

ADOBE® ACTIONSCRIPT® 3.0

Dynamic Stream API

© 2009 Adobe Systems Incorporated. All rights reserved.

Adobe® ActionScript® 3.0 Dynamic Stream API

If this guide is distributed with software that includes an end user agreement, this guide, as well as the software described in it, is furnished under license and may be used or copied only in accordance with the terms of such license. Except as permitted by any such license, no part of this guide may be reproduced, stored in a retrieval system, or transmitted, in any form or by any means, electronic, mechanical, recording, or otherwise, without the prior written permission of Adobe Systems Incorporated. Please note that the content in this guide is protected under copyright law even if it is not distributed with software that includes an end user license agreement.

The content of this guide is furnished for informational use only, is subject to change without notice, and should not be construed as a commitment by Adobe Systems Incorporated. Adobe Systems Incorporated assumes no responsibility or liability for any errors or inaccuracies that may appear in the informational content contained in this guide.

Please remember that existing artwork or images that you may want to include in your project may be protected under copyright law. The unauthorized incorporation of such material into your new work could be a violation of the rights of the copyright owner. Please be sure to obtain any permission required from the copyright owner.

Any references to company names in sample templates are for demonstration purposes only and are not intended to refer to any actual organization.

Adobe, the Adobe logo, Adobe AIR, ActionScript, Flash, and Flex are either registered trademarks or trademarks of Adobe Systems Incorporated in the United States and/or other countries. All other trademarks are the property of their respective owners.

This work is licensed under the Creative Commons Attribution Non-Commercial 3.0 License. To view a copy of this license, visit <http://creativecommons.org/licenses/by-nc/3.0/us/>

This product includes software developed by the Apache Software Foundation (<http://www.apache.org/>).

MPEG Layer-3 audio compression technology licensed by Fraunhofer IIS and Thomson Multimedia (<http://www.mp3licensing.com>).

Speech compression and decompression technology licensed from Nellymoser, Inc. (www.nellymoser.com).

Video compression and decompression is powered by On2 TrueMotion video technology. © 1992-2005 On2 Technologies, Inc. All Rights Reserved. <http://www.on2.com>.

This product includes software developed by the OpenSymphony Group (<http://www.opensymphony.com/>).

This product contains either BSAFE and/or TIPEM software by RSA Security, Inc.

Sorenson Sorenson Spark™ video compression and decompression technology licensed from Sorenson Media, Inc.
Spark Incorporated, 345 Park Avenue, San Jose, California 95110, USA.

Notice to U.S. Government End Users. The Software and Documentation are "Commercial Items," as that term is defined at 48 C.F.R. §2.101, consisting of "Commercial Computer Software" and "Commercial Computer Software Documentation," as such terms are used in 48 C.F.R. §12.212 or 48 C.F.R. §227.7202, as applicable. Consistent with 48 C.F.R. §12.212 or 48 C.F.R. §§227.7202-1 through 227.7202-4, as applicable, the Commercial Computer Software and Commercial Computer Software Documentation are being licensed to U.S. Government end users (a) only as Commercial Items and (b) with only those rights as are granted to all other end users pursuant to the terms and conditions herein. Unpublished-rights reserved under the copyright laws of the United States. Adobe Systems Incorporated, 345 Park Avenue, San Jose, CA 95110-2704, USA. For U.S. Government End Users, Adobe agrees to comply with all applicable equal opportunity laws including, if appropriate, the provisions of Executive Order 11246, as amended, Section 402 of the Vietnam Era Veterans Readjustment Assistance Act of 1974 (38 USC 4212), and Section 503 of the Rehabilitation Act of 1973, as amended, and the regulations at 41 CFR Parts 60-1 through 60-60, 60-250, and 60-741. The affirmative action clause and regulations contained in the preceding sentence shall be incorporated by reference.

