Akamai® Plug-ins for Open Source Media Framework

Akamai Confidential
For Customer Use Under NDA Only

Updated: March 22, 2011

Akamai Technologies, Inc.

Akamai Customer Care: 1-877-425-2832 or, for routine requests, e-mail ccare@akamai.com

The EdgeControl® Management Center, for customers and resellers: http://control.akamai.com

US Headquarters
8 Cambridge Center
Cambridge, MA 02142

Tel: 617.444.3000
Fax: 617.444.3001

US Toll free 877.4AKAMAI (877.425.2624)

For a list of offices around the world, see:
http://www.akamai.com/en/html/about/locations.html

Akamai® Plug-Ins for Open Source Media Framework

Table of Contents

PREFACE **4**

Overview 4

Akamai Basic Streaming Plug-in 4

Akamai Advanced Streaming Plug-in 4

CHAPTER 1. LOADING PLUG-INS **5**

Loading Statically vs. Dynamically 5

Obtaining the Plug-ins 7

The Akamai Basic Streaming Plug-in 7

The Akamai Advanced Streaming Plug-in 7

CHAPTER 2. PLAYING MEDIA **8**

Overview 8

Using "MediaPlayerSprite" to Play Media 9

Using "MediaContainerUIComponent" to Play Media 10

CHAPTER 3. AKAMAI METADATA **11**

Overview 11

Attaching Resource Metadata 11

Akamai Basic Streaming Plug-in Metadata 12

Plug-in Resource Metadata 12

Media Resource Metadata 13

Akamai Advanced Streaming Plug-in Metadata 13

CHAPTER 4. SPECIAL USE CASES **14**

Resolving Plug-in Conflicts 14

Playing Akamai HD Network 1.0 for Flash Content without using the Akamai SMIL File Format 16

CHAPTER 5. LINKS TO SAMPLES WITH SOURCE **17**

Preface

In This Chapter Overview • 4

Overview

Akamai offers the following plug-ins for media players built with the Open Source Media Framework (OSMF):

- The Akamai Basic Streaming Plug-in
- The Akamai Advanced Streaming Plug-in

This document assumes that the reader has a general working knowledge of OSMF and OSMF plug-ins. For more information about OSMF, please see the documentation on the official OSMF site.

Akamai Basic Streaming Plug-in

This plug-in supports loading and playback of progressive download, FMS live, FMS on-demand, and FMS dynamic streaming content (the OSMF SMIL plug-in is required for FMS dynamic streaming). The plug-in provides improved buffer settings over the default OSMF buffer settings, support for connection level and stream level auth tokens, and support for recovery from live encoder crashes.

Akamai Advanced Streaming Plug-in

This plug-in is a superset of the *Akamai Basic Streaming Plug-in* and supports all media types including HTTP streaming (also known as Akamai HD Network -- the Flash Player 10.1 version of the plug-in is required for Akamai's HDN 2.0).

The table that follows compares both plugins:

	Open Source	Can Be Loaded Statically	Can Be Loaded Dynamically	Progressive Download	FMS Live	FMS On-Demand	FMS Multi-bitrate	Akamai HDN 1.0	Akamai HDN 2.0
Akamai Basic Streaming Plug-in	✓	✓	✓	✓	✓	✓	✓*		
Akamai Advanced Streaming Plug-in			✓	✓	✓	✓	✓	✓	✓**

* Requires the **OSMF SMIL Plugin**

** Requires the **Flash Player 10.1** build

Chapter 1. Loading Plug-ins

In This Chapter

Loading Statically vs. Dynamically • 5
Obtaining the Plug-ins • 7

Loading Statically vs. Dynamically

OSMF plug-ins can be loaded statically (at compile time) or dynamically (at run-time):

- **Loading Statically** - The plug-in source, or a static library (SWC) version of the plug-in, is required to load an OSMF plug-in statically.
- **Loading Dynamically** – The plug-in as a compiled SWF file can be loaded dynamically over HTTP or FILE protocols. When only a SWF is supplied the plug-in must be loaded dynamically.

The OSMF *MediaFactory* class is used to load plug-ins. For convenience, OSMF provides a *DefaultMediaFactory* class that contains the most commonly used *MediaFactoryItem* definitions. For details on OSMF and OSMF plug-ins, see the official OSMF page here.

