

INTRODUCTION TO FLASH® LITE™ 2.x AND 3.0 ACTIONSCRIPT™

© 2007 Adobe Systems Incorporated. All rights reserved.

Introduction to Flash® Lite™ 2.x and 3.0 ActionScript™

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, ActionScript, Flash, Flash Lite, and Macromedia 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.



Sorenson Spark™ video compression and decompression technology licensed from Sorenson Media, Inc.

Fraunhofer-IIS/Thomson Multimedia: MPEG Layer-3 audio compression technology licensed by Fraunhofer IIS and Thomson Multimedia (<http://www.iis.fhg.de/amm/>).

Independent JPEG Group: This software is based in part on the work of the Independent JPEG Group.

Nellymoser, Inc.: Speech compression and decompression technology licensed by Nellymoser, Inc. (<http://www.nelly-moser.com>).

Opera® browser Copyright © 1995-2002 Opera Software ASA and its suppliers. All rights reserved.

Macromedia Flash 8 video is powered by On2 TrueMotion video technology. © 1992-2005 On2 Technologies, Inc. All Rights Reserved. <http://www.on2.com>.

Visual SourceSafe is a registered trademark or trademark of Microsoft Corporation in the United States and/or other countries.

Updated Information/Additional Third Party Code Information available at <http://www.adobe.com/go/thirdparty/>.

Adobe Systems 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

Introduction	5
Chapter 1: Unsupported and partially supported ActionScript elements	7
Commands issued through fsCommand and fsCommand2	7
Global properties	10
Chapter 2: Unsupported and partially supported classes	11
Button class	12
Date class	14
Key class	15
Mouse class	16
MovieClip class	18
NetConnection class	21
NetStream class	21
Selection class	22
SharedObject class	23
System class	24
capabilities (System.capabilities) class	25
Sound class	28
Stage class	30
TextField class	31
TextFormat class	32
Video class	34

Introduction

Macromedia® Flash® Lite™ 1.0 and Macromedia Flash Lite 1.1 software from Adobe®, the first versions of Flash Lite, are based on Macromedia® Flash® Player 4 software from Adobe.

Macromedia® Flash Lite 2.0 and Macromedia Flash Lite 2.1 software from Adobe, along with Adobe® Flash® Lite™ 3.0, are based on Macromedia Flash Player 7 from Adobe, but differ from it in the following respects:

- Flash Lite supports some features only partially.
- Flash Lite adds some features specifically for mobile devices.

This document describes the differences between the ActionScript™ supported for Flash Lite 2.0 and 2.1 (which are referenced collectively as 2.x) and the ActionScript that was supported for Flash Player 7. It also describes the differences between the ActionScript supported in Flash Lite 2.x and Flash Lite 3.0. Wherever applicable, this document includes cross-references to the following documents that provide additional details about the various classes and how to use them when writing Flash Lite applications:

- *Flash Lite 2.x and 3.0 ActionScript Language Reference*
- *Developing Flash Lite 2.x and 3.0 Applications*
- *Learning ActionScript 2.0 in Adobe Flash*

Unsupported and partially supported ActionScript elements

This chapter describes the global functions and properties, constants, operators, statements, extensions, and keywords that are either partially supported or not supported by ActionScript for Macromedia® Flash® Lite™ 2.x or 3.0 software from Adobe.

Commands issued through <code>fsCommand</code> and <code>fsCommand2</code>	7
Global properties	10

Commands issued through `fsCommand` and `fsCommand2`

The `fsCommand()` and `fsCommand2()` global functions let the SWF file communicate with either Flash Player or the program that is hosting Flash Player, such as a web browser.

Flash Lite 2.x modifies the standard Macromedia Flash Player 7 commands, and adds commands that are specific to embedded devices. Flash Lite 2.x also supports the `fsCommand2()` function, which provides similar functionality to `fsCommand()`, with the exception that the command is executed immediately, not deferred until after the calling frame is processed.

For more information on the `fsCommand2()` function, see the *Flash Lite 2.x and 3.0 ActionScript Language Reference*.

Unsupported commands

The following table lists the commands that are not supported by `fsCommand()` when using ActionScript 2.0 to create Flash Lite content.

Command	Description
<code>quit</code>	Closes the projector.
<code>fullscreen</code>	Specifying <code>true</code> sets Flash Player to full-screen mode. Specifying <code>false</code> returns the player to normal menu view.
<code>allowscale</code>	Specifying <code>false</code> sets the player so the SWF file is always drawn at its original size and is never scaled. Specifying <code>true</code> forces the SWF file to scale to 100% of the screen.
<code>showmenu</code>	Specifying <code>true</code> enables the full set of context menu items. Specifying <code>false</code> hides all the context menu items except About Flash Player and Settings.
<code>exec</code>	Executes an application from within the projector.
<code>trapallkeys</code>	Specifying <code>true</code> sends all key events, including accelerator keys, to the <code>onClipEvent(keyDown/keyUp)</code> handler in Flash Player.

Commands summary

The following table lists the ActionScript commands, functions, and keywords that are partially or not supported by Flash Lite 2.x and later.