Contents

Adobe ActionScript 3.0 Dynamic Stream API

DynamicStream class	1
DynamicStreamItem class	8

Adobe ActionScript 3.0 Dynamic Stream API

The Dynamic Stream API includes two classes: `DynamicStream` and `DynamicStreamItem`. Use these classes to develop applications that switch between media streams encoded at different bit rates when network conditions change. Adobe® Flash® Media Server switches between streams seamlessly. Dynamic streaming provides a continuous playback experience for all viewers, regardless of their computing power or internet connection speed.

You can develop dynamic streaming applications in Adobe Flash and Adobe Flex™ for the Adobe Flash Player 10 and Adobe AIR™ 1.5 runtimes. Dynamic streaming applications require any edition of Adobe Flash Media Server 3.5.

For more information on dynamic streaming, see the following articles:

- “Understanding the Adobe ActionScript 3.0 Dynamic Stream API” at www.adobe.com/go/fms_dynstream_advanced
- “Encoding Best Practices for Dynamic Streaming” at www.adobe.com/go/fms_dynstream_bestpractices
- “Dynamic Streaming” in the *Adobe Flash Media Server 3.5 Developer Guide* at www.adobe.com/go/learn_fms_dynstream_en

DynamicStream class

Class `public class DynamicStream`

Inheritance `DynamicStream -> NetStream -> EventDispatcher -> Object`

Language version ActionScript 3.0

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Use the `DynamicStream` class to manage a set of content streams. The typical use case is to switch between streams encoded at different bit rates when network conditions change. Dynamic stream switching provides clients the best viewing experience seamlessly.

The `DynamicStream` class inherits from the `NetStream` class. It adds a `startPlay()` method that you use to pass in an array of streams encoded at different bit rates. The class contains algorithms that use the Quality of Service data available in Flash Player 10 to determine when to switch streams.

To use dynamic streaming with live streams, set the `DynamicStreamItem.start` property to `-1`. To use dynamic streaming with recorded (vod) streams, use the default value of `-2` or a value greater than or equal to `0`.

Workflow

Use the following workflow to add dynamic streaming to an existing application. The workflow assumes that the application has a class file that contains `NetConnection` code for connecting to Flash Media Server. It also assumes that you have several streams of the same content encoded at various bit rates.

- 1 Place the `DynamicStream.as` and `DynamicStreamItem.as` class files in the same directory as your application class file.

- 2 In your application class file, create a `DynamicStream` object and pass it the `NetConnection` object for this connection:

```
var ds:DynamicStream = new DynamicStream(nc);
```

- 3 Create a `DynamicStreamItem` object:

```
var dsi:DynamicStreamItem = new DynamicStreamItem();
```

- 4 Call the `addStream()` method to add streams encoded at various bit rates to the `DynamicStreamItem` object:

```
dsi.addStream("mp4:test_800k.mov", 800);  
dsi.addStream("mp4:test_1000k.mov", 1000);  
dsi.addStream("mp4:test_2000k.mov", 2000);
```

- 5 Call the `startPlay()` method and pass it the `DynamicStreamItem` object:

```
ds.startPlay(dsi);
```

The complete code sample is as follows:

```
var ds:DynamicStream = new DynamicStream(nc);  
var dsi:DynamicStreamItem = new DynamicStreamItem();  
  
// add a new stream/bitrate pair  
dsi.addStream("mp4:test_800k.mov", 800);  
dsi.addStream("mp4:test_1000k.mov", 1000);  
dsi.addStream("mp4:test_2000k.mov", 2000);  
  
// start playing  
ds.startPlay(dsi);
```

DynamicStream()

```
public function DynamicStream(nc:NetConnection)
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Constructor function. Creates a stream that you can use to switch between media files encoded at various bit rates.

Parameters

nc:NetConnection The `NetConnection` object for this connection.

Example

```
var nc:NetConnection = new NetConnection();  
nc.connect("rtmp://fmsexamples.adobe.com/sampleapp");  
var ds:DynamicStream = new DynamicStream(nc);
```

aggressiveModeBufferLength

```
aggressiveModeBufferLength:Number [read/write]
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The lowest acceptable buffer length, in seconds. The default value is 4.