The code snippet below shows how to load OSMF plug-ins either by the fully qualified namespace of the plug-in's *PluginInfo* class (i.e., "*com.akamai.osmf.plugins.MyPluginInfo*") for a static load, or by supplying an HTTP or FILE protocol URL for a dynamic load. In the example below "*mediaFactory"* can be any class derived from the OSMF *MediaFactory* class, but for most OSMF-built players, it will be an instance of the OSMF class, *DefaultMediaFactory*.

```
public function loadPlugin(source:String):void
{
    var pluginResource:MediaResourceBase;

    if (source.substr(0, 4) == "http" ||
     source.substr(0, 4) == "file")
    {
        // This is a URL, create a URLResource
        pluginResource = new URLResource(source);
    }
    else
    {
        // This is a class
        var pluginInfoRef:Class = flash.utils.getDefinitionByName(source)
            as Class;
        pluginResource = new PluginInfoResource(new pluginInfoRef);
    }

    loadPluginFromResource(pluginResource);
}

private function
loadPluginFromResource(pluginResource:MediaResourceBase):void
{
    setupListeners();
    mediaFactory.loadPlugin(pluginResource);
```

```
    function setupListeners(add:Boolean=true):void
    {
        if (add)
        {
            mediaFactory.addEventListener(
                MediaFactoryEvent.PLUGIN_LOAD,
                onPluginLoad);
            mediaFactory.addEventListener(
                MediaFactoryEvent.PLUGIN_LOAD_ERROR,
                onPluginLoadError);
        }
        else
        {
            mediaFactory.removeEventListener(
                MediaFactoryEvent.PLUGIN_LOAD,
                onPluginLoad);
            mediaFactory.removeEventListener(
                MediaFactoryEvent.PLUGIN_LOAD_ERROR,
                onPluginLoadError);
        }
    }

    function onPluginLoad(event:MediaFactoryEvent):void
    {
        trace("plugin loaded successfully.");
        setupListeners(false);
    }

    function onPluginLoadError(event:MediaFactoryEvent):void
    {
        trace("plugin failed to load.");
        setupListeners(false);
    }
}
```

Obtaining the Plug-ins

The Akamai Basic Streaming Plug-in

The source for this plug-in is provided with OSMF and can be loaded either dynamically or statically. In the OSMF distribution, look in this folder:

/plugins/samples/AkamaiBasicStreamingPlugin

The Akamai Advanced Streaming Plug-in

The source for this plug-in is not publicly available. The plug-in can be loaded dynamically from the locations outlined in the sections that follow.

Compiled for Flash Player 10.1

This URL will always hold the current version of the plug-in. You may link to this URL to dynamically load the plug-in:

http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-plugin/fp10.1/current/AkamaiAdvancedStreamingPlugin.swf

Important: *The plug-in swf at the above URL will change when new versions are released. Unless you always want to load the latest release, you should reference a version-specific URL.*

To refer to an explicit version, you may access it via the archive URL at (notice the "v1.1" in the link):

http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-plugin/fp10.1/v1.1/AkamaiAdvancedStreamingPlugin.swf

Compiled for Flash Player 10.0

This URL will always hold the most current version of the plug-in:

http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-plugin/fp10.0/current/AkamaiAdvancedStreamingPlugin.swf

Important: *The plug-in swf at the above URL will change when new versions are released. Unless you always want to load the latest release, you should reference a version-specific URL.*

To refer to an explicit version, you may access it via the archive URL at (notice the "v1.1" in the link):

http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-plugin/fp10.0/v1.1/AkamaiAdvancedStreamingPlugin.swf

Chapter 2. Playing Media

In This Chapter

Overview • 8
Using "MediaPlayerSprite" to Play Media • 9
Using "MediaContainerUIComponent" to Play Media • 10

Overview

To use plug-ins in your OSMF-built player, you must use the OSMF *DefaultMediaFactory* class or another *MediaFactory* subclass, rather than instantiating media elements directly. OSMF uses *MediaFactory* to interact with plug-ins. OSMF plug-ins are required to provide a media description of type *MediaFactoryItem*, which has a *canHandleResource* function that identifies the resource types the plug-in can handle.