Command, function, or keyword	Description	Support
<code>asfunction</code>	A protocol for URLs in HTML text fields.	Not supported
<code>fscommand()</code>	Function that lets the SWF file communicate with either Flash Player or the program hosting Flash Player, such as a web browser.	Partially supported
<code>on</code>	Prefix for events to execute when the event occurs. Limitation: The supported events are <code>rollout</code> , <code>rollover</code> , and <code>keyPress</code> .	Partially supported

Command, function, or keyword	Description	Support
<code>onClipEvent</code>	Event handler; triggers actions defined for a specific instance of a movie clip. Limitations: Supported events are <code>press</code> , <code>load</code> , <code>unload</code> , <code>enterFrame</code> , <code>keyDown</code> , <code>keyup</code> , and <code>data</code> . The <code>mouseDown</code> , <code>mouseUp</code> , and <code>mouseMove</code> events are supported if either <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to <code>true</code> .	Partially supported
<code>onUpdate</code>	The <code>onUpdate</code> event handler is defined for a live preview used with a component.	Not supported
<code>print()</code>	The <code>print()</code> function targets the movie clip according to the boundaries specified in the parameter.	Not supported
<code>printAsBitmap()</code>	A function that prints the target movie clip as a bitmap.	Not supported
<code>printAsBitmaNum()</code>	A function that prints a level in Flash Player as a bitmap according to the boundaries specified in the parameter (<code>bmovie</code> , <code>bmax</code> , or <code>bframe</code>).	Not supported
<code>printNum()</code>	A function that prints the level in Flash Player according to the boundaries specified in the <code>boundingBox</code> parameter (<code>bmovie</code> , <code>bmax</code> , or <code>bframe</code>).	Not supported
<code>startDrag()</code>	A function that makes the target movie clip draggable while the SWF is playing. Only one movie clip can be dragged at a time. Limitation: Supported if mouse or stylus interface is supported.	Partially supported
<code>stopDrag()</code>	A function that stops the current drag operation. Limitation: Supported if mouse or stylus interface is supported.	Partially supported
<code>updateAfterEvent()</code>	A function that updates the display (independent of the frames per second set for the SWF file) when you call it in an <code>onClipEvent()</code> handler or as part of a function or method that you pass to <code>setInterval()</code> .	Not supported

Global properties

The following table lists the ActionScript global properties that are partially supported or not supported by Flash Lite 2.x and later.

Properties	Description	Support
<code>_droptarget</code>	Read-only property that returns the absolute path in slash (/) syntax notation of the movie clip instance on which <code>draggableInstanceName</code> (the name of a movie clip instance that was the target of a <code>startDrag()</code> function) was dropped. This property always returns a path that starts with a slash (/).	Not supported
<code>_highquality</code>	Global property that specifies the level of anti-aliasing applied to the current SWF file. This property can also control bitmap smoothing. Limitation: Flash Lite does not support bitmap smoothing.	Partially supported
<code>_url</code>	Read-only property that retrieves the URL of the SWF file from which the movie clip was downloaded.	Not supported

Unsupported and partially supported classes

This chapter describes ActionScript 2.0 classes that are either partially or not supported by Adobe's Macromedia Flash Lite 2.0. It also describes the extensions that are specific to ActionScript for Flash Lite 2.x and 3.0.

Button class	12
Date class	14
Key class	15
Mouse class	16
MovieClip class	18
NetConnection class	21
NetStream class	21
Selection class	22
SharedObject class	23
System class	24
capabilities (System.capabilities) class	25
Sound class	28
Stage class	30
TextField class	31
TextFormat class	32
Video class	34

Button class

All button symbols in a SWF file are instances of the Button object. The Button class provides methods, properties, and event handlers for working with buttons.

For more information about the Button class, see the following:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 10, “Handling Events,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Property summary

The following table lists the properties that are either partially or not supported by the Button class when using ActionScript for Flash Lite 2.x and 3.0.

Property	Description	Support
<code>menu</code>	An object that associates a ContextMenu object with a button.	Not supported
<code>trackAsMenu</code>	A Boolean value that indicates whether other buttons can receive mouse release events. Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>useHandCursor</code>	A Boolean value that indicates whether the pointer appears when the mouse passes over a button.	Not supported
<code>_xmouse</code>	Read-only; the x coordinate of the pointer, relative to a button instance. Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>_ymouse</code>	Read-only; the y coordinate of the pointer, relative to a button instance. Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported

Event summary

The following table lists the event handlers that are partially supported by the Button class when using ActionScript for Flash Lite 2.x and 3.0.

Event	Description
onDragOut	Invoked when the mouse button is clicked over the button and the pointer then dragged outside of the button. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
onDragOver	Invoked when the pointer is dragged over the button. Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
onReleaseOutside	Invoked when the mouse is released while the pointer is outside the button after the button is pressed while the pointer is inside the button. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.

Date class

The Date class shows how dates and times are represented in ActionScript, and it supports operations for manipulating dates and times. The Date class can also obtain the current date and time from the operating system.