When this value is reached, the server switches to a stream with the lowest possible bit rate. Set this property to prevent the buffer from emptying. An empty buffer can cause a pause or stutter in streaming media.

This property is not used for live streams.

Example

```
ds.aggressiveModeBufferLength = 6;
```

currentStreamBitRate

currentStreamBitRate:Number [read-only]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The bit rate, in kbps, of the stream currently playing. The server does not calculate the bit rate of the stream. The value of the `currentStreamBitRate` property is the `bitRate` argument you passed to the `DynamicStreamItem.addStream()` method.

Example

```
currBitRate = ds.currentStreamBitRate;
```

currentStreamName

currentStreamName:String [read-only]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Returns the name of the stream that is playing. The value of the `currentStreamName` property is the `streamName` argument you passed to the `DynamicStreamItem.addStream()` method.

Example

```
currStreamName = ds.currentStreamName;
```

droppedFramesLockDelay

droppedFramesLockDelay:Number [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The delay, in seconds, the class implements when it encounters dropped frames. During this delay, the server does not switch streams. The default value is 300 seconds (5 mins).

When a stream drops more than 25% of its frames, the class switches to a lower bitrate stream. The class does not switch back to the bitrate that was dropping frames for 5 mins, or the value of `droppedFramesLockDelay`. After the delay, the class unlocks that bitrate.

Example

```
ds.droppedFramesLockDelay = 400;
```

manualSwitchMode()

```
public function manualSwitchMode(mode:Boolean) : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Toggles manual switching in the DynamicStream class. When set to `true`, you can use the methods `switchToStreamName()`, `switchToStreamRate()`, `switchUp()`, and `switchDown()`. When set to `false`, the methods are not available.

It is most common to use automatic switch mode. Use manual switch mode if you want to demonstrate how to explicitly control the stream rates.

Parameters

mode:Boolean `true` to enable manual switching; `false` to disable it.

Example

```
public function switchMode():void {  
    if(shiftMode.selected){  
        ds.manualSwitchMode(true);  
        shiftMode.label = "Switch mode - Manual";  
        upShift.visible = true;  
        downShift.visible = true;  
    } else if(!shiftMode.selected){  
        ds.manualSwitchMode(false);  
        shiftMode.label = "Switch mode - Automatic";  
        upShift.visible = false;  
        downShift.visible = false;  
    }  
}
```

maxBandwidth

```
maxBandwidth:Number [read-only]
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Returns the maximum bandwidth capacity of the stream that is playing in kbps. This property measures client bandwidth, not server bandwidth. It changes depending on conditions to which the client is exposed.

Example

```
mBand = ds.maxBandwidth;
```

preferredBufferLength

```
preferredBufferLength:Number [read/write]
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The length of the buffer, in seconds, after a stream begins to play. The default value for recorded video is 8. The default value for live video is 10.

The server uses the value of `startBufferLength` to buffer the stream before it begins to play. Once the stream begins playing, the server switches to the value of `preferredBufferLength`. Set this value to a higher value than `startBufferLength`, such as 30-60 seconds.

Example

```
ds.preferredBufferLength = 2;
```

setBandwidthLimit()

```
public function setBandwidthLimit(limit:Number) : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The maximum bandwidth stream a client can play. For example, call this function to limit a client to a 1000 kbps stream even though there is a stream of 2000 kbps and the client has more than 2 Mbps of available bandwidth.

Parameters

limit:Number The bandwidth limit in kilobits per second (kbps). The default value of -1 sets the bandwidth to unlimited.

Example

```
ds.setBandwidthLimit(1000);
```

startBufferLength

```
startBufferLength:Number [read/write]
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The length of the buffer, in seconds, before a stream begins to play. The default value is 2. The server uses this property to buffer the stream before it plays. When the stream begins playing, the server switches to the value of the `preferredBufferLength` property.

Set this property to a low value for a quick start. Set this property high enough to let the server compute the maximum bandwidth available to the stream.

Example

```
ds.startBufferLength = 1;
```

startPlay()

```
public function startPlay(dsi:DynamicStreamItem) : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Plays the streams passed to the `DynamicStreamItem.addStream()` method.