When the OSMF *MediaFactory* class or a derived class loads a plug-in, it adds the *MediaFactoryItem* objects the plug-in declares to its internal collection. The *MediaFactoryItem* objects that don't have "org.osmf" in their ID get the first shot at creating media elements. Further, the first plug-in loaded gets the first shot. This is something that player developers need to be aware of if they are loading multiple OSMF plug-ins that create *MediaElement* objects. For info on how to resolve these conflicts, see Chapter 4. Special Use Cases..

Once the plug-in is loaded, the player can call *createMediaElement* on the same *MediaFactory* instance that loaded the plug-in. For example:

```
var resource:URLResource = new StreamingURLResource(source,
   StreamType.LIVE_OR_RECORDED);
var mediaElement:MediaElement=mediaFactory.createMediaElement(resource);
```

Once the *MediaElement* is created, it can be assigned to the *MediaPlayer* controller class. The *MediaPlayer* class is a non-UI controller class. It needs to be attached to a display object.

For media playback, developers have a few choices with OSMF:

1. ***MediaPlayerSprite*** – a pure ActionScript class that "wraps" the capabilities of *MediaPlayer, MediaContainer*, and *MediaFactory*.
2. ***MediaContainerUIComponent*** – a Flex UI component that "wraps" *MediaContainer*.
3. Work directly with the low-level OSMF classes such as *MediaContainer*.

The first two options are detailed below. For more information on the third option, visit the Strobe Media Playback site, and inspect the source code for that player.

Using "MediaPlayerSprite" to Play Media

The *MediaPlayerSprite* class in OSMF provides the capabilities of the *MediaPlayer, MediaContainer,* and *MediaFactory* classes all in one Sprite-based class.

The sample below shows how to use the *MediaPlayerSprite* class in an ActionScript project. Notice the sample could have used the set *resource* property and the built-in *MediaFactory* in the *MediaPlayerSprite* class, but the sample shows the flexibility of OSMF by instantiating and using its own media factory instance.

```
[SWF(backgroundColor="0x000000", frameRate="25", width="640",
height="360")]
public class Sample extends Sprite
{
    private var mediaPlayerSprite:MediaPlayerSprite;
    private var mediaFactory:DefaultMediaFactory;

    // Content
    private static const HDN_MULTI_BITRATE_VOD:String =
    "http://mediapm.edgesuite.net/edgeflash/public/debug/assets/smil/
    seas2.smil";
    // Plugins
    private static const AKAMAI_PLUGIN:String =
    "http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-
    plugin/fp10.1/current/AkamaiAdvancedStreamingPlugin.swf";

    public function Sample()
    {
        super();

        stage.scaleMode = StageScaleMode.NO_SCALE;
        stage.align = StageAlign.TOP_LEFT;

        mediaPlayerSprite = new MediaPlayerSprite();
        addChild(mediaPlayerSprite);

        mediaPlayerSprite.scaleMode = ScaleMode.NONE;
        mediaPlayerSprite.width = stage.stageWidth;
        mediaPlayerSprite.height = stage.stageHeight;
        mediaPlayerSprite.mediaPlayer.addEventListener(
            MediaErrorEvent.MEDIA_ERROR,
            onMediaError);

        stage.addEventListener(Event.RESIZE, onStageResize);

        mediaFactory = new DefaultMediaFactory();

        loadPlugin(AKAMAI_PLUGIN);
    }
```

Using "MediaContainerUIComponent" to Play Media

MediaContainerUIComponent is a sample class distributed with OSMF that extends *UIComponent* in the Flex SDK and serves as a wrapper for the *MediaContainer* class in OSMF.