For more information about the Date class, see the following:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 4, “Data and Data Types,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods (functions) that have been added to the Date class when using ActionScript for Flash Lite 2.x and 3.0.

Method	Description
<code>getLocaleLongDate()</code>	This function sets a parameter to a string that represents the current date, in long form, formatted according to the currently defined locale. The parameter is passed in by name. The returned value is a multiple-character, variable-length string. The actual formatting depends on the device and the locale.
<code>getLocaleShortDate()</code>	This function sets a parameter to a string that represents the current date, in abbreviated form, formatted according to the currently defined locale. The parameter is passed in by name. The returned value is a multiple-character, variable-length string. The actual formatting depends on the device and the locale.
<code>getLocaleTime()</code>	This function sets a parameter to a string that represents the current time, formatted according to the currently defined locale. The parameter is passed in by name. The returned value is a multiple-character, variable-length string. The actual formatting depends on the device and the locale.

Key class

The Key class provides methods and properties for obtaining information about the keyboard itself, and the keypresses input into it.

For more information about the Key class, see the following:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 14, “Creating Interaction with ActionScript,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Flash Lite property extensions

The following table lists the property that has been added to the Key class when using ActionScript for Flash Lite 2.x and later.

Property	Description	Support
<code>isToggled</code>	Returns <code>true</code> if the Num Lock or Caps Lock key is active.	Not supported

Flash Lite method extensions

The following table lists the method extension that has been added to the Key class when using ActionScript for Flash Lite 2.x and later.

Method	Description
<code>getCode()</code>	Returns the virtual key code of the last key pressed.

The Flash Lite 2.x and 3.0 implementation of the `getCode()` method returns a string or a number, depending on what the platform passed in. The only valid key codes are the standard key codes accepted by this class and the “special” key codes listed as properties of the `ExtendedKey` class. This restriction is enforced by the player. For valid key code values, see the Key class in the *Flash Lite 2.x and 3.0 ActionScript Language Reference*. This reference provides tables that map keys to codes for letters, numbers, the numeric keypad, function keys, special constant keys, and other keys.

Mouse class

The Mouse class lets you control the mouse in a SWF file; for example, this class lets you hide or show the mouse pointer.

For more information about the Mouse class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 14, “Creating Interaction with ActionScript,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods that are either partially or not supported by the Mouse class when using ActionScript for Flash Lite 2.x and later.

Method	Description	Support
<code>addListener()</code>	Registers an object to receive notifications of the <code>onMouseDown</code> , <code>onMouseMove</code> , <code>onMouseUp</code> , and <code>onMouseWheel</code> listeners. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>hide()</code>	Hides the pointer in a SWF file.	Not supported
<code>removeListener()</code>	Removes an object that was previously registered with <code>addListener()</code> . Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>show()</code>	Displays the pointer in a SWF file.	Not supported

Event summary

The following table lists the events that are either partially or not supported by the Mouse class when using ActionScript for Flash Lite 2.x and later.

Event	Description	Support
onMouseDown	Notified when the mouse is pressed. Limitations: Supported if System.capabilities.hasMouse or System.capabilities.hasStylus is set to true.	Partially supported
onMouseMove	Notified when the mouse moves. Limitations: Supported if System.capabilities.hasMouse is set to true.	Partially supported
onMouseUp	Notified when the mouse is released. Limitations: Supported if System.capabilities.hasMouse or System.capabilities.hasStylus is set to true.	Partially supported
onMouseWheel	Notified when the user rolls the mouse wheel.	Not supported

MovieClip class

The `MovieClip` class lets you use listener callback functions that provide status information while SWF or JPEG files load (download) into movie clips. To use `MovieClip` features, use `MovieClipLoader.loadClip()` instead of `loadMovie()` or `MovieClip.loadMovie()` to load SWF files.

For more information about the `MovieClip` class, see the following:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 11, “Working with Movie Clips,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods that are either partially or not supported by the `MovieClip` class when using ActionScript for Flash Lite 2.x and later.

Method	Description	Support
<code>attachAudio()</code>	Captures and plays local audio from the device’s microphone hardware.	Not supported
<code>getTextSnapshot()</code>	Returns a <code>TextSnapshot</code> object that contains the text in the static text fields in the specified movie clip.	Not supported
<code>startDrag()</code>	Specifies a movie clip as draggable and begins dragging the movie clip. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>stopDrag()</code>	Stops a <code>MovieClip.startDrag()</code> method. A movie clip that was made draggable with <code>startDrag()</code> remains draggable until a <code>stopDrag()</code> method is added, or until another movie clip becomes draggable. Only one movie clip is draggable at a time. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported

Property summary

The following table lists the properties that are either partially or not supported by the MovieClip class when using ActionScript for Flash Lite 2.x and later.

