Limelight OSMF Player 1.0.0

UNDER CONSTRUCTION

Sample Limelight OSMF Player

Overview

The Limelight OSMF sample player demonstrates how to use the OSMF framework with various types of media provided by the Limelight CDN.

- As a basic player, it downloads and plays any provider's media content using the Limelight CDN for distribution. It provides a control bar to manage the media.
- As a basic application, it can be embedded on a web page to play back media.
- It supports all standard Limelight media content types.
- It provides external interface calls to control the user interface, demonstrating how to change the media source on the fly.

The OSMF sample player project contains test assets to verify the following features:

- Progress
- Streaming
- Media Valet
- Live Stream
- MBR (multi bitrate)

Features

The following features are provided in the sample OSMF player:

- Load (URL parameter can be passed to the SWF)
- Play state (play, pause, stop)
- Seek
- Volume (increase, decrease)
- Elapsed and total play time
- Multiple Bitrate
- Control bar hiding (toggle, `autoHideControlBar` parameter can be passed to the SWF)
- Full screen (toggle)
- Auto play (toggle)
- Loop/Auto replay (toggle)
- Poster frame (toggle, definable source URL)
- Background fill (`backgroundColor` parameter can be passed to the SWF)
- Debug support

By leveraging the *DefaultMediaFactory* class, the player can distill the following media element types from the input URL:

- *VideoElement*, using either:
  - NetLoader (streaming or progressive)

    - DynamicStreamingNetLoader (MBR streaming)
    - HTTPStreamingNetLoader (HTTP streaming)
    - F4MLoader (Flash Media Manifest files)

- *SoundElement*, using either:
    - SoundLoader (progressive)
    - NetLoader (streaming)

- *ImageElement*
- *SWFElement*

Installation

Instructions for Flex Builder

1. Unzip/copy the `LimelightOSMFPlayer` project into your Flex Builder workspace folder.
2. In Flex Builder (or Flash Builder), go to the *File* menu and select *Import*.
3. Browse to the Flex Builder workspace folder.
4. Click *Finish*. This will import the project.
5. Link to `OSMF.swc` library.
6. Build the project.

Note: To compile this player in Flex Builder for use with HTTP Streaming (Zeri), you may need to update the `playerglobal.swc` to support the Flash Player 10.1 API.

Note: To compile this player in Flash Builder, you may need to update the `OSMF.swc` in the default SDK.

Embedding in a Web Page

The `html-template\index.template.html` file shows how a Player SWF can be embedded in a web page using `SWFObject`. This template also shows the properties that can be set on the Player SWF.

Linking to the Plugin

This sample player requires the installation of the Limelight Streaming Plugin. It is currently linked to a copy of the plugin located on the Limelight CDN for test and demonstration purposes. A copy of the plugin should be downloaded and linked on the customer's network before going into production.

FlashVar Parameters

Setting	Parameter	Required	Possible values	Default	Description
Player Id	id				Used for analytics.
Media URL	url	x		none	URL of the media to be played.
Media stream type	streamType		liveOrRecorded, live, recorded, dvr	liveOrRecorded	The type of media stream to support. The default setting plays both live and recorded media, with no digital video recording (DVR) features.
Looping behavior	loop		false, true	false	Specifies if media playback is restarted when the end of the file is reached. The default behavior for the player is not to loop.
Automatic playback	autoPlay		false, true	false	Specifies if the media starts playing automatically, without user input. The default behavior for the player is to require the user to start playback.
Play Button Overlay visibility	playButtonOverlay		false, true		Specifies if a play button is displayed in the center of the player window before playback begins and when video is paused.

Method of scaling content	scaleMode		letterbox, none, stretch, zoom	letterbox	Determines how the source content is sized within the player window. The default value, *letterbox*, allows the content to be resized to fit the player window, but constrains the dimensions of the content to maintain its original aspect ratio. A value of *none* does not allow the content to be resized. A value of *stretch* sets the dimensions of the content to that of the player window, possibly changing the aspect ratio of the content in the process. A value of *zoom* fills the player window with the content while maintaining its original aspect ratio; this may cause cropping of the content's horizontal or vertical edges.
Control bar position	controlBarMode		docked, floating, none	docked	Controls the display location of the player's controls. The default value, *docked*, sets the control bar along the bottom of the player window. A value of *floating* displays the control bar hovering over the content, near the bottom of the window. A value of *none* means that no control bar is displayed.
Control bar visibility	autoHideControlBar		true, false	true	Specifies if the player's controls are hidden initially. With the default value of *true*, the controls are not displayed unless the user is hovering the mouse over the player. With a value of *false*, the controls are continuously visible and may reduce the amount of the player window available to display content.
Background color	backgroundColor		*Hexadecimal*		A six digit hexadecimal value that sets the background color of the player.
Poster bitmap	poster		*URL*		A URL specifying an initial bitmap image to display in the player window before playback begins. If no default poster image is provided, the player displays the current background color.
Limelight Stream Type	llnwStreamType		Progressive, Streaming, LiveStream, MBR, LiveMBR, DVR, FLVSeek, MOOVSeek		The Limelight stream type to support.
URL Includes FMS Application Instance	useAppInstance		true, false	true	Specifies if an instance name is used in the URL.
Player Volume	volume		0-100	50	Sets the initial volume level. Value represents a percentage from 0 to 100.
Automatic Quality Switching with MBR	autoSwitchQuality		true, false	N/A	Used with Multi Bit-Rate (MBR) to determine if the player will automatically switch between streams to provide the best possible viewing experience. (Not Currently Supported)
Buffer Time Before Play	bufferTime		*Number*	N/A	Used to set the size of the buffer to fill before the media will play. (Not Currently Supported)
XML configuration file	configuration		*URL*		A URL specifying the location of an XML configuration file. This provides the player with configuration information, including plugins to load and parameters from this table.