Parameters

dsi:DynamicStreamItem The object that contains the streams to switch between.

Example

```
// Create a DynamicStreamItem object
var dsi:DynamicStreamItem = new DynamicStreamItem();
// Add stream/bitrate pairs
dsi.addStream("mp4:sample1_150kbps.f4v", 150);
dsi.addStream("mp4:sample1_500kbps.f4v", 500);
dsi.addStream("mp4:sample1_700kbps.f4v", 700);
dsi.addStream("mp4:sample1_1000kbps.f4v", 1000);
dsi.addStream("mp4:sample1_1500kbps.f4v", 1500);
ds.preferredBufferLength = 2;
ds.startBufferLength = 0;
// Play stream or append stream to the playlist
ds.startPlay(dsi);
```

switchDown()

```
public function switchDown() : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Switches to the stream in the [streams](#) array with the next lowest bitrate. If the stream encoded at the lowest bitrate is playing, the server ignores this command. This command works in manual switch mode only. See [manualSwitchMode\(\)](#).

The `DynamicStreamItem` class passes in an array of video streams sorted by bitrate. When you switch up or down, it moves the position up or down in that array and tells the server to switch.

Example

The following MXML code creates buttons that manually switch the stream bitrate up and down:

```
<mx:Button visible="true" x="450" y="50" label="Shift up" click="ds.switchUp()" id="upShift"/>
<mx:Button visible="true" x="525" y="50" label="Shift down" click="ds.switchDown()"
id="downShift"/>
```

switchToStreamName()

```
public function switchToStreamName(name:String) : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Searches for and switches to the specified stream name in the [streams](#) array. This command works in manual switch mode only. See [manualSwitchMode\(\)](#).

Call this function to switch to a stream by name instead of by array position. This function lets you skip multiple up or down switches.

Parameters

name:String The name of an object in the [streams](#) array. If no match is found, the command is ignored.

Example

```
ds.switchToStreamName(streamName);
```

switchToStreamRate()

```
public function switchToStreamRate(rate:int) : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Switches to the specified rate or the highest rate that does not exceed the rate requested in the `streams` array. This command works in manual switch mode only. See `manualSwitchMode()`.

Call this function to switch to a stream by rate, instead of by array position. This function lets you skip multiple up or down switches.

Parameters

rate:int The rate of an object in the `streams` array. If no match is found, the highest rate that does not exceed that value is used.

Example

```
ds.switchToStreamRate(streamRate);
```

switchQOSTimerDelay

```
switchQOSTimerDelay:Number(default=4) [read/write]
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The frequency, in seconds, with which the stream checks its performance. The default value is 4 seconds.

By default, the client reviews Quality of Service statistics every 4 seconds. The client uses the statistics to determine whether to switch to a stream with a higher or lower bit rate. The Quality of Service statistics are available through the `NetStreamInfo` class.

Example

```
ds.switchQOSTimerDelay(6);
```

switchUp()

```
public function switchUp() : void
```

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Switches to the stream in the `streams` array with the next highest bitrate. If the stream encoded at the highest bitrate is playing, the server ignores this command. This command works in manual switch mode only. See `manualSwitchMode()`.

Example

The following MXML code creates buttons that manually switch the stream bitrate up and down:

```
<mx:Button visible="true" x="450" y="50" label="Shift up" click="ds.switchUp()" id="upShift"/>
<mx:Button visible="true" x="525" y="50" label="Shift down" click="ds.switchDown()"
id="downShift"/>
```

DynamicStreamItem class

Class `public class DynamicStreamItem`

Inheritance `DynamicStreamItem -> Object`

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Contains methods and properties that let you manage details about the streams in an application that uses dynamic streaming. A `DynamicStreamItem` object contains items that are stream/bit rate pairs. Pass the `DynamicStreamItem` object to the `DynamicStream.startPlay()` method. You can also use the `DynamicStreamItem` class to specify playback properties that take effect when you call the `startPlay()` method.