Property	Description	Support
<code>_droptarget</code>	Returns the absolute path in slash-syntax notation of the movie clip instance on which this movie clip was dropped. The <code>_droptarget</code> property always returns a path that starts with a slash (/). To compare the <code>_droptarget</code> property of an instance to a reference, use the <code>eval()</code> function to convert the returned value from slash syntax to a dot-syntax reference. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>menu</code>	Associates the specified ContextMenu object with the movie clip.	Not supported
<code>_quality</code>	Sets or retrieves the rendering quality used for a SWF file. Device fonts are always aliased and therefore are unaffected by the <code>_quality</code> property.	Partially supported
<code>trackAsMenu</code>	A Boolean value that indicates whether other buttons or movie clips can receive mouse release events. The <code>trackAsMenu</code> property lets you create menus. You can set the <code>trackAsMenu</code> property on any button or movie clip object. If the <code>trackAsMenu</code> property does not exist, the default behavior is false. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>useHandCursor</code>	A Boolean value that determines whether the hand icon appears when the mouse rolls over a movie clip.	Not supported
<code>_xmouse</code>	Returns the x coordinate of the mouse position. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported
<code>_ymouse</code>	Returns the y coordinate of the mouse position. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.	Partially supported

Event summary

The following table lists the event handlers that are partially supported by the MovieClip class when using ActionScript for Flash Lite 2.x and later.

Event Handler	Description
<code>onDragOut</code>	Invoked when the mouse button is pressed and the pointer rolls outside the object. You must define a function that executes when the event handler is invoked. You can define the function on the timeline or in a class file that extends the MovieClip class or is linked to a symbol in the library. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
<code>onDragOver</code>	Invoked when the pointer is dragged outside and then over the movie clip. You must define a function that executes when the event handler is invoked. You can define the function on the timeline or in a class file that extends the MovieClip class or is linked to a symbol in the library. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
<code>onMouseDown</code>	Invoked when the left mouse button is pressed. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
<code>onMouseMove</code>	Invoked every time the mouse moves. Limitations: Supported if <code>System.capabilities.hasMouse</code> is set to true.
<code>onMouseUp</code>	Invoked every time the left mouse button is pressed. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.
<code>onReleaseOutside</code>	Invoked when the mouse button is pressed over a movie clip and released while the pointer is outside the movie clip's area. Limitations: Supported if <code>System.capabilities.hasMouse</code> or <code>System.capabilities.hasStylus</code> is set to true.

NetConnection class

The NetConnection class lets you create an object that you can use with a NetStream object to invoke commands on a remote application server or to play back streaming Flash Video (FLV) files either locally or from a server. Support for this class was added in Flash Lite 3.0.

For more information about the NetConnection class, see the following:

- Chapter 5, “Working with Video and Images,” in *Developing Flash Lite 2.x and 3.0 Applications*
- “Working with Images, Sound and Video” in *Learning ActionScript 2.0 in Adobe Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

NetStream class

The NetStream class lets you create a stream that can be used with a NetConnection object to play FLV files from a local file system, an HTTP address, or a Flash Media Server. Support for this class was added in Flash Lite 3.0.

For more information about the NetStream class, see the following:

- Chapter 5, “Working with Video and Images,” in *Developing Flash Lite 2.x and 3.0 Applications*
- “Working with Images, Sound and Video” in *Learning ActionScript 2.0 in Adobe Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Property summary

The following table lists the property that is unsupported by the NetStream class when using ActionScript for Flash Lite 3.0.

Property	Description
<code>checkPolicyFile</code>	Specifies whether the player should attempt to download a cross-domain policy file from the loaded FLV file’s server before beginning to load the FLV file itself.

Selection class

The Selection class lets you set and control the text field in which the insertion point is located (that is, the field that has focus).

Partial support for the Selection class was added to support inline text input in Flash Lite 2.1. For Flash Lite, the Selection object is valid only when a device supports inline text entry. If a device does not support inline text entry, and instead relies on an FEP (front-end processor) to enter text, all calls to the Selection object are ignored.

For more information about the Selection class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods that are unsupported by the Selection class when using ActionScript for Flash Lite 2.x and later.

Method	Description
<code>getBeginIndex()</code>	Returns the index at the beginning of the currently focused selection span.
<code>getCaretIndex()</code>	Returns the index of the blinking insertion point (caret) position.
<code>getEndIndex</code>	Returns the index at the end of the currently focused selection span.

SharedObject class

A Flash Lite shared object, as implemented in the ActionScript SharedObject class, allows Flash Lite content to save data to the device when the application is closed and load the data from the device when the application is played again.

For more information about the SharedObject class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 14, “Creating Interaction with ActionScript” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods that are partially supported by the SharedObject class when using ActionScript for Flash Lite 2.x and later.

Method	Description
<code>flush()</code>	Immediately writes a locally persistent shared object to a local file. Limitations: The write operation is asynchronous and the result is not immediately available.
<code>getLocal()</code>	Returns a reference to a locally persistent shared object that is available only to the current client. Limitations: In Flash Lite, a shared object cannot be shared between two SWF files.

Flash Lite method extensions

The following table lists the methods that have been added as extensions to the SharedObject class when using ActionScript for Flash Lite 2.x and later.

Method	Description
<code>GetMaxSize()</code>	Flash Lite returns the maximum size allotted for persistent storage of a SWF file.