To use this component you will need to first import the *MediaContainerUIComponent* project into Flash Builder:

1. Select **"File >> Import >> Flash Builder Project…"** and browse to the OSMF source folder. You'll find the *MediaContainerUIComponent* project in the following directory:

   /apps/samples/framework/MediaContainerUIComponent

2. Right-click your Flash Builder player project, select **"Properties"**, then select **"Flex Build Path"** in the left column.
3. On the **"Library Path"** tab, click **"Add Project…"** and select the *MediaContainerUIComponent* project you just imported.
4. In your MXML file, add the namespace for the OSMF samples package, which *MediaContainerUIComponent* is a part of:

```
<?xml version="1.0" encoding="utf-8"?>
<s:VGroup xmlns:fx="http://ns.adobe.com/mxml/2009"
xmlns:s="library://ns.adobe.com/flex/spark"
       xmlns:mx="library://ns.adobe.com/flex/mx"
       xmlns:samples="org.osmf.samples.*"
creationComplete="init()" >
```

5. You can now instantiate the component with MXML:

```
<samples:MediaContainerUIComponent
   id="mediaContainerUIComponent"
   width="{MAX_VIDEO_WIDTH}"
   height="{MAX_VIDEO_HEIGHT}"  />
```

6. Next, in your *init* method that is called from *applicationComplete* or *creationComplete* (depending on whether you are working on an application or a component) add the following to create a new MediaContainer:

```
private function init():void
{
   mediaContainerUIComponent.container = new MediaContainer();
}
```

7. To set the media and begin playback (Where "value" in this case is a *MediaElement* object):

```
mediaContainerUIComponent.container.addMediaElement(value);
```

For a complete, working sample that uses the *MediaContainerUIComplete,* please see Chapter 5. Links to Samples with Source.

Chapter 3. Akamai Metadata

In This Chapter

Overview • 11
Attaching Resource Metadata • 11
Akamai Basic Streaming Plug-in Metadata • 12
Akamai Advanced Streaming Plug-in Metadata • 13

Overview

OSMF metadata is a very powerful feature. Key-value pairs, where keys are strings and values are arbitrary objects, can be attached to plug-in resources and *MediaElement* resources.

A plug-in resource is simply the URL to the plug-in SWF (or the plug-in's *PluginInfo* class if loading statically), such as the following example:

http://myserver/osmf/plugins/myplugin.swf

A *MediaElement* resource is a URL to a piece of media you intend to play in an OSMF player, such as rthe following:

rtmp://myserver/app/vod/mp4:myvideo.mp4

Attaching Resource Metadata

The sample code below shows how to attach metadata to a resource. In this sample, radio buttons in a player UI are inspected to determine if connection level or stream level auth tokens should be added to the resource metadata:

```
var resource:URLResource = new StreamingURLResource(url,
          StreamType.LIVE_OR_RECORDED);

var authMetadata:Metadata = new Metadata();

if (authType.selection.value == RB_CONNECT_AUTH_VALUE)
{
    authMetadata.addValue(
    AkamaiBasicStreamingPluginInfo.AKAMAI_METADATA_KEY_CONNECT_AUTH_PARAMS,
        authParams);
}
else if (authType.selection.value == RB_STREAM_AUTH_VALUE)
{
    authMetadata.addValue(
    AkamaiBasicStreamingPluginInfo.AKAMAI_METADATA_KEY_STREAM_AUTH_PARAMS,
        authParams);
}
resource.addMetadataValue(
    AkamaiBasicStreamingPluginInfo.AKAMAI_METADATA_NAMESPACE,
    authMetadata);
}
mediaElement = mediaFactory.createMediaElement(resource);
```

Akamai Basic Streaming Plug-in Metadata

The constant string values for the plug-in metadata can be found in the *AkamaiBasicStreamingPluginInfo* class since the source for this plug-in is provided with OSMF.

The metadata namespace is defined as:

```
/**
 * The namespace for setting/getting metadata on both the plug-in
 * resource and the media resource.
 **/
public static const AKAMAI_METADATA_NAMESPACE:String =
      "http://www.akamai.com/basicstreamingplugin/1.0";
```

Plug-in Resource Metadata

These values affect the behavior of the plug-in for all media the plug-in can handle (meaning "load" and "play"). These metadata keys and their respective values should be placed on the plug-in resource, not the media resource.

The table that follows consists of plug-in resource metadata keys and their descriptions:

Metadata Key	Description
AKAMAI_METADATA_KEY_RETRY_LIVE	For live streams, if the primary and secondary encoders stop, the plug-in will try to restart the stream every AKAMAI_METADATA_KEY_RETRY_INTERVAL seconds. This value overrides the default value of *true*. Valid values are *true* or *false* typed as a *Boolean*.
AKAMAI_METADATA_KEY_RETRY_INTERVAL	The live stream retry interval in seconds. This value overrides the default value of 15 seconds.
AKAMAI_METADATA_KEY_LIVE_TIMEOUT	The live stream time out in seconds. If the live stream does not start playing within this time, a "stream not found" error is dispatched. This value overrides the default value of 20 minutes.