DynamicStreamItem()

`public function DynamicStreamItem()`

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Constructor function. Creates an object of stream names and bit rates (and optionally, other properties) to pass to a `DynamicStream` object.

Example

The following code creates a `DynamicStreamItem` object:

```
var dsi:DynamicStreamItem = new DynamicStreamItem();
```

addStream()

`public function addStream(streamName:String, bitRate:Number) : void`

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Adds a stream name and bitrate pair to the `DynamicStreamItem` object.

Parameters

streamName:String The name of the stream. This method does not know the encoding format of the stream. Specify the format correctly in the stream name to ensure the server can play the file:

- For H.264/AAC files, prefix the stream name with `mp4:`, for example, `"mp4:myStream_150k.mp4"` or `"mp4:myF4vStream.f4v"`.

If the file on the server uses a filename extension, specify it.

- For FLV files, do not use a prefix or filename extension, for example, `"myStream_150k"`.

bitRate:Number The bit rate of the stream in kilobits per second. The application uses this value to determine which higher or lower bit-rate stream to switch to. Specify the value at which the stream was encoded. The `addStream()` method does not inspect the stream to verify that this value is accurate.

Example

The following code adds streams to the `DynamicStreamItem` object:

```
dsi.addStream("mp4:sample1_150kbps.f4v", 150);  
dsi.addStream("mp4:sample1_500kbps.f4v", 500);  
dsi.addStream("mp4:sample1_700kbps.f4v", 700);  
dsi.addStream("mp4:sample1_1000kbps.f4v", 1000);  
dsi.addStream("mp4:sample1_1500kbps.f4v", 1500);
```

len

len:Number (default = -1) [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The length of playback of the stream, in seconds. The default value is -1, which plays the entire stream. A value of 0 plays a single frame. Any value greater than 0 plays the stream for that number of seconds.

Example

```
dsi.len = 10;
```

reset

reset:Boolean (default = true) [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Clears any previous play calls and plays the specified stream immediately. The default value is true.

This value is converted into a `NetStreamPlayTransitions` value. Use this property to create playlists with the `DynamicStream` class.

Example

```
dsi.reset = false;
```

start

start:Number (default = 0) [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The time, in seconds, at which the stream begins playing.

- The default value is 0. The application attempts to play a recorded stream with the name specified in the `streamName` argument passed to `DynamicStreamItem.addStream()`.
- A value of -1 attempts to play a live stream with the name specified in the `streamName` argument passed to `DynamicStreamItem.addStream()`.
- A value of 0 or greater plays a recorded stream with the name specified in the `streamName` argument passed to `DynamicStreamItem.addStream()`, at the start position specified.

Example

```
dsi.start = 30;
```

startRate

`startRate:int` (default = -1) [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The bitrate value of the stream to play first. The default value is -1. The default value plays the lowest bitrate stream and increments to the highest bitrate that plays smoothly. To start playback at a higher bitrate, set this value to one of the `bitRate` arguments you passed to the `addStream()` method.

Example

```
dsi.startRate = 700;
```

streamCount

`streamCount:int` [read/write]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

Overrides the length of the array contained in the `streams` property. If you do not specify a value, the player calculates the value when you call the `DynamicStream.startPlay(dsi:DynamicStreamItem)` method.

Example

```
dsi.streamCount = 3;
```

streams

`streams:Array` [read-only]

Runtime versions AIR 1.5, Flash Player 10

Server version Flash Media Server 3.5

The array of objects passed to the `addStream()` method. Each object has a `name` and `rate` property that contain the `streamName` and `bitRate` arguments passed to the `addStream()` method. The array is sorted by `bitRate`.

Example

```
firstStreamName = dsi.streams[0].streamName;
```