System class

The System class contains properties that are related to certain operations on the user's computer, such as operations with shared objects, local settings for cameras and microphones, and using the Clipboard. The following additional properties and methods are in specific classes within the System package: the capabilities class, the security class, and the IME class.

For more information about the System class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- *Flash Lite 2.x and 3.0 ActionScript Language Reference*
- Chapter 7, “Classes,” in *Learning ActionScript 2.0 in Flash*

Method summary

The following table lists the methods that are not supported by the System class when using ActionScript for Flash Lite 2.x and later.

Method	Description	Support
<code>setClipboard()</code>	Replaces the contents of the clipboard with a specified text string.	Not supported
<code>showSettings()</code>	Shows the specified Flash Player Settings panel.	Not supported

Property summary

The following table lists the properties that are not supported by the System class for use with ActionScript for Flash Lite 2.x and later.

Property	Description	Support
<code>exactSettings</code>	Specifies whether to use superdomain or exact-domain matching rules when accessing local settings.	Not supported

Event summary

The following table lists the event handlers that are partially supported by the System class for use with ActionScript for Flash Lite 2.x and later.

Event handler	Description	Support
onStatus	Provides a super event handler for certain objects. The SharedObject property is supported. The LocalConnection and NetStream properties are not supported.	Partially supported

capabilities (System.capabilities) class

The System.capabilities class determines the abilities of the system and player hosting a SWF file, which lets you tailor content for different formats. For example, the screen of a cell phone (black and white, 100 square pixels) is different than the 1000-square-pixel color PC screen. To provide appropriate content to as many users as possible, you can use the System.capabilities object to determine the type of device a user has. You can then either specify to the server to send different SWF files based on the device capabilities or tell the SWF file to alter its presentation based on the capabilities of the device.

Property summary

The following table lists the properties that are not supported by the System.capabilites class when using ActionScript for Flash Lite 2.x and later.

Property	Description
hasIME	Indicates whether the system has an input method editor (IME) installed.
manufacturer	A string that indicates the manufacturer of Flash Player, in the format "Adobe OSName" (OSName could be "Windows", "Macintosh", "Linux", or "Other OS Name").
pixelAspectRatio	Indicates the pixel aspect ratio of the screen.
playerType	Indicates the type of player: stand-alone, external, plug-in, or ActiveX.
screenColor	Indicates whether the screen is color, grayscale, or black and white.
screenDPI	Indicates the screen resolution, in pixels (such as dpi).
serverString	A URL-encoded string that specifies values for each System.capabilities property.

Flash Lite property extensions

The following table lists properties that are extensions to the `System.capabilities` class for use with ActionScript for Flash Lite 2.x and later.

Property	Description
<code>hasCompoundSound</code>	A read-only Boolean value that is <code>true</code> if the player can process compound sound data; <code>false</code> otherwise.
<code>hasEmail</code>	A read-only Boolean value that is <code>true</code> if the player can send e-mail messages using the <code>GetURL</code> ActionScript command; <code>false</code> otherwise.
<code>hasMMS</code>	A read-only Boolean value that is <code>true</code> if the player can send multimedia message service (MMS) messages using the <code>GetURL</code> ActionScript command; <code>false</code> otherwise.
<code>hasSMS</code>	A read-only value whose variable <code>_capSMS</code> indicates whether Flash Lite can send short message service (SMS) messages using the <code>GetURL</code> ActionScript command. If Flash Lite can send SMS messages, this variable is defined and has a value of 1. Otherwise, this variable is not defined.
<code>hasMFI</code>	A read-only Boolean value that is <code>true</code> if the player can play sound data in the Melody Format for i-mode (MFI) audio format; <code>false</code> otherwise.
<code>hasMIDI</code>	A read-only Boolean value that is <code>true</code> if the player can play sound data in the MIDI audio format; <code>false</code> otherwise.
<code>hasSMAF</code>	A read-only Boolean value that is <code>true</code> if the player can play sound data in the Synthetic music Mobile Application Format (SMAF); <code>false</code> otherwise.
<code>hasDataLoading</code>	A read-only Boolean value that is <code>true</code> if the player can dynamically load additional data through calls to <code>loadMovie()</code> , <code>loadMovieNum()</code> , <code>loadVariables()</code> , <code>loadVariablesNum()</code> , <code>XML.parseXML()</code> , <code>Sound.loadSound()</code> , <code>MovieClip.loadVariables()</code> , <code>MovieClip.loadMovie()</code> , <code>MovieClipLoader.loadClip()</code> , <code>LoadVars.load()</code> , and <code>LoadVars.sendAndLoad()</code> ; <code>false</code> otherwise.
<code>has4WayKeyAS</code>	A read-only Boolean value that is <code>true</code> if the player can execute ActionScript attached to <code>keyEvent</code> handlers associated with the right, left, up and down keys; <code>false</code> otherwise. If the value of this variable is <code>true</code> , when one of the four-way keys is pressed, the player first looks for a handler for that key. If none is found, Flash control navigation is performed. However, if an event handler is found, no navigation action occurs for that key. In other words, the presence of a keypress handler for a down key disables the ability to navigate down.

