affect the timing of any other (e.g., sibling) elements. Absolute times are prefixed by "@" (e.g., start="@15.5").

- *positioned* - like absolute times, the start time is set with the specified offset from the start time of the nearest containing (displayed) parent element. However, positioned elements do affect the timing of relatively-timed sibling elements. Positioned times have no prefix (e.g., start="15.5" to start 15.5 seconds after the start of the parent element).

- *relative* - the normal start time will be adjusted by this amount. For an element following a sibling, relatively-timed element, the normal start time would coincide with the completion of the element before it. Relative times are prefixed by either a '+' or a '-' (e.g., start="-1.2" for starting 1.2 seconds earlier than normal). This is analagous to CSS relative positioning.

If no start time is given for an elament, then it is treated as relatively timed with a 0 adjustment.

Note that durations of child elements are often considered as suggestions, and may be scaled as needed to exactly fill the duration of the parent element.

In addition, to allow only a portion of an audio or video element to be played, clip start & end times can be specified for these elements, for example:

```
    clip="12.3,21.1"
```

to only play the portion between 12.3 and 21.1 seconds into the audio or video.

Note that this can also be accomplished by specifying the startime and endtime on the URL to the media element as desribed here, like the following

```
<video src="http://media.scenescape.com/ex/cowboy.mp4#t=10.4,20.0">
```

Also note that "duration" and the "clipduration" for a video or audio element are different. For example, if the clipduration (or full duration) of a video is 10 seconds, and the element duration is 15 seconds, then the video will be played exactly one and a half times.

3.2.1 Timed Effects

SceneScape script elements can be assigned effects which can specify start or end transitions, or other visual or audio effects. They are specified using the "effect" tag and have the following structure:

```
effect="[<effect start time>]<effect name>(<effect args>)"
```

where the 'effect start time' specifies start time of the effect as a '+' or '-' offset from the scene's start ('S') or end ('E') time, start being the default.

Multiple effects can be delimited by semicolons.

For example, to start a fadeIn effect that last 100 msecs and starts immediately with the start of the element, plus a fadeOut effect that also lasts 100 msecs and starts 200 msecs before the end of this element's duration:

```
effect="s+0:fadeIn(0.1); E-.2:fadeOut(0.1)"
```

Or, because an offset of 0 secs from the start ('s') are the defaults, this could also be specified as:

```
effect="fadeIn(0.1); E-.2:fadeOut(0.1)"
```

Like other default attributes (see section on Scene Defaults), default effects can also be specified in parent elements. For example, to specify the above effects for all children (and grandchildren…) photo elements, use:

```
photo-effect="fadeIn(0.1); E-.2:fadeOut(0.1)"
```

3.2.2 Possible Future Addition

- *fixed* - the start time is set with the specified offset from the start of this scene being displayed and analagous to CSS fixed positioning, such elements do not affect the timing of any other sibling elements. Fixed times are prefixed with "!" (e.g., start="!15.5").

3.3 Starting a Scene Somewhere in the Middle

Anchor tags can specify the time to jump to in a scene using the following format:

```
<a href="dave.scenescape.com/Trips/NYC/lesmis#t=34.5"> Love this part! </a>
```

This would link to 34.5 seconds into the "Les Mis" scene.

4 Scene Properties & Defaults

4.1 Sceneprops

Scene properties uses include: * providing summary information of a scene for presentation in a parent scene, and * specifying parameters used by the scene's stage for presenting the scene.

<sceneprops> elements allow scene properties to be specified in a way that makes it clear that they are separate from the synchronously timed content elements of the scene. However, all of these properties can be specified as either attributes of <scene> or as children of a <sceneprops> element, or even as direct children of <scene>. One form is often preferred over another depending on the circumstances (e.g., whether a scene is being described, or a scene is being included in a parent scene with some scene property being overridden).