Other Notes

The Limelight MBR URL (used in previous players) will work with the sample player described in this document. The sample player contains an additional class for parsing the URL to pass to the `DynamicStreamingResource`. It will *not* work with Strobe or other custom players. This was done for backward compatibility with previous Limelight players.

For multi bitrates (MBR), the sample Player knows whether a switch up or down is possible based on the current rendering index of the switching profile (the contents of the `DynamicStreamingResource` object).

Limelight Streaming Plugin

Stream Types

The Limelight Streaming Plugin enables OSMF support of Limelight CDN Content. It supports stream types of Progressive, Streaming, Live Stream, MBR, Live MBR, HTTP Streaming, FLVSeek, MOOVSeek and DVR. However, it is only required for Live Stream, Live MBR, FLVSeek, and MOOVSeek.

Note: Streaming URLs can be manually configured to run without the plugin; however, if you are using the standard Limelight format (as used in previous players) the plugin is required.

Examples:

```
http://llnwqa.vo.llnwd.net/o18/content/lexsamplecontent/king_1500flv.flv (plugin required)
http://llnwqa.vo.llnwd.net/o18/content/lexsamplecontent/king_1500flv (plugin not required)
http://llnwqa.vo.llnwd.net/o18/content/lexsamplecontent/king_1500h264.f4v (plugin required)
http://llnwqa.vo.llnwd.net/o18/content/lexsamplecontent/mp4:king_1500h264.f4v (plugin not required)
```

Resource Files

A resource file should be used to pass a list of streams in OSMF. Options such as `f4m, xml`, and `smil` are all supported. Typically, the best option is `f4m` because it is supported natively by OSMF and does not require an additional plugins.

Example:

```
<?xml version="1.0" encoding="utf-8"?>
<manifest xmlns="http://ns.adobe.com/f4m/1.0">
 <baseURL>rtmp://llnwqa.fcod.llnwd.net/a1218/o18/</baseURL>
 <urlIncludesFMSApplicationInstance>true</urlIncludesFMSApplicationInstance>
 <media url="mp4:lexsamplecontent/king_500.f4v"  bitrate="500" width="480" height="270" />
 <media url="mp4:lexsamplecontent/king_800.f4v"  bitrate="800" width="480" height="270" />
 <media url="mp4:lexsamplecontent/king_1200.f4v"  bitrate="1200" width="640" height="360" />
 <media url="mp4:lexsamplecontent/king_1500.f4v"  bitrate="1500" width="640" height="360"/>
 <media url="mp4:lexsamplecontent/king_2400.f4v"  bitrate="2400" width="640" height="360"/>
</manifest>
```

Plugin Usage Instructions

The Limelight Streaming Plugin is loaded in the standard way via *DefaultMediaFactory*. To associate the plugin with a specific resource, the resource must have the Limelight metadata identifier.

1. Set the Limelight metadata identifier as a constant:

   ```
   public const LIMELIGHT_IDENTIFIER:String = "com.limelight.video";
   ```

2. Declare the Limelight metadata value object:

   ```
   var limelightMetadata:Metadata  = new Metadata();
   limelightMetadata.addValue("llnwStreamType",llnwStreamType);
   ```

   where *llnwStreamType* = Progressive, Streaming, LiveStream, MBR, LiveMBR, DVR, FLVSeek, or MOOVSeek.

3. Add the Limelight metadata identifier to the resource when it is created. Pass the stream type to the plugin via the metadata object:

   ```
   resource.addMetadataValue(limelightConstants.LIMELIGHT_IDENTIFIER,limelightMetadata);
   ```

   where *resource* = URLResource, StreamingURLResource, or DynamicStreamingResource.

Troubleshooting

A debug window is available for troubleshooting. This window is opened with CTRL/SHFT + Upper Arrow; it is closed with CTRL/SHFT + Down Arrow.

Plugin is loaded but not being used

1. Verify that the Limelight metadata identifier is added to the metadata and that the stream type is passed in.

Example:

```
streamingURLResource.addMetadataValue(LIMELIGHT_IDENTIFIER,limelightMetadata);
```

Limelight videos will not play in custom player

1. Verify that `urlIncludesFMSApplicationInstance` is set to true.
2. If the stream type is `LiveStream`, verify that the `StreamingURLResouce streamType` is set to LIVE_OR_RECORDED.
3. Verify that the Limelight metadata identifier is added to the metadata and the stream type is passed in.

Example:

```
var streamingURLResource:StreamingURLResource = new StreamingURLResource(url);

streamingURLResource.urlIncludesFMSApplicationInstance = true;
if (configuration.streamType=="LiveStream") {
  streamingURLResource.streamType = StreamType.LIVE_OR_RECORDED;
}
streamingURLResource.addMetadataValue(LIMELIGHT_IDENTIFIER,limelightMetadata);
```

Live Stream videos will not play

1. Verify that the `StreamingURLResouce streamType` is set to LIVE or LIVE_OR_RECORDED.
2. If the `urlIncludesFMSApplicationInstance` is set to true, verify that _definst_ is used in the URL for the instance name.
    a. `urlIncludesFMSApplicationInstance` must be set to true for (almost) all Limelight URLs to play. When the `urlIncludesFMSApplicationInstance` set to true, an instance name is required.
    b. A custom player could set `urlIncludesFMSApplicationInstance` to false and _definst_ would not be needed for live streaming.