Media Resource Metadata

These values can be applied to media resources. Following is a table of media resource metadata keys and their descriptions:

Metadata Key	Description
AKAMAI_METADATA_KEY_CONNECT_AUTH_PARAMS	The name-value pairs required for invoking connection authorization services on the Akamai network. Typically these include the "auth", "aifp" and "slist" parameters. These name-value pairs must be separated by a "&" and should not begin with a "?", "&" or "/". An example of a valid string would be: **auth=dxaEaxdNbCdQceb3aLd5a34hjkl3mabbydbbx-bfPxsv-b4toa-nmtE&aifp=babufp&slist=secure/babutest**
AKAMAI_METADATA_KEY_STREAM_AUTH_PARAMS	The name-value pairs required for invoking stream-level authorization services against streams on the Akamai network. Typically these include the "auth" and "aifp" parameters. These name-value pairs must be separated by a "&" and should not begin with a "?", "&" or "/". An example of a valid string would be: **auth=dxaEaxdNbCdQceb3aLd5a34hjkl3mabbydbbx-bfPxsv-b4toa-nmtE&aifp=babufp**

Akamai Advanced Streaming Plug-in Metadata

Since the source for the *Akamai Advanced Streaming Plug-in* is not provided, the string constants for the metadata keys are placed in a library (SWC) called *HDCore*. Rather than put string literals into your player code, you can use the string constants found in the library to ensure you are using the same strings the plug-in is using. A link for downloading the specific version of HDCore to be used with the plug-in can be found here:

http://mediapm.edgesuite.net/akamai-osmf-plugins/samples/index.html

Note: *See the ASDocs in the HDCore release bundle for the individual metadata keys available.*

All of the string constants and their descriptions for plug-in and resource metadata can be found in a class called *AkamaiStrings* in the *com.akamai.osmf.utils* package.

The metadata namespace is defined as follows:

```
/**
 * The namespace for getting/setting metadata on both the plug-in
 * resource and/or the media resource.
 **/
public static const AKAMAI_ADVANCED_STREAMING_PLUGIN_METADATA_NAMESPACE:String
       = "http://www.akamai.com/advancedstreamingplugin/1.0";
```

Chapter 4. Special Use Cases

In This Chapter

Resolving Plug-in Conflicts • 14

Playing Akamai HD Network 1.0 for Flash Content without using the Akamai SMIL File Format • 16

Resolving Plug-in Conflicts

It is possible for two or more loaded plug-ins to conflict, meaning the plug-in that should be creating media is not given a chance to do so because another plug-in is trying and failing before the correct plug-in can get a chance.

Note: *For details on OSMF plug-ins see the OSMF Plug-in Developers guide at http://osmf.org.*

The solution for resolving plug-in conflicts is a custom *MediaFactory* class that overrides the *resolveItems* method. A custom *MediaFactory* class has complete control over which plug-ins get a chance at media creation.

The *HDCore* library (SWC) contains a custom *MediaFactory* class called *AkamaiMediaFactory*. The overridden *resolveItems* method in *AkamaiMediaFactory* looks for resources with Akamai metadata attached, which identify the media type. If that metadata is missing, it bypasses all *MediaFactoryItem* objects containing an ID of "com.akamai.osmf" so that other plug-ins can get first shot at creating the media.