Here are equivalent ways to express scene properties:

Scene property	Attribute of <scene>	Child of <scene> or <sceneprops>
Title	title="…"	<title>...</title>
Description	description="…"	<description>...</description>
Period	period="t1,t2"	<period begin="t2" end="t2"/>
Summary photo	sceneshot="…"	<sceneshot src="..."/>
Summary audio	scenecast="…"	<scenecast src="..."/>
Tag	tag="type(user):clip:value"	<tag type".." clip=".." user="..">..</tag>
		<variable type="type2" value="value2"/>
Background image	stagedrop="…"	<stagedrop src="..."/>
Background audio	stagescore="…"	<stagescore src="..."/>
Image map x axis	stagexaxis="type:min,max"	<stagexaxis type=".." min=".." max=".."/>
Image map y axis	stageyaxis="type:min,max"	<stageyaxis type=".." min=".." max=".."/>
Path to this scene	scenepath="…"	<scenepath>...<scenepath/>

For a Tag property specified as an attribute, the id and value are mandatory, and the clip and user are optional. See the "Scene Tags" section below for details.

4.2 Default Scene Attributes

Default values for attributes can be set in parent elements (such as in <scene>). If the default should only apply to certain types of elements, prefix the name by the element name followed by a hyphen.

For instance, to set the default duration for all child (and grandchild, etc.) elements to 5 seconds, use:

The Variable property is used for positioning in image map stages, where the variable "type" could be 'latitude', 'longitude', 'date', 'time', 'datetime', or a custom-defined type giving the average July temperature of the region captured by the scene.

```
<scene ... duration="5">
```

Or, to set a default for just video elements, use:

```
    <scene ... video-duration="10">
```

4.3 Scene Tags

Scene tags provide a flexible way to annotate scenes with additional metadata and user input. Scene tags can appear either in the scenprops element, or can be embedded in the content of the scene.

Scene tags have the following structure:

```
  <tag type="..." clip="..." user="...">
     ... value ...
  </tag>
```

where "type" and "value" are required, and "clip" and "user" are optional. "clip" can provide the start and stop time within the relevant scene or scene element where the tag applies (e.g., clip="4.5,6.7" for 4.5 secs to 6.7 secs into the scene or scene element), and "user" can optionally provide one, or a list of, user ID(s) that apply.

Tags can equivalently be specified as attributes on or other elements using the following format:

```
   tag="type1(user1):clip1:value1;type2(user2):clip2:value2;..."
```

where clip and user are optional. Any colons or semicolons in the value must be escaped with a leading ". Here are some example tag attributes:

```
   tag="ss-date:1June2016"
   tag="ss-lat:37.338N;ss-lon:121.893W"
   tag="ss-comment(17):Very cool \:-)"
   tag="ss-comment(17):17.3,18.4:Nice move!"
```

The tag 'type's should correspond with a type in a ".tag" file within the SceneScape hierarchical search path of the relevant scene. Tag types should consist of alphanumeric characters Semicolons are used as a separator when a tag is specified in a script element's attribute.

The *.tag files have the following format:

```
<scenetags>
  <tagdef type="...">
    <title>       ... </title>
    <description> ... </description>
  </tagdef>
  ...
</scenetags>
```

TBD consider list of valid values.

4.3.1 Common Scene Tags

Common tags include:

Tag type	Title	Description
ss-lat	Latitude	
ss-lon	Longitude	
ss-data	Date	
ss-time	Time	
ss-datetime	Date & time	

4.3.2 Custom Scene Tags

Custom scene tags could be used for various purposes, such as:

- Categorizing scenes or scene clips with custom categories
- Assigning metrics to scenes or scene clips

5 String Templating

SWIG-style, string templating instructions can be included in SceneScape Scripts to permit string replacements, looping, conditionals, and other instructions.

5.1 Variable extensions

A number of system variables are available to the template, including the following:

Variable	Description
v_path	Path to the current scene
v_dir	Directory portion of the path

5.2 Function extensions

The SWIG templating is extended to allow for convenience methods like “f_range” and “f_easeIn”, as used in an example above. See here for how functions are implemented to be made available to SWIG. Extension functions include:

Function	Description
f_range(max)	
f_easeIn(num, maxNum, minTime, maxTime, power)	
f_sceneSearch	

5.3 Template Insertion extension

Partial scene templates can be created and inserted into Scene Scripts (or even other templates) using the following notation:

```
{% insert t_<templateName>(<templateArgs>) %}
```

Where `<templateName>` is replaced by the name of the template file, and `<templateArgs>` replaced by args (if any) used in rendering the template before insertion.

5.4 On the Server

Typical string templating replacement is done on the server, prior to passing the script (in the form of a JSON structure) of the REST API to the client for execution.

5.5 On the Client

STILL UNDER INVESTIGATION

Also, the following modified notation can be used to pass templating instructions to the browser, where the outer brackets replaced by square brackets,

```
[{ ... }]
[% ... %]
[# ... #]
```

6 Stage Types

The following initial stage types are planned:

- *simple* - This default stage for a scene, shows one, or a sequence, of scenes along with their sounds, text & comments in a full-page manner, one video or image at a time, similar to a photo slideshow.
- *listing* - This default stage for a folder, shows a listing of the scenes along with their sceneshot (summary photo), each linked to its scene.
- *map* - a stage that previews the sites and sounds of one or more scenes of a scenescape, each located somewhere on a background image "map". The image map may represent an area, or some other image with x and y axes associated with a location or some other measure. The scenes being previewed at any moment depends on the viewer's mouse pointer location on the image map. The bounds stage property must be provided for image map stages, and may represent latitude and longitude, or just about any other measure, such as height, proximity to a major city, average summer in July, etc. Exmaple image maps include:
- A google map with the axes representing lat/lon location.
- A distorted map of a wedding reception, and nearby town, with x & y representing locations but distorted by the not-to-scale distortions of the image (likely requiring a handful anchor points to allow the distorted map algorithm to more accurately position the scenes).
- A simple graph, perhaps overlayed on a "friendlier" image, representing two characteristics such as average temperature increase versus people's opinion on climate change.

7 Photo Sprites

Image "sprites" are composite photo files that contain multiple individual photos photos and allow quickly switching between photos without needing to individually download each one. Two attributes for photo elements are used with sprite images:

- *spritedim* - to specify the dimensions, width by height in pixels, of each sprite image in pixels. (If only one is given, it is used for both width and height.)
- *sprite* - to indicate which sprite to show in the form of row by column. (If only one the row is given, column 0 is used.)

An example use, for a sprite image consisting of a column of images, each one 400px in width and 200px in height.

```
<photo src="http://media.scenescape.com/dave/costarica-sprite.png"
       spritedim="400,200", sprite="4">
```

8 Importing and Annotations

8.1 Importing Scenes

Before a scene can accept any annotations (such as viewer comments or likes), or before it can be reliably referenced from another scene, it must be "imported" via the SceneScape server.

When a SceneScape script is "imported", the scene, and each element in the scene that doesn't already have a unique identifier is assigned one (something like "ss2-334s2kd"), so that it can be referenced by viewer annotations or by other scenes. Any update to the SceneScape Script should retain these same identifiers, so that any annotation references (see next section) aren't lost.

8.2 Viewer Annotations

Although possible to include in the scene itself, comments, likes, face and voice annotations are typically inserted into a scenescript by the server before delivering to the client by pulling in the annotations associated with the scene element identifiers from a separate database.

Common viewer annotations include:

Tag type	Title	Description
comment	Viewer comment	
like	Viewer comment	
face	Facial identification	
voice	Voice identification	

Annotation elements include a mandatory "user" and an optional "clip" attribute. Here are some examples:

```
<like user="17"/>
<comment user="1234" clip="17.7,22.0> Very cool! </comment>
```

These annotations also include the following attributes (at least stored in the database):

- ownerId: userId of the person who added the annotation
- created: when annotation was added
- modified: last time annotation was modified (if it was)

9 Outstanding questions

- Metalanguage definition for Scene Templates used for crowd- and friend-sourced scenes? Perhaps this is based on the String Templating script extensions above? Also, consider combining with auto-generated scenes.