Property	Description
hasMouse	A read-only Boolean value that is <code>true</code> if the player can send mouse-related events and <code>false</code> if the platform does not support a mouse.
hasMappableSoftKeys	Allows user to set soft-key values and handle events from those soft keys.
hasStylus	A read-only Boolean value that is <code>true</code> if the player can send stylus-related events and <code>false</code> if the platform does not support a stylus. The <code>onMouseMove</code> event is not supported by a stylus. This flag allows the content to check if this event is supported.
hasCMIDI	A read-only Boolean value that is <code>true</code> if the platform supports CMIDI sound, and <code>false</code> if the platform does not support CMIDI sound.
hasXMLSocket	(Added in Flash Lite 2.1) Indicates whether the host application supports XML sockets.
softKeyCount	A number specifying the number of soft keys that the platform supports.
hasSharedObjects	A read-only Boolean value that is <code>true</code> if Flash content playing back in this application can access Flash Lite shared objects; <code>false</code> otherwise.
hasQWERTYKeyboard	A read-only Boolean value that is <code>true</code> if ActionScript can be attached to all keys found on a standard QWERTY keyboard and the Backspace key; <code>false</code> otherwise.
audioMIMETypes	A read-only property that contains an array of MIME types that the device's audio codecs support and that can be used by the ActionScript Sound object.
imageMIMETypes	A read-only property that contains an array of MIME types that the device's image codecs support and that can be used by the <code>loadMovie</code> ActionScript function.
videoMIMETypes	A read-only property that contains an array of MIME types that the device's video codecs support and that can be used by the ActionScript Video object.
MIMETypes	A read-only property that contains an array of all the MIME types that the Sound and Video objects support and that can be used by the <code>loadMovie()</code> ActionScript function.

Sound class

ActionScript for Flash Lite 2.x and later supports device sound through the Sound class and through System.capabilities values. The Sound class is fully supported for native sounds supported in Flash Player 7, but it is only partially supported for device sounds.

Flash Lite 2.x added support that lets you synchronize device sound playback with rendering animation.

NOTE

Flash Lite 2.x and later does not support sound recording.

Method summary

The following table lists the methods that are partially supported by the Sound class for when using ActionScript for Flash Lite 2.x and later.

Method	Description	Support
<code>getPan()</code>	Returns the pan level set in the last <code>setPan()</code> call as an integer from -100 (left) to +100 (right). (0 sets the left and right channels equally.) The pan setting controls the left-right balance of the current and future sounds in a SWF file. Limitations: Supported for use with native Flash sound; not supported for use with device sound.	Partially supported
<code>getTransform()</code>	Returns the sound transform information for the specified Sound object set with the last <code>Sound.setTransform()</code> call. Limitations: Supported for use with native Flash sound; not supported for use with device sound.	Partially supported
<code>loadSound()</code>	Loads an MP3 file into a Sound object. You can use the <code>isStreaming</code> parameter to indicate whether the sound is an event or a streaming sound. Event sounds are completely loaded before they play. They are managed by the ActionScript Sound class and respond to all methods and properties of this class. Limitations: The streaming parameter is ignored when used with Flash Lite 2.x and later.	Partially supported

Method	Description	Support
setPan()	Determines how the sound is played in the left and right channels (speakers). For mono sounds, pan determines which speaker (left or right) the sound plays through. Limitations: Supported for use with native Flash sound; not supported for use with device sound.	Partially supported
setTransform()	Sets the sound transform (or balance) information for a Sound object. The <code>soundTransformObject</code> parameter is an object that you create using the constructor method of the generic Object class, with parameters specifying how the sound is distributed to the left and right channels (speakers). Limitations: Supported for use with native Flash sound; not supported for use with device sound.	Partially supported
setVolume()	Sets the volume for the Sound object. Limitations: Supported for use with native Flash sound; not supported for use with device sound.	Partially supported

Property summary

The following table lists the properties of the Sound class that are partially supported when using ActionScript for Flash Lite 2.x and later.

Property	Description
duration	The duration of a sound, in milliseconds. Limitations: Supported for use with native Flash sound; not supported for use with device sound.
position	The number of milliseconds a sound has been playing. Limitations: Supported for use with native Flash sound; not supported for use with device sound.

Flash Lite method extensions

The following table lists new methods in the Sound class that are specific to ActionScript for Flash Lite 2.x and later.

Method	Description
<code>getPan()</code>	Returns the value of the previous <code>setPan()</code> call. This method is not supported for device sound.
<code>getTransform()</code>	Returns the value of the previous <code>setTransform()</code> call. This method is not supported for device sound.
<code>loadSound()</code>	Loads sound data of any format into Flash Player. This method is different from the Flash Player 7 implementation because sound data loaded using this method is always treated as event sound. Therefore, the second parameter of this method is always ignored. In the following example, the value <code>true</code> is ignored: <pre>my_sound.loadSound("mysnd.mp3", true);</pre>

Stage class

The Stage class is a top-level class whose methods, properties, and handlers you can access without using a constructor. Use the methods and properties of this class to access and manipulate information about the boundaries of a SWF file.

For more information, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Flash Lite property extensions

The following table lists new properties in the Stage class that are specific to ActionScript for Flash Lite 2.x and later.

Property	Description
<code>showMenu</code>	Shows or hides the default items in the Flash Player context menu.

Property	Description
<code>getTransform()</code>	Returns the value of the previous <code>setTransform()</code> call. This method is not supported for device sound.
<code>loadSound()</code>	Loads sound data of any format into Flash Player. This method is different from the Flash Player 7 version because sound data loaded using this method is always treated as event sound, so the second parameter of this method is always ignored. For example, in the following call, the value <code>true</code> is ignored: <pre>my_sound.loadSound("mysnd.mp3", true);</pre>

TextField class

Text fields are visual elements on the Stage that let you display text to a user. Similar to an input text field or text area form control in HTML, Flash lets you set text fields as editable (read-only), allow HTML formatting, and enable multiline support.

You use the `TextField` class to create text fields. All dynamic and input text fields in a SWF file are instances of the `TextField` class. You can give a text field an instance name in the Property inspector and use the methods and properties of the `TextField` class to manipulate it with ActionScript. `TextField` instance names are displayed in the Movie Explorer and in the Insert Target Path dialog box in the Actions panel.

To create a text field dynamically, do not use the `NEW` operator; instead, use `MovieClip.createTextField()`.

The methods of the `TextField` class let you set, select, and manipulate text in a dynamic or input text field that you create during authoring or at runtime.

All properties of the `TextField` class are supported in Flash Lite 2.x and later, but you can only use text fields to display device fonts. Device fonts are special fonts in Flash that are not embedded in a SWF file. Flash Lite uses whatever font on the mobile device most closely resembles the device font. Because font outlines are not embedded, a SWF file size is smaller than using embedded font outlines. However, because device fonts are not embedded, the text that you create with these fonts looks different than expected on devices that do not have a font installed that corresponds to the device font. Flash includes three device fonts: `_sans` (similar to Helvetica and Arial), `_serif` (similar to Times Roman), and `_typewriter` (similar to Courier).

For more information about the `TextField` class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 12, “Working with Text and Strings,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Property summary

The following table lists the properties of the TextField class that are not supported when using ActionScript for Flash Lite 2.x and later.

Property	Description
<code>antialiasType</code>	Indicates the type of anti-aliasing used for this TextField instance.
<code>menu</code>	Associates a ContextMenu object with a text field
<code>mouseWheelEnabled</code>	A Boolean value that indicates whether Flash Player should automatically scroll multiline text fields when the mouse pointer clicks a text field and the user rolls the mouse wheel.
<code>restrict</code>	Indicates the set of characters a user can enter into the text field.
<code>sharpness</code>	The sharpness of the glyph edges in this TextField instance.
<code>styleSheet</code>	Attaches a style sheet to the text field.
<code>thickness</code>	Indicates the thickness of the glyph edges in this TextField instance.

Method summary

The following table lists the methods of the TextField class that are not supported when using ActionScript for Flash Lite 2.x and later.

Method	Description
<code>getFontList</code>	Returns the names of fonts on the player's host system as an array.

TextFormat class

The TextFormat class represents character formatting information. Use the TextFormat class to create specific text formatting for text fields. You can apply text formatting to static and dynamic text fields. Some properties of the TextFormat class are not available for embedded and device fonts.

The TextFormat class lets you apply formatting to a text field or to certain characters within a text field. Some examples of text formatting options that can be applied to text are alignment, indenting, bold, color, font size, margin widths, italics, and letter spacing. You can apply text formatting to static and dynamic text fields. Some properties of the TextFormat class are not available for embedded and device fonts.

You must use the constructor `TextFormat()` to create a `TextFormat` object before calling its methods.

NOTE

Flash Lite 2.x and later provides partial support for the formatting feature available in the `TextFormat` class. Formatting features are not available when you use device fonts.

Flash Lite 2.x and later provides partial support for the `TextFormat` class. For example, `TextFormat.font`, `TextFormat.bold`, and `TextFormat.tabstop` are not supported when you use device fonts.

For more information about the `TextFormat` class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 12, “Working with Text and Strings,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Property summary

The following table lists the properties that are partially supported when using ActionScript 2.0 to create Flash content.

Property	Description
<code>bold</code>	A Boolean value that specifies whether the text is boldface. Limitations: For use only with paragraph-level formatting; you cannot apply bold to individual characters.
<code>bullet</code>	A Boolean value that indicates that the text is part of a bulleted list. Limitations: For use only with embedded fonts.
<code>color</code>	Indicates the color of text. Limitations: For use only with paragraph-level formatting; you cannot apply color to individual characters.
<code>font</code>	The name of the font for text in this text format, as a string. Limitations: For Flash Lite, this property works for embedded fonts only. This property is not supported for Arabic, Hebrew, and Thai.
<code>italic</code>	A Boolean value that indicates whether text in this text format is italic. Limitations: For use only with paragraph-level formatting; you cannot apply italics to individual characters.
<code>size</code>	The point size of text in this text format. Limitations: For use only with paragraph-level formatting; you cannot apply different font sizes to individual characters.
<code>tabStops</code>	Specifies custom tab stops as an array of non-negative integers. Limitations: For use with embedded fonts only.
<code>underline</code>	A Boolean value that indicates whether the text that uses this text format is underlined (<code>true</code>) or not (<code>false</code>). Limitations: For use only with paragraph-level formatting.

Video class

Flash Lite 2.x lets you work with device-specific video formats, and supports the following types of video playback:

- Video embedded in a SWF file
- Video available as a separate file on the device
- Video streamed over the network (in real time)

Flash Lite 2.x supports device video. Device video is stored in the published SWF file in the device's native video format. To play the device video, Flash Lite passes the video data to the device, which then decodes and plays the video.

NOTE

Flash Lite 2.x ActionScript does not support the `NetConnection` or `NetStream` classes.

Flash Lite 3.0 adds support for Flash Video (FLV) using versions of the On2 and Sorenson codecs optimized for mobile devices. FLV video is rendered directly in the Flash Lite player rather than by the device, so you no longer need to be concerned about whether your target devices support a particular video format. The ActionScript `NetConnection` and `NetStream` classes, which were not previously available in Flash Lite, let you control the playback of FLV video from a local drive or HTTP address. These classes are described in Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*. Streaming video data over an RTMP connection to a Flash Media Server is also supported in Flash Lite 3.0 (but RTMPT and RTMPS connections are not). The `Camera` class and recording video are not supported for Flash Video.

Flash Lite 3.0 also includes a new property in the `Video` class, `attachVideo`, that specifies a video stream to be displayed within the boundaries of the `Video` object on the Stage. You use the methods of the `NetStream` class rather than those of the `Video` class to control playback of FLV (non-device) video. For example, to pause device video, you use the `Video.pause` method, but to pause FLV video, you use `NetStream.pause`.

For more information about the `Video` class, see the following references:

- *Developing Flash Lite 2.x and 3.0 Applications*
- Chapter 15, “Working with images, Sound, and Video,” in *Learning ActionScript 2.0 in Flash*
- Chapter 2, “ActionScript Classes,” in *Flash Lite 2.x and 3.0 ActionScript Language Reference*

Method summary

The following table lists the methods that are not supported by the `Video` class when using ActionScript for Flash Lite 2.x and later.

Method	Description	Support
<code>clear</code>	Clears the image currently displayed in the <code>Video</code> object. This is useful when, for example, you want to display standby information without having to hide the <code>Video</code> object.	Not supported

Property summary

The following table lists the properties of the Video class that are not supported when using ActionScript for Flash Lite 2.x.

Property	Description
<code>deblocking</code>	Indicates the type of deblocking filter applied to decoded video as part of postprocessing. Two deblocking filters are available: one in the Sorenson codec and one in the On2 VP6 codec.
<code>height</code>	An integer specifying the height of the video stream, in pixels.
<code>smoothing</code>	Specifies whether the video should be smoothed (interpolated) when it is scaled. For smoothing to work, the player must be in high-quality mode.
<code>width</code>	An integer specifying the width of the video stream, in pixels.

Flash Lite method extensions

The Video class for Flash Lite 2.x adds the following new methods.

Method	Description
<code>play()</code>	Opens a video source and begins playing the video.
<code>close()</code>	Stops playing the video, frees the memory associated with this Video object, and clears the Video area onscreen.
<code>stop()</code>	Stops playing the video and continues to render the current frame onscreen. A subsequent call to <code>Video.resume()</code> resumes playing from the first frame of the video.
<code>pause()</code>	Stops playing the video and continues to render the current frame onscreen. A subsequent call to <code>Video.resume()</code> resumes playing from the current position.
<code>resume()</code>	Resumes playing the video.

The Video class for Flash Lite 3.0 adds the following new method.

Method	Description
<code>attachVideo()</code>	Specifies a video stream (source) to be displayed within the boundaries of the Video object on the Stage.

Index

A

ActionScript 2.x and 1.x
differences between 5

B

Button class
 unsupported or partially supported event handlers
 13
 unsupported or partially supported properties 12

D

Date class
 unsupported or partially supported methods 14

F

Flash Player 4 and Flash Player 7
 differences between 5
fsCommand() and fsCommand2() 7
 partially supported commands 8
 unsupported commands 8

G

global properties
 partially supported 10

K

Key class
 new methods 15
 new properties 15

M

Mouse class
 unsupported or partially supported events 17
 unsupported or partially supported methods 16
MovieClip class
 partially supported event handlers 20
 unsupported or partially supported methods 18

S

Selection class 22
 unsupported methods 21, 22
SharedObject class
 new methods 23
 partially supported methods 23
Sound class
 new methods 30
 partially supported methods 28
 partially supported properties 29
Stage class
 new properties 30
System class
 partially supported events 25
 unsupported methods 24
 unsupported properties 24
System.capabilities class
 new properties 26
 unsupported properties 25

T

TextField class
 unsupported methods 32
 unsupported properties 32
TextFormat class
 partially supported properties 34

V

Video class

new methods 36

unsupported methods 35

unsupported properties 36