The following code sample loads two plug-ins, uses the *HDCore* library to detect and attach the Akamai resource metadata containing the media type, and then plays a SMIL file containing an OSMF serial composition of videos. Note if the *AkamaiMediaFactory* class were not used in this case, the Akamai plug-in would have tried to load and parse the SMIL resulting in a media load failure.

```
[SWF(backgroundColor="0x000000", frameRate="25", width="640",
height="360")]
public class PluginSample extends Sprite
{
    private var mediaPlayerSprite:MediaPlayerSprite;
    private var mediaFactory:AkamaiMediaFactory;

    // Content
    private static const SMIL_TEST_SERIAL:String      =
"http://mediapm.edgesuite.net/osmf/content/test/smil/seq-test.smil";

    // Plugins
    private static const AKAMAI_PLUGIN:String =
"http://players.edgesuite.net/flash/plugins/osmf/advanced-streaming-
plugin/fp10.1/current/AkamaiAdvancedStreamingPlugin.swf";
    private static const OSMF_SMIL_PLUGIN:String =
"http://mediapm.edgesuite.net/chuck/osmf/plugins/advanced-streaming-
plugin/samples/as3/sample1/SMILPlugin.swf";

    public function PluginSample()
    {
```

```
        super();

        stage.scaleMode = StageScaleMode.NO_SCALE;
        stage.align = StageAlign.TOP_LEFT;

        mediaPlayerSprite = new MediaPlayerSprite();
        addChild(mediaPlayerSprite);

        mediaPlayerSprite.scaleMode = ScaleMode.NONE;
        mediaPlayerSprite.width = stage.stageWidth;
        mediaPlayerSprite.height = stage.stageHeight;
        mediaPlayerSprite.mediaPlayer.addEventListener(
                    MediaErrorEvent.MEDIA_ERROR,
                    onMediaError);
        mediaPlayerSprite.mediaPlayer.addEventListener(
                    PlayEvent.PLAY_STATE_CHANGE,
                    onPlayStateChange);

        stage.addEventListener(Event.RESIZE, onStageResize);

        mediaFactory = new AkamaiMediaFactory();

        mediaFactory.loadPluginFromString(AKAMAI_PLUGIN);
        mediaFactory.loadPluginFromString(OSMF_SMIL_PLUGIN);
        mediaFactory.resolveAndSetMedia(
                    mediaPlayerSprite.mediaPlayer,
                    SMIL_TEST_SERIAL);
    }

    private function onPlayStateChange(event:PlayEvent):void
    {
        trace(">>>> onPlayStateChange() - event.playState="+
              event.playState);
    }

    private function onMediaError(event:MediaErrorEvent):void
    {
        trace("!!!!! OSMF Media Error : "+event.error);
    }

    private function onStageResize(event:Event):void
    {
        mediaPlayerSprite.width = stage.stageWidth;
        mediaPlayerSprite.height = stage.stageHeight;
    }
}
```

Playing Akamai HD Network 1.0 for Flash Content without using the Akamai SMIL File Format

The easiest way to play all content over the Akamai network is to use the *Akamai Advanced Streaming Plug-in* and simply give it a media resource. The plug-in will determine what the resource is pointing at and take the correct steps to play the media. However, there may be cases where player developers cannot use the Akamai SMIL file format.

In this case, the player developer will need to manually add resource metadata before trying to play the media.

Here is some sample code showing how to play a live Akamai HD Network for Flash stream without using the Akamai SMIL format:

```
private function playMedia():void
{
    // It doesn't matter in this case what the resource is,
    // it just needs to use the HTTP protocol and have a .smil extension
    var resource:URLResource =
          new URLResource("http://www.example.com/example.smil");
    var hdMBRObject:HDMBRObject = new HDMBRObject();

    // LIVE sample
    hdMBRObject.httpBase = "http://live.edgeflash.akamai.com.edgesuite.net/";
    hdMBRObject.addStream("akamai247_2_800@13903", 800); // Bitrates are in kbps
    hdMBRObject.addStream("akamai247_2_1600@13903", 1600);

    // VOD sample - same as above, just add this line
    //hdMBRObject = HDVODFormatter.format(hdMBRObject);

    var hdMBRObjects:Array = new Array();
    hdMBRObjects.push(hdMBRObject);

    AkamaiMediaUtils.addAkamaiMediaTypeToResource(resource,
              AkamaiStrings.AKAMAI_MEDIA_TYPE_HDN_MBR,
              hdMBRObjects);
    mediaPlayerSprite.media = mediaFactory.createMediaElement(resource);
}
```

Chapter 5. Links to Samples with Source

Samples will be added and updated on a regular basis. For pure ActionScript 3 samples, their source, and any other plug-in samples, including using the AkamaiAdvancedStreamingPlugin with Strobe Media Playback, please see this page:

http://mediapm.edgesuite.net/akamai-osmf-plugins/samples/